Università degli Studi di Napoli Federico II

Dipartimento di Scienze Economiche e Statistiche
Dottorato in Economia – XXXII ciclo

Machine Learning Methods and Applications in Economics

Candidato
Luca Coraggio

Tutor
Prof. Marco Pagano
Prof. Pietro Coretto

Coordinatore
Prof. Marco Pagano

Alla mia famiglia. Grazie.

Machine Learning Methods and Applications in Economics

Luca Coraggio

Contents

Chapter 1

Introduction to Machine Learning

1.1 Introduction

Nowadays, there is a huge, growing interest in Machine Learning (see Figure 1.1). This field became known to the large public relatively recently, but its roots go deep into the past century. General interest is motivated by popular applications like: machine learning algorithms beating the Go world champion (Silver et al., 2016 and Silver et al., 2017); self-driving cars being safer than human drivers (Teoh and Kidd, 2017); algorithms for facial recognition achieving astonishing accuracy rates (Parkhi, Vedaldi, and Zisserman, 2015). The excitement for these and other applications made the name for algorithms being capable of "superhuman" intelligence.

Aside from these popular applications, the interest in these methodologies is rightfully motivated by the fact that, viewed as a general purpose technology, they are literally (re)shaping our society: web search engines; advertisements; customers profiling; social media; financial systems; health sector and many others. All of these applications use some machine learning algorithms, and some would likely not be possible without it. These algorithms make it possible to analyse huge sets of data efficiently and effectively, and this motivates why they are interesting to Economics.

There are two main lens under which Economics may look at Machine Learning. The first one is analysing and understanding the impact that these new technologies are having on our societies. Machine Learning is part of what is sometimes referred to as the fourth industrial revolution,[1] and it is having and will have a tremendous impact on economic growth, inequalities, productivity, innovations, employment, competition, consumers' demand, etc. Also, this poses serious problems for regulations. For example, Calvano et al., 2018 show that in some type of algorithmic pricing[2] the algorithms learn to collude with one another; this clearly undermines competition and raise regulatory issues.

The second point that motivates economists' interest in Machine Learning resides in applied economic research. In fact, machine learning methodologies have a lot to bring to more traditional empirical research. We will focus on this second aspect in this work. For an excellent, broad collection containing discussions on the impact of Machine Learning from the perspective of

[1] This term is first attributed to the Economist Klaus Schwab (Schwab, 2015).

[2] The paper analyses Q-learning, a type of reinforcement learning algorithm, which is designed to maximize a stream of rewards based on a set of actions. Simply put, the algorithm learns to play the best action.

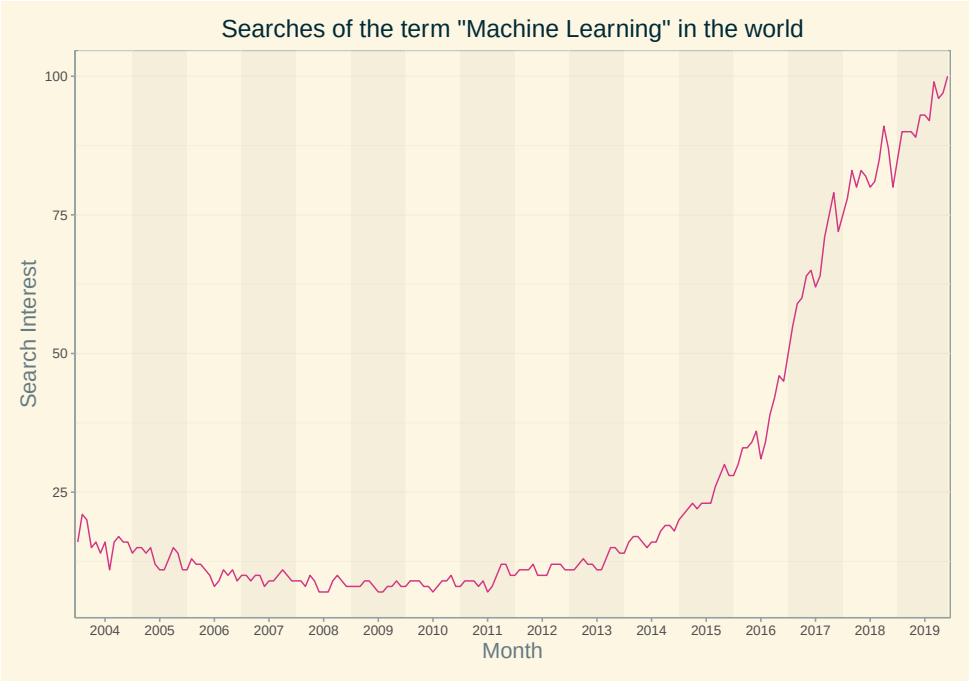


Figure 1.1: Web searches for the term "Machine Learning" (on Google's browsers) across all over the world, since 2004. Values on the y-axis are between 0 (insufficient data) to 100 (highest amount of searches). The graph shows a clear increasing interest in the topic as proxied by the frequency of the searches. Source: Google Trends (https://www.google.com/trends).

several different fields in Economics, see Goldfarb, Gans, and Agrawal, 2019.

In this introduction, we briefly introduce two broad categories of machine learning methods. Section 1.2 introduces supervised learning and unsupervised learning and briefly present different applications in Economics. In Section 1.3 we concludes the chapter detailing supervised and unsupervised learning approaches that will be functional for the contributions of the next two chapters. This introduction is by no means a comprehensive survey of the literatures. However, we will try to point out useful references and concepts relevant to the subsequent analyses.

1.2 Supervised and Unsupervised learning and Economics

In this section, we introduce both supervised and unsupervised learning, providing the general ideas and goals of the two. Then we are going to briefly motivate why they are of interest to applied research in Economics.

1.2.1 Supervised Learning

Supervised Learning (in the Machine Learning community); *Supervised Pattern Recognition*; *Classification*. These names all refer to the same problem, of which several definitions can be found.

> *The problem of searching for patterns in data is a fundamental one and has a long and successful history. For instance, the extensive astronomical observations of Tycho Brahe in the 16th century* [...]. *The field of pattern recognition is concerned with the automatic*

discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories. (Bishop, 2006)

In pattern recognition [...] *each object is assumed to belong to one of a known number of classes, whose characteristics have been determined using a training set, and the aim is to identify the class to which the object should be assigned.* (Gordon, 1999)

[...] *to extract important patterns and trends, and understand "what the data says." We call this learning from data.*

In supervised learning, the goal is to predict the value of an outcome measure based on a number of input measures.

In a typical scenario, we have an outcome measurement, usually quantitative (such as a stock price) or categorical (such as heart attack/no heart attack), that we wish to predict based on a set of features (such as diet and clinical measurements). We have a training set of data, in which we observe the outcome and feature measurements for a set of objects (such as people). Using this data we build a prediction model, or learner, which will enable us to predict the outcome for new unseen objects. A good learner is one that accurately predicts such an outcome. (Hastie, Tibshirani, and Friedman, 2009)

All of the above definitions share the elements peculiar to supervised learning. These basic ingredients, that are present in every classification task, are the following:

- A set of objects (data) with some measured characteristics (*features*, or to build a parallel with classical Econometrics terminology, these play the same role as the *regressors*.)
- An outcome variable of interest (*class labels*) associated/known for a subset of the objects. These labelled objects constitutes the *training set*.
- A learner/algorithm/model that, using the information of the training set, builds a mapping from the features to the outcome.
- A set of non-labelled objects to be classified using the mapping above.

The fourth point above is not essential to perform classification tasks, but motivates the need for supervised learning in the first place; there are two types of supervised learning tasks, namely *classification* and *regression*. A classification problem is one in which the outcome variable is a categorical one. On the other hand, in a regression problem the outcome variable is a quantitative one.

For a visual example, refer to Figure 1.2. Here the objects are geometric shapes of different size and number of sides, the color of the shapes is the outcome we are interested in. We would like to infer a rule to assign colors to objects. We can structure the data information in a matrix as shown in Table 1.1. Note that the size was coded as a dummy variable taking value 1 for big size shapes and 0 otherwise. This is typical in Machine Learning: categorical variables often needs to be encoded, and the type of encoding may affect the final classification performances and this

is not always a straightforward task (Micci-Barreca, 2001). Despite its simplicity, this example shows the archetypal supervised learning problem. In our example, the training set would consist of the first 13 objects (or observations) having a label (Color) attached. A learning algorithm would use the training information to infer a map m : Number of Sides $\times$ Size Big $\rightarrow$ Color. Every classification algorithm (e.g., classification trees, artificial neural network, multinomial logistic regression ...) will produce such a mapping, which depends both on the training data at hand and on the particular specification of the chosen algorithm. A "good learner" should be able to predict "magenta" color for the 14-th object. Note that a learner using only the information on size is likely to get the color wrong, assigning a "blue" label. This shows the importance of model complexity (or degrees of freedom): more complex models can usually adapt better to the data and have an higher predicting power on the training data (e.g. think about the R^2 in linear regression with as many regressors as there are observations). However, if too complex, a model can perform poorly on unseen data. This is known as the *bias variance trade-off* (Hastie, Tibshirani, and Friedman, 2009).

Usually, to control for this trade-off, it is common to split the training set in two or three parts (see Figure 1.3). A portion of the data is used to train the algorithm (train set); another portion is used to make out of sample prediction (validation set); (optionally) a third portion is used to make out of sample predictions after tuning (test set). The splitting is motivated as follows. Typically, every classification algorithm needs to be defined in its structural/architectural components. For example, in a regression model, we need to decide how many regressors to include, or if we use a penalized estimation method we need to decide the value of the penalty. Another example is the choice of nodes and layers in neural networks models.

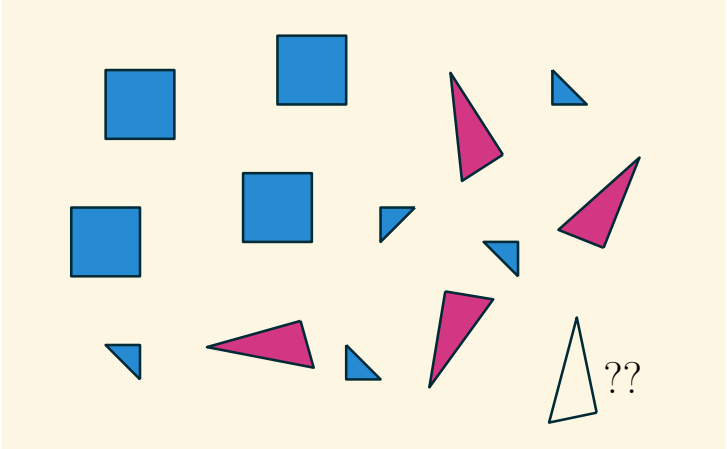


Figure 1.2: A toy example for supervised learning. The outcome variable of interest is the color (binary: magenta/blue). The shape size and number of sides are the features. The classification task is to infer the color of the non-colored object.

Table 1.1: Colored shapes dataset.

Object Index	Number of sides	Size Big	Color
1	4	1	Blue
2	4	1	Blue
3	4	1	Blue
4	4	1	Blue
5	3	0	Blue
6	3	0	Blue
7	3	0	Blue
8	3	0	Blue
9	3	0	Blue
10	3	1	Magenta
11	3	1	Magenta
12	3	1	Magenta
13	3	1	Magenta
14	4	1	??

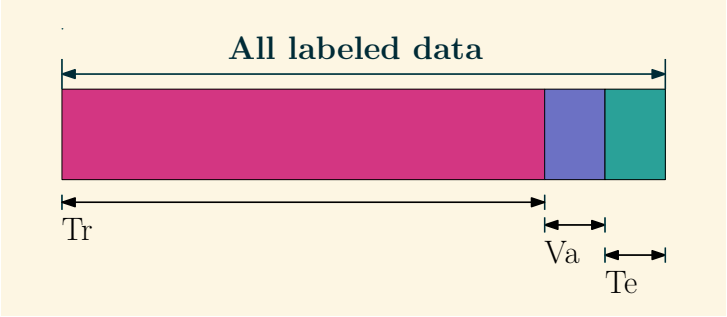


Figure 1.3: Train/Validation/Test split. Proportions vary according to the sample size. For moderate sample (> 1000 points), typical splits are 60/20/20 % or 50/25/25 %. For very large data, (> 1000000) 98/1/1 % might suffice.

These decisions are formalized in terms of quantities called *hyperparameters*, and these define the model complexity. Once these parameters are set, the algorithm is "trained" on the training set. To evaluate its out-of-sample performance, we use the validation set. If the performances are poor on both training set and validation set, it is likely that the classifier is either too simple, or the data are insufficient for the task provided. If the performances are good on the training set and bad on the validation set it is likely due to a lack of similarity between train and validation or to overfitting problems (i.e., the classifier is way too adapted to the observed data and generalizes poorly on unseen data).

The scores on the training and validation set give a rough indication on the overall quality of the classifier, which will in turn be adjusted in a trial and error fashion to improve the score on both sets. Since both sets are used in this optimization problem (the training for estimation of parameters and the validation for hyperparameter *tuning*[3]) a third set, the test set, is used for an unbiased assessment of the out-of-sample quality of the final classifier (note that no optimization is ever carried on this set). Tuning a classifier is not a simple task and tunable components differ from classifier to classifier. Several classifiers may then be compared and selected on their performances on the training and validation sets. There are several methods (more or less sophisticated) to perform this model selection task (see Arlot and Celisse, 2010 and Kohavi, 1995).

Now let us summarize the typical workflow in supervised learning as described so far:

- Define a research question.
- Collect labelled data; perform opportune preprocessing to structure data information.
- Define a set of competing models/learning algorithms to be trained on the data.
- Split the labelled data; train the models and obtain performance metrics; optimize the models with the aid of validation set.
- Select a model according to some criteria (as for example, the model maximizing a score function on the validation dataset).
- Get an unbiased estimate of model performance on a test set, and use the selected model to answer the research question.

1.2.2 Unsupervised learning

Clustering, or to immediately highlight its main difference with classification, *unsupervised learning* has a long history. Let us describe it via the words of some great contributors to the field.

> *Clustering is the grouping of similar objects.*
>
> *A cluster is a set of similar objects. The basic data in a clustering problem consist of a number of similarity judgements on a set of objects. The clustering operations attempt to*

[3] In general, the *tuning* of a classifier is the "optimizations" of its hyperparameters (determining complexity and architecture) in order to improve results both on validation and training set

> *represent these similarity judgements accurately in terms of standard similarity structure as a partition or a tree.* (Hartigan, 1975, pp. 1; 9.)

> [...] *the classes are not known at the start of the investigation: the number of classes, their defining characteristics and their constituent objects all require to be determined.* (Gordon, 1999, p. 3.)

> *Informally speaking, clustering means finding groups in data. Aristotele's classification of living things was one of the first-known clusterings, a hierarchical clustering.*

> *The aim of cluster analysis is often described as collecting similar objects in the same cluster and having large dissimilarity between objects in different clusters.* (Hennig et al., 2015, pp. 2; 7.)

The basic characteristics of a clustering problem are as follows.

- A set of objects with some measured characteristics (*variables* or *features*).
- An definition of what type of clustering we require. That is, what is a "good" grouping of objects? To what criteria should it respond to?
- A measure of similarity between objects coherent with the required clustering.
- A strategy (method) to cluster the objects together.

A fundamental aspect, which dramatically separates supervised and unsupervised learning, is that the we do not have any "truth" to use as a reference. In supervised learning, we had to classify points to a given number of classes. This immediately clarifies two aspects.

1. We know how many classes (or groups) we should look for.
2. We already have an understanding of what a good grouping is. This also guides intuition on how the objects should be studied to deliver the correct classification.

In clustering, we do not have a correct classification. There is no label, nor any underlying "truth" that guides the analysis. In the absence of classes (or groups) labels, how should the be objects grouped in the first place? How many groups (or classes) should we look for? Consider the example in Figure 1.4. The data is similar to that in Figure 1.2. Here we have either triangles or square in the space. We observe their location in the space (say an Euclidean bi-dimensional space) and their number of sides. The main difference is that we do not have class labels (actually, we do not even know what is a "class" in this case). The figure shows three possible groupings. We may want to define groups as being geometrical known objects, as square or triangles. In this case, we can use as a similarity criterion the number of sides of an objects. Thus, we obtain the third solution in Figure 1.4, which separates squares and triangles. Or maybe, we might not be interested in geometrical shapes, but in physical objects' proximity. Then, we may measure objects similarity by the Euclidean distance. In this case, the "good" clusters are shown by the first two solutions in the figure: in the first solution, we looked for two groups of most similar objects; in the second solution we looked for three groups. Now the question is: which of the

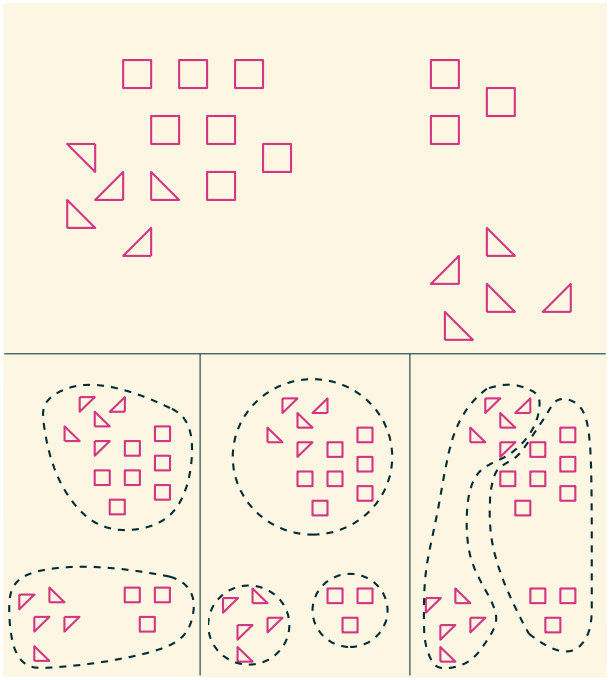


Figure 1.4: Example of a clustering problem (top) and three different clustering solution (bottom). The first two solutions are obtained by measuring objects distances and looking for 2 and 3 groups respectively. The third one is obtained by considering the number of sides of the objects. Groupings or clusters are obtained highlighted by the dashed line.

tree solutions is the best? As usual, it depends. All the three solutions seem reasonable, and all of them answer (reasonably well) to different clustering definitions. This example was used to briefly introduce the problem: what is a cluster? How do we retrieve clusters? How many clusters? Of course, there is no simple answer to these questions and there is a plethora of equally valid approaches to the problem. To be precise, this is not even the full picture. Up to now, we implicitly considered a cluster to be a partition of objects (into clusters). However, we might want to "softly" split objects into clusters. That is, we may want to assign a degree of class membership to objects for each class (also known as *fuzzy clustering*). Also, the number of clusters we look for in the data, might be suggested by the data themselves (this can be done via *clustering validation* and *clustering selection* methods). Finally, there are many different ways to approach clustering: hierarchical clustering; spectral clustering; density-based clustering; model based clustering; and others. Describing these different many approaches is not in the scope of this introduction. However, we point at Hennig et al., 2015 for further references. This is a very comprehensive collection, containing surveys on all the most relevant clustering approaches and techniques. Also, a clear and wide perspective is given by Jain, Murty, and Flynn, 1999, which is an extensive and systematic review. The authors introduce the problem and also provide a typical workflow. They make a distinction in two fundamental approaches to clustering (hierarchical and partitional; further distinction follows) and treat a number of more subtle issues (e.g. data representation, computational costs, ...). It is important to stress two points here:

> *There is no clustering technique that is universally applicable in uncovering the variety of structures present in multidimensional data sets.*
>
> [...] *clustering algorithms often contain implicit assumptions about cluster shape or multiple-cluster configurations based on the similarity measures and grouping criteria used.*
>
> (Jain, Murty, and Flynn, 1999)

1.2.3 Economics and Machine Learning

There are many ways in which the interaction between Economics and Machine Learning can be beneficial. Here we will illustrate some of them. We identify three main areas where Machine Learning offers interesting solutions to Economics. These may be summarized as follows.

- Off-the-shelf usage of machine learning algorithms in applied Economics.
- Literature on Machine Learning and Causal Inference.
- Model selection methodologies.

In the following, we are going to treat each of them in order.

Nowadays, the great and large availability of new type of data opens many possibilities for economic analysis. Usually, most of this information can not be processed with standard techniques from Econometrics. These are typically scenarios in which off-the-shelf machine learning methods can be applied. An excellent example of this are text analysis techniques. In general, with this term one refers to the collection of methods to analyse and process textual data. In fact, a large amount of them are unsupervised learning techniques. Gentzkow, Kelly, and Taddy, 2019 provide a systematic review of text analysis methodologies. These allow to extract information from text data and to summarize text similarities via, for example, distance-based clustering or model-based clustering. For example, Bandiera et al., 2017 cluster textual information on CEOs' activities collected at a frequency of 15 minutes blocks for one week. They are able to identify two different behaviours for them, which they call "managers" and "leaders", and also use this information in analysing performance of the hiring firm. In general, these techniques can be used to create new variables to be used in applied research.

Another example is satellite data. These are image data, which typically need machine learning methodologies to be efficiently processed. Jean et al., 2016 use a convolutional neural network on daytime and nighttime satellite data to identify features in the daytime images that correlates with socioeconomic indicators such as household consumption expenditure and asset wealth in African countries.

These are cases where machine learning methods are used to synthesize information from new data that would not have been possible to exploit otherwise (other examples can be found in Mullainathan and Spiess, 2017).

There are other cases where off-the-shelf ML methods can be applied improving on classical approaches. These are prediction problems, and machine learning algorithms have proven to be extremely good at prediction tasks. We distinguish three main type of prediction problems. The first one is prediction in policy problems. Kleinberg et al., 2015 gives several examples: predicting *which* teacher to hire for the most value (assessing *whether* to do it is a causality problem); evaluating creditworthiness of borrowers; assessing whether or not to detain an arrestee based on his/her probability of committing a crime; whether or not a patient should undergo medical surgery, etc. Another category of prediction problems is that of prediction in estimation tasks (Mullainathan and Spiess, 2017). For example, in instrumental variable two stage approach, the first stage of regressing the explanatory variable on the instrument can be seen as a prediction task. After all, only the predictions from this stage enter the subsequent one. Finally, the last

category is that of predictions used to test economic theories that make statements about predictability issues (e.g. efficient market hypotheses). For further insight see Mullainathan and Spiess, 2017.

Those described above are all cases in which plain vanilla machine learning methods can be applied without further modifications. However, ML methods are also interesting to the literature on Causal Inference. In effect, this concerns more supervised learning than unsupervised learning. Indeed, thinking at the standard regression problem, unsupervised learning has totally different goals as we saw in Subsection 1.2.2. The former is interested in the relationship among explanatory and outcome variables, the latter is interested in similarity between objects. On a different note, supervised learning seems much more closer to the regression framework. In fact, in this case we are willing to find a relationship between features and outcomes (see Subsection 1.2.1). Nonetheless, there is one substantial difference between the classical ML approach and the causal inference approach, namely their different focus: machine learning algorithms are typically optimized to make predictions, not causal statements. We can think of them as maximizing the R^2 (as a measure of goodness-of-fit) of a regression problem. On the other hand, in causal inference we are usually willing to trade off explanatory power with the identification of parameters, so as to being able to assess causal relationships. A typical example is as follows (Athey, 2018). Imagine to have available data on prices and hotels' occupancy rates. Typically, at higher prices are associated higher occupancy rates as hotels tend to increase prices when they fill up. Thus, price would be a good predictor of occupancy rate, so that if we were to make an estimate of the latter, looking at price would be a good choice. Nonetheless, if we were to run our hotel, we would never conclude from this that if we raised the prices enough we would immediately fill up all the vacancies. How occupancy rate changes as the price change is a causal inference question. Machine learning algorithms are typically meant to solve the prediction tasks instead (Mullainathan and Spiess, 2017 discuss this aspect in an extremely clear way with LASSO regression). Does this mean that ML algorithms can not be used for causal inference? The answer is no. In principle, there is no conceptual difference between the aim of a regression or that of a neural network: they both minimizes the sum of squared differences between a function of explanatory variables and the outcome variable. In practice, there are several reasons why ML algorithms can not be directly used for assessing causal relationships: (i) difficult interpretability for more complex methods (e.g. it is not really clear how one should interpret nodes' weights in a neural network); (ii) classical implementations of ML algorithms are pursuing a different target than what needed for causal inference; (iii) lack of an identification strategy for the parameters of interests. Upon solving these problems, ML methods could also be used in assessing causal relationships. Typically, this requires to either reformulate the objective function that the algorithm optimizes (for example this is done in trees used for treatment effects estimation proposed in Athey and Imbens, 2016) or to design carefully the employment existing methods (an example of this is the double LASSO procedure proposed in Belloni, Chernozhukov, and Hansen, 2014). Athey, 2018 reviews many different settings where the literature investigates the application of machine learning methodologies to classical causal inference problems. As the author argues, one important aspect is that, while it is likely that methodologies are going to

change to include more sophisticated ML algorithms, the basic identification strategies are going to stay unmodified. Only the latter can ensure the validity of causal inferences. We quote from Athey, 2018 the identification strategies that received attention from the literature:

- "treatment randomly assigned (experimental data)";
- "treatment assignment unconfounded (conditional on covariates)";
- "instrumental variables";
- "panel data settings (including difference-in-difference" design)";
- "structural models of individual or firm behaviour".

For each of these strategies, different problems of interests are: estimating average treatment effects, estimating heterogeneous treatment effect (parametrically and nonparametrically), estimating optimal treatment assignment policies and identifying groups of individuals that are similar in terms of their treatment effects. In these cases, advantages of machine learning methods are to be found in their higher degree of complexity with respect to standard methods and their ability to handle high dimensional data efficiently.

Finally, Machine Learning is also interesting to applied research in Economics for its systematic approaches to model selection. For example, methodologies as cross-validation (see also Subsection 1.3.2) can be extremely beneficial to researchers to aid the selection of a model specification, especially in those contexts where complex data structure are at hand. In such cases, it would be an hard task to fully document the procedure that led to the selection of a particular model. On the contrary, model selection methods from Machine Learning gives a data-driven, principled and systematic way to assess this, being also reproducible (see Athey, 2018).

1.3 Learning: introductory material

In this section, we review the supervised and unsupervised learning concepts introduced above. This discussion will set the basis for the next two chapters. The review here is extremely limited. We only introduce basic notions (e.g. misclassification rate, bias-variance trade-off, etc.), and methods useful for Chapers 2 and 3.

1.3.1 Probability models and related methods for clustering

McLachlan and Rathnayake, 2015 report that one of the first works adopting a model-based approach to cluster analysis is Scott and Symons, 1971. Since then this approach has been widely developed and applied. The well known McLachlan and Peel, 2000 book gives an exhaustive clear treatment of mixture models and related issues (not confined to cluster analysis only). Other good surveys on model-based clustering are given in: Fraley and Raftery, 1998 Fraley and Raftery, 2002, and McLachlan and Rathnayake, 2015 (which is also the opening reference of this section). We borrow from these sources to introduce model-based clustering.

Model-based clustering is a probabilistic approach. Probabilistic in the sense that we assume that there is a probabilistic model generating the data and we try to retrieve this model. The model itself has an implied clustering scheme. It is typical to assume that the underlying generating process is a mixture distribution specified by density:

$$f(x) := \sum_{k=1}^{K} \pi_k f_k(x), \tag{1.1}$$

where K is the number of mixture components, π_k are mixing proportions such that $\forall k\ 0 \leqslant \pi_k \leqslant 1$, and $\sum_k \pi_k = 1$, and f_k are density functions. For example, one typical formulation is the use of Gaussian density functions, so that for each k, f_k is represented by:

$$f_k(x) \equiv \phi_k(x) := \phi(x; \mu_k, \Sigma_k) = \left(\frac{1}{2\pi}\right)^{\frac{p}{2}} \left|\Sigma_k\right|^{-\frac{1}{2}} \exp\left\{-\frac{(x-\mu_k)'\Sigma_k^{-1}(x-\mu_k)}{2}\right\},$$

where $\phi(\,\cdot\,; \mu, \Sigma)$ indicates the (multivariate) Gaussian density with mean μ and (co)variance (matrix) Σ, p is data dimensionality, i.e. $x \in \mathbb{R}^p$. However, in general, densities are not restricted to be Gaussian (McLachlan and Peel, 2000). Then, we may identify groups or clusters with the mixture components, i.e. each component defines a cluster.

Figure 1.5 gives a graphical example. In the top figure, we see a two dimensional mixture distribution with 3 Gaussian components. In the bottom figures we see the implied clustering by some model-based method. Each component defines a single cluster, these are highlighted by different colors and are well separated. Thus, we say that a good clustering is the one that is implied by the mixture components or that mimics the mixture components. This is evident in case of well separated mixture components. However, when the components overlap, it may not be desirable to identify each cluster with a component. We will see this later.

Practically, one observes only points $\mathbb{X}_n = \{x_1, \ldots, x_n\}$. These are assumed to be realizations of independent and identically distributed random variable X_i, $i = 1, \ldots n$, with density function f (see (1.1)).[4] Usually, also the form of the components' density, f_k's, is assumed beforehand; everything else needs to be estimated. This is usually done via *maximum likelihood*, where the data likelihood is defined as:

$$L(\theta) := \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k f_k(x_i), \tag{1.2}$$

where θ collects all the parameters of the model, $\theta = [\pi_1, \mu_1, \Sigma_1, \ldots, \pi_K, \mu_K, \Sigma_K]$. Let $\theta^* = \arg\max_\theta L(\theta)$. This can be used to cluster data at hand by assigning points to the most "suitable" component described by θ^*. We will make this clear this in what follows. Analytic maximization of (1.2) over θ is not feasible in general, so that typically other approaches are needed, like the EM (Expectation Maximization) algorithm. In order to introduce the latter, we first need to rephrase the problem in terms of an incomplete information problem.

We will consider each of the X_i random variable as arising from one of the K components. Thus, consider n i.i.d. random vectors, Z_i, $i = 1, \ldots, n$, such that $Z_i = [Z_{i,1}, \ldots, Z_{i,K}]^T$, where each $Z_{i,k}$ can take value 0 or 1 with probability of being 1 equal to π_k, and $\sum_k Z_{i,k} = 1$. Thus,

[4] It would be more precise to say that a mixture distribution is hold to be good representation of the data at hand.

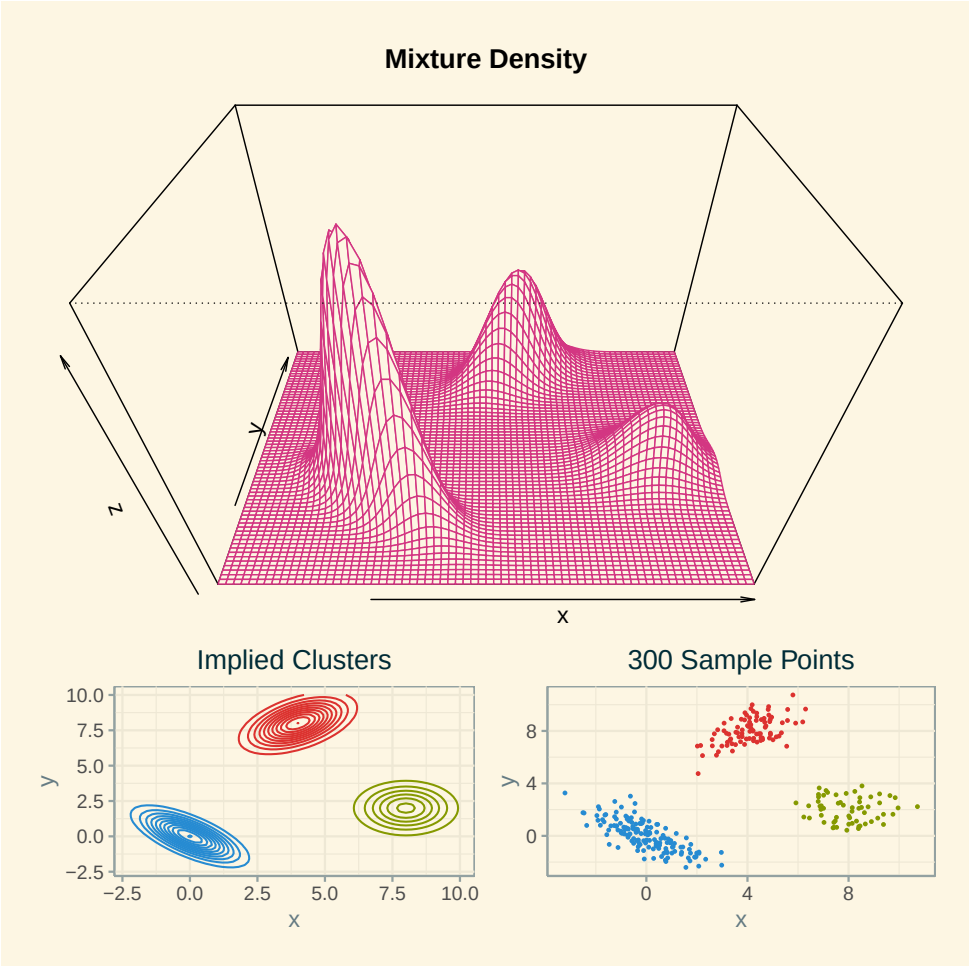


Figure 1.5: (Top) A 3 components Gaussian mixture distribution. (Bottom-Left) Contour sets of the 3 component Gaussian mixture; each component is highlighted with a different color. In model-based clustering we assume that each component constitutes a cluster. (Bottom-Right) A sample from the 3 components mixture; points are colored (clustered) as the generating mixture component.

for each i, $z = [z_1, \ldots, z_k]^T$, $z_k \in \{0, 1\}$, $\sum_k z_k = 1$:

$$P\{Z_i = z\} = \prod_{k=1}^{K} \pi_k^{z_k},$$

so that Z_i is a draw from a multinomial distribution with probabilities π_k's. Conditioning on $Z_{i,k} = 1$ we assume that X_i has density f_k. Thus, the unconditional distribution of X_i in given by (1.1), so that this framework gives the mixture model introduced above. Had we observed the full information, we could compute the complete data likelihood:

$$CL(\theta) := \prod_{i=1}^{n} \prod_{k=1}^{K} \Big(\pi_k f_k(x_i)\Big)^{z_{i,k}}, \tag{1.3}$$

where

$$z_{i,k} := \begin{cases} 1 & \text{if } x_i \text{ was drawn form component } k, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the observed data, $\{x_1, \ldots, x_n\}$, is said to be incomplete because we miss the information on $z_{i,k}$. The complete data is $\{(x_1, z_1), \ldots, (x_n, z_n)\}$ ($z_i = [z_{i,1}, \ldots, z_{i,K}]^T$), where the indicator variables, z_k's, are realization of the variable Z_i introduced above (see McLachlan and Peel, 2000).

It is preferable to work with logarithmic transformation of the above quantities, so that:

$$l(\theta) := \log L(\theta) = \sum_{i=1}^{n} \log \Big(\sum_{k=1}^{K} \pi_k f_k(x_i) \Big), \tag{1.4}$$

$$cl(\theta) := \log CL(\theta) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{i,k} \log \big(\pi_k f_k(x_i) \big). \tag{1.5}$$

Now consider the following (Biernacki, Celeux, and Govaert, 2000):

$$\begin{aligned} l(\theta) - cl(\theta) &= \sum_{i=1}^{n} \log \Big(\sum_{k=1}^{K} \pi_k f_k(x_i) \Big) - \sum_{i=1}^{n} \sum_{k=1}^{K} z_{i,k} \log \big(\pi_k f_k(x_i) \big) = \\ &\sum_{i=1}^{n} \Bigg(\log \Big(\sum_{k=1}^{K} \pi_k f_k(x_i) \Big) - \sum_{k=1}^{K} z_{i,k} \log \big(\pi_k f_k(x_i) \big) \Bigg) = \\ &\sum_{i=1}^{n} \log \left(\frac{\sum_{k=1}^{K} \pi_k f_k(x_i)}{\prod_{k=1}^{K} \big(\pi_k f_k(x_i) \big)^{z_{i,k}}} \right) = - \sum_{i=1}^{n} \log \left(\frac{\prod_{k=1}^{K} \big(\pi_k f_k(x_i) \big)^{z_{i,k}}}{\sum_{k=1}^{K} \pi_k f_k(x_i)} \right) = \\ &- \sum_{i=1}^{n} \sum_{k=1}^{K} z_{i,k} \log \left(\frac{\pi_k f_k(x_i)}{\sum_{k=1}^{K} \pi_k f_k(x_i)} \right) = - \sum_{i=1}^{n} \sum_{k=1}^{K} z_{i,k} \log(\tau_{i,k}) =: -h(\theta). \end{aligned}$$

It is possible to show that τ's are the expected values of the variables Z's given observed data and parameters; that is the posterior distribution of the indicator variables (McLachlan and Peel, 2000):

$$\tau_{i,k} = \tau_k(x_i; \theta) := \frac{\pi_k f_k(x_i)}{\sum_{k=1}^{K} \pi_k f_k(x_i)} = \mathbb{E} \left\{ Z_{i,k} = 1 | X_i = x_i, \theta \right\}. \tag{1.6}$$

Therefore, the sample log-likelihood function can be decomposed as:

$$l(\theta) = cl(\theta) - h(\theta).$$

We can not maximize $l(\theta)$ yet, since the z's are still unknown. Finally, consider taking the expectation over the indicator variables conditional on data and parameter vector θ' (i.e. $z_{i,k}$ should be replaced by $Z_{i,k}$, and we are going to take expectation along these variables; see Redner and Walker, 1984 or McLachlan and Krishnan, 2007):

$$\mathbb{E} \left\{ l(\theta) | \mathbb{X}_n; \theta' \right\} = l(\theta) = Q(\theta|\theta') - H(\theta|\theta') = \mathbb{E} \left\{ cl(\theta) - h(\theta) | \mathbb{X}_n; \theta' \right\}, \tag{1.7}$$

where, making explicit the parameters θ or θ' with respect to which the quantities are computed,

$$\mathbb{E}\left\{cl(\theta)|\mathbb{X}_n;\theta'\right\} := Q(\theta|\theta') = \sum_{i=1}^{n}\sum_{k=1}^{K}\tau_{i,k}(\theta')\log\left(\pi_k(\theta)f_k(x_i;\theta)\right),$$
$$\mathbb{E}\left\{h(\theta)|\mathbb{X}_n;\theta'\right\} := H(\theta|\theta') = \sum_{i=1}^{n}\sum_{k=1}^{K}\tau_{i,k}(\theta')\log(\tau_{i,k}(\theta)).$$

These quantities can be computed and used to maximize $l(\theta)$. The EM algorithm does this by maximizing $Q(\theta|\theta')$ in an iterative fashion, as shown in algorithm 1, which also details the closed form solutions for the Gaussian case.[5] The procedure will give an estimate, $\hat{\theta}^{(*EM)}$, of θ^*.

It is possible to show that $l(\theta^{(i+1)}) \geqslant l(\theta^{(i)})$, so that at every iteration the likelihood function is monotonically increasing (Dempster, Laird, and Rubin, 1977). In general, for a sequence of likelihood values $l(\theta^t)$ bounded from above, the EM algorithm is guaranteed to converge to some value of the likelihood $l(\tilde{\theta}^*)$ that is also guaranteed to be a stationary point. This, however, need not to be a local or global maximum (see McLachlan and Krishnan, 2007). Under additional assumptions the convergence to global or local maxima for the EM algorithm can be established (Boyles, 1983; Redner and Walker, 1984; McLachlan and Krishnan, 2007).

The EM algorithm is widely used in practice but may present drawbacks (Fraley and Raftery, 1998): conditions ensuring convergence to optima are usually not verifiable or not satisfied; it may be sensitive to initialization; it is typically computationally slow when the number of components K is large.

The estimated $\hat{\theta}^{*EM}$ is then used to cluster data. This may be used to retrieve an "hard" assignment of points to clusters or a "fuzzy" one by computing the following quantities:

$$\hat{\tau}_{i,k}^{(EM)} := \tau_k(x_i;\hat{\theta}^{*EM}),\ k=1,\ldots,K; \qquad \hat{z}_{i,k}^{(EM)} := \underset{k=1,\ldots,K}{\arg\max}\,\tau_k(x_i;\hat{\theta}^{*EM}). \tag{1.8}$$

For each $i = 1,\ldots,n$ and each $k = 1,\ldots,K$, the $\hat{\tau}$'s (estimated posterior probabilities) give a measure of the confidence with which point x_i is assigned to cluster k. This is a fuzzy clustering in the sense that points are not definitively assigned to one cluster or another. Rather we estimate for each point the probability that it belong to a particular cluster. An hard assignment, which assign each point x_i to a single cluster k, defines a partition of the sample data given by the $\hat{z}'s$. These are computed as the optimal Bayes allocation, so as to assign point x_i to the cluster, k, most likely to have generated it, according to the posterior probabilities $\hat{\tau}$'s. $\hat{z}$'s is also called the *MAP* estimator of class assignment.

Up to now, we implicitly assumed a fixed number of mixture components, i.e. clusters, given by K. However, the number of clusters is typically not known in advance and needs to be selected (along with other modelling aspects). This will be discussed in length in Chapter 2.

Another common approach in model-based clustering maximizes the *classification likelihood*, defined as:

$$L_c(\theta, z_1,\ldots,z_n) := \prod_{i=1}^{n}\phi_{z_i}(x_i),$$

[5] The presented one is a basic implementation of the algorithm. There are many different choices, including: initialization methods, different convergence criteria, and restriction on estimated parameters. Also, this can be adapted when the M-step does not have closed form solution (GEM); see McLachlan and Krishnan, 2007.

Algorithm 1: EM algorithm. (GM: Gaussian Mixture updates.)

Input: data $\mathbb{X}_n$; initialization of parameters $\theta^{(0)}$; threshold ϵ.
Output: $\hat{\theta}^*$.

$i \leftarrow 0$
$Converged \leftarrow 0$
while $Converged < 1$ **do**
  E-step: Compute $Q(\theta|\theta^{(i)}) = \mathbb{E}\left(cl(\theta)|\mathbb{X}_n, \theta^{(i)}\right)$
  M-step: Compute $\theta^{(i+1)} = \arg\max_\theta Q(\theta|\theta^{(i)})$
  if $\left\{l\left(\theta^{(i+1)}\right) - l\left(\theta^{(i)}\right) < \epsilon\right\}$ **then**
    $Converged \leftarrow 1$
  $i \leftarrow i + 1$
$\hat{\theta}^{*EM} \leftarrow \theta^{(i)}$

In case of Gaussian mixtures, i.e. $f_k = \phi_k$, $k = 1, \ldots, K$, the E-step and the M-step can be conveniently expressed in closed forms solution:

E-step (GM):

$$\tau_{i,k}(\theta^{(i)}) \leftarrow \frac{\pi_k(\theta^{(i)})\phi_k(x_i;\theta^{(i)})}{\sum_{k=1}^K \pi_k(\theta^{(i)})\phi_k(x_i;\theta^{(i)})}$$

M-step (GM):

$$\pi_k^{(i+1)} \leftarrow \sum_{i=1}^n \frac{\tau_{i,k}(\theta^{(i)})}{n}; \qquad \mu_k^{(i+1)} \leftarrow \frac{\sum_{i=1}^n x_i \tau_{i,k}(\theta^{(i)})}{\sum_{i=1}^n \tau_{i,k}(\theta^{(i)})};$$
$$\Sigma_k^{(i+1)} \leftarrow \frac{\sum_{i=1}^n (x_i - \mu_k^{(i+1)})(x_i - \mu_k^{(i+1)})^T \tau_{i,k}(\theta^{(i)})}{\sum_{i=1}^n \tau_{i,k}(\theta^{(i)})};$$
$$\theta^{(i+1)} = \left[\pi_1^{(i+1)}, \mu_1^{(i+1)}, \Sigma_1^{(i+1)}, \ldots, \pi_K^{(i+1)}, \mu_K^{(i+1)}, \Sigma_K^{(i+1)}\right].$$

---

where $z_i = k$ if point i belongs to component k (more generally, ϕ is replaced by density function f_i composing the mixture; these may be different from Gaussian densities of course). This is similar to the complete data likelihood, (1.4), except for the weight π's. Also, the direct maximization of the classification likelihood treats z's as parameters to be estimated as well, introducing a combinatorial aspect that makes this task typically hard to solve. For this reason, other approaches are preferred, like the CEM (classification EM algorithm), which is a modification of the standard EM, introducing the C-step in between the E and M steps. The C-step hard assigns points to cluster via the posterior probabilities from the E-step, the M-step maximizes parameters cluster-wise (see for example Celeux and Govaert, 1992).[6] This method Agglomerative hierarchical clustering is another option (i.e., successively agglomerating points in clusters in order to maximally increase L_c. See Fraley and Raftery, 2002).

Lastly, we briefly discuss restriction and parametrizations for covariance matrices in Gaussian mixture models.

[6] It must be noted that several authors used a CEM-type algorithm more or less explicitly; for earlier references than Celeux and Govaert, 1992, see Scott and Symons, 1971, Bryant and Williamson, 1978 and McLachlan, 1982.

The degeneracy of covariance matrices may cause a failure of the EM algorithm due to the unboundedness of the likelihood (e.g. see McLachlan and Peel, 2000). This happens when one of the component is centred on a sample point and its covariance tends to a singular matrix. This motivates the introduction of some restrictions to prevent the covariance matrices to become singular. For example, in univariate case Hathaway, 1985 proposed to restrict components' variances so that $min_{i,j}\{\sigma_i/\sigma_j\} \geqslant c > 0$, for all $i \neq j, i, j = 1, \ldots, K$, for some $c > 0$. This guarantees the existence of a global maximizer, θ^*, of the likelihood. The author also proposes a possible extension to the multivariate case. This is reformulated via eigenratio constraint by Ingrassia, 2004 (in this work and also in Ingrassia and Rocci, 2007 EM algorithms implementing these constraints are proposed). Eigenratio constraints were also proposed in the context of classification likelihood in Garcìa-Escudero et al., 2008. Under few regularity conditions, this constraint guarantees existence and consistency of the solution to the maximum likelihood problem (see Coretto and Hennig, 2017). The eigenratio constraint is as follows:

$$\frac{\lambda_{\max}}{\lambda_{\min}} \leqslant \gamma < +\infty, \tag{1.9}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the highest and smallest eigenvalues of all the components' covariance matrices.

That this constraint affect the geometric shape of the clusters. For example, setting $\gamma = 1$ implies that all the covariance matrices are spherical (as in k-means clustering). However, since the parameter γ can change continuously, it is not easy to state its effect on the shapes for other values. Intuitively, higher values of γ allow for more flexible shapes of the covariance matrices. However, the shape of the clusters may be more easily controlled via parametrizations of the covariance matrices.

In model-based clustering, the geometric shape of the mixture components determine also the clusters' shape: elliptical clusters, spherical clusters, elongated and so on. In Gaussian mixtures this is essentially controlled via the components' covariance matrices. Collecting other proposals in the literature (Fraley and Raftery, 1998), Banfield and Raftery, 1993 proposed to consider the eigenvalue decomposition of the covariance matrices given by

$$\Sigma_k = \lambda_k D_k A_k D_k^T, \tag{1.10}$$

where

- $\lambda_k = \left|\Sigma_k\right|^{1/p}$ is the highest eigenvalue, controlling for k-th cluster size, meaning the volume of the cluster in the p-dimensional space.
- D_k's are the eigenvector matrices, controlling for the orientation of the cluster in the p-dimensional space.
- A_k is a diagonal matrix of normalized eigenvalues sorted in decreasing order, $\{a_{k1}, a_{k2}, \ldots, a_{kp}\}$, $\left|A_k\right| = 1$. This controls for the shape of the cluster. For example, if $a_{k1} \gg a_{k2}$ then the cluster is basically concentrated on a line (note: the orientation of the line depends on D_k), while if all the eigenvalues are of the same magnitude we have a spherical cluster.

Table 1.2: Covariance eigenvalue parametrizations. p is data dimensionality. α counts the number of parameters other than those in covariance matrices, i.e. means μ's and mixing proportions π's; $\alpha = Kp + K - 1$; if mixing proportions are restricted to be equal, $\alpha = Kp$. β counts the number of parameters in a covariance matrix; $\beta = p(p+1)/2$.

Σ_k	# parameters to estimate	Implied Distribution	Clusters' Volume	Clusters' Shape	Clusters' Orientation
λI	$\alpha + 1$	Spherical	Equal	Equal	–
$\lambda_k I$	$\alpha + p$	Spherical	Variable	Equal	–
λA	$\alpha + p$	Diagonal	Equal	Equal	Coord. Axes
$\lambda_k A$	$\alpha + p + K - 1$	Diagonal	Variable	Equal	Coord. Axes
λA_k	$\alpha + Kp - K + 1$	Diagonal	Equal	Variable	Coord. Axes
$\lambda_k A_k$	$\alpha + Kp$	Diagonal	Variable	Variable	Coord. Axes
λDAD^T	$\alpha + \beta$	Ellipsoidal	Equal	Equal	Equal
$\lambda DA_k D^T$	$\alpha + \beta + K - 1$	Ellipsoidal	Equal	Variable	Equal
$\lambda_k DAD^T$	$\alpha + \beta + (K-1)(p-1)$	Ellipsoidal	Variable	Equal	Equal
$\lambda_k DA_k D^T$	$\alpha + \beta + (K-1)p$	Ellipsoidal	Variable	Variable	Equal
$\lambda D_k AD_k^T$	$\alpha + K\beta - (K-1)p$	Ellipsoidal	Equal	Equal	Variable
$\lambda_k D_k AD_k^T$	$\alpha + K\beta - (K-1)(p-1)$	Ellipsoidal	Variable	Equal	Variable
$\lambda D_k A_k D_k^T$	$\alpha + K\beta - (K-1)$	Ellipsoidal	Equal	Variable	Variable
$\lambda_k D_k A_k D_k^T$	$\alpha + K\beta$	Ellipsoidal	Variable	Variable	Variable

The parametrization (1.10) is a flexible one and allows to easily take into account several different clustering scenarios, controlling for the relative shapes of the clusters by constraining some of the components to be equal across k. One of the main motivation for these parametrization is to have more parsimonious models: some parametrizations, indeed, allow for a great reduction in the number of estimable parameters, which helps reducing the variance of maximum likelihood estimates, especially in smaller samples (e.g. see Table 1.2 ahead). Celeux and Govaert, 1995 give a detailed treatment of 14 different models of interest arising from this parametrization and they detail the M-step for both EM and CEM approaches (E and C steps stay as usual). The *Mclust* package in *R* language implements EM estimation with these parametrizations. Table 1.2 joins Table 3 of Scrucca et al., 2016 and Table 1 of Celeux and Govaert, 1995. This table shows all the 14 different cases together with the number of parameters to be estimated (note that when the subscript k is present, it means that the model allows the corresponding object to vary across clusters).

It must be noted that these parametrizations are also extendible to non-Gaussian mixture (Banfield and Raftery, 1993). For example, the popular *EMMIXskew* Package (Wang, Ng, and McLachlan, 2018) allows one to estimate not only (skewed) Gaussian mixtures, but also mixtures of (skewed) t distributions use similar alternative parametrizations.

1.3.2 Model fitting, approximation error and the bias-variance trade-off

Let $\mathcal{X}$ be the feature space, i.e. the space where the features are represented. In our framework, $\mathcal{X}$ is a space of the form $\mathbb{R}^{p_1} \times \mathbb{N}^{p_2} \times \{0,1\}^{p_3}$, for some integers p_1, p_2, p_3. In other words, the features can be quantitative, categorical or binary. We will denote a single element of the space $\mathcal{X}$ as x and address it as *features*. If $p_1 + p_2 + p_3 = p$, we say that the number of features is p, and $x := [x_1, x_2, \ldots, x_p]^T$. Let $\mathcal{Y}$ be the space of the outcome variable; in our application $\mathcal{Y} \in \mathbb{N}$, i.e. a categorical variable. Thus, we are interested in classification problems. In other settings, $\mathcal{Y}$ could be binary or quantitative as well. A further generalization is to have more than one dimension for $\mathcal{Y}$: these are known as multi-label classification problems (e.g. see Tsoumakas

and Katakis, 2007 for a brief introduction and known methodologies). An element of $\mathcal{Y}$ will be denoted by y and referred as *label*. We assume to observe n labelled objects from $\mathcal{X}$, namely we observe n pairs from $(\mathcal{X}, \mathcal{Y})$; we refer to those pairs as (x_i, y_i), $i = 1, \ldots, n$. Note that the pairs (x, y) may be seen as realizations of random variables (X, Y), taking values in $(\mathcal{X}, \mathcal{Y})$ and defined on an underlying probability space $(\Omega, \mathcal{F}, P)$. With a slight abuse of notation, we will refer to P also for the probability induced by the random variable X and Y, so that $P(X, Y)$ indicates their joint probability. In fact, we are interested in modelling some aspects of the conditional distribution $P(Y|X) = P(X, Y)/P(X)$. Typically, this is modelled through mappings from the feature space to the outcome space:

$$f(\cdot) : \mathcal{X} \to \mathcal{Y}.$$

this mapping is chosen in order to optimize some criterion. Building a parallel with Econometrics, suppose we are interested in predicting the values of a continuous variable Y (i.e. $\mathcal{Y} \subseteq \mathbb{R}$) using information on X. As a criterion to evaluate our predictions we may look at the classical squared error $\big(Y - f(X)\big)^2$. Now, minimizing the average value of this criterion leads to:

$$\begin{aligned}
&\mathbb{E}_{X,Y}\big(Y - f(X)\big)^2 = \mathbb{E}_{X,Y}\big(Y - f(X) + \mathbb{E}_{Y|X}\,Y - \mathbb{E}_{Y|X}\,Y\big)^2 = \\
&\quad \mathbb{E}_X\,\mathbb{E}_{Y|X}\Big(\big(Y - \mathbb{E}_{Y|X}\,Y\big)^2 + \big(f(X) - \mathbb{E}_{Y|X}\,Y\big)^2 - 2\big(Y - \mathbb{E}_{Y|X}\,Y\big)\big(f(X) - \mathbb{E}_{Y|X}\,Y\big)\Big) = \\
&\qquad\qquad \mathbb{E}_X\,\mathbb{E}_{Y|X}\Big(\big(Y - \mathbb{E}_{Y|X}\,Y\big)^2 + \big(f(X) - \mathbb{E}_{Y|X}\,Y\big)^2\Big),
\end{aligned}$$

which is minimized at:

$$f(X) = \mathbb{E}_{Y|X}\,Y =: f^*,$$

(see Hastie, Tibshirani, and Friedman, 2009, or Angrist and Pischke, 2008). Of course, f^* is not available and needs to be approximated somehow: the learning algorithms differ in the way they approximate this function. The function f^* was determined minimizing the particular criterion $\big(Y - f(X)\big)^2$. These criteria are often called *loss functions* and we will denote a generic loss function by L. A common choice of L for classification problems is the so-called 0-1 loss, which takes value 1 if the predicted class is the true one and 0 otherwise. A generalization of the square loss presented above is Cucker and Smale, 2002; for a comparison of loss functions for both classification (binary) and regression problem see Rosasco et al., 2004; for a survey on classification strategies with categorical output variables see Aly, 2005.

Generalizing the discussion above, the aim of a supervised learning problem is to minimize the risk R, where:

$$R(f) := \mathbb{E}_{X,Y}\,L(Y, f(X)),$$

for a given L. The minimum achievable risk is defined as:

$$R(f^*), \quad f^* = \underset{f \in \text{all functions}}{\arg\min}\ \mathbb{E}_{X,Y}\,L(Y, f(X)). \tag{1.11}$$

Typically, (1.11) is not feasible. With learning algorithms, we try to approximate f^*, by restricting the possible functions f to a subset, say H, the dimension of which depends on the algorithm's complexity. If $P(X, Y)$ was known, the minimum achievable risk, according to a

given algorithm, would then be:

$$R(\tilde{f}), \quad \tilde{f} = \arg\min_{f \in \mathrm{H}} \mathbb{E}_{X,Y}\, L(Y, f(X)). \tag{1.12}$$

However, (1.12) is not achievable either, because we do not usually know $P(X, Y)$. However, $P(\cdot, \cdot)$ can be learned from the data. Typically, the empirical measure replaces P so that (1.12) is solved in terms of the empirical version:

$$R(\hat{f}_n), \quad \hat{f}_n = \arg\min_{f \in \mathrm{H}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)). \tag{1.13}$$

Note that $R(f^*) \leqslant R(\tilde{f}) \leqslant R(\hat{f}_n)$. Then it is natural to ask: using $\hat{f}_n$ in place of f^*, how far from the optimum are we? In term of the above quantities, this can be expressed as:

$$R(\hat{f}_n) - R(f^*) = \big(R(\tilde{f}) - R(f^*)\big) - \big(R(\hat{f}_n) - R(\tilde{f})\big),$$

where we call *approximation error* the first term on the right-hand side and *estimation error* the second term. The approximation error depends on the complexity of the learner. The more complex is the learner, the richer is the set H, and the closer are $\tilde{f}$ and f^* (refer to Figure 1.6). The estimation error arises because we are estimating $\tilde{f}$ with sample information. It depends both on the complexity of the learner and on the sample information. Increasing the complexity of the learner, the approximation error usually drops, while it can be shown that the estimation error increases (this is basically due to the overfitting bias). This is also sometimes referred as the *bias-variance* trade-off. It is worth to notice that for some algorithms, there exist bounds for the estimation error.[7]

In order to evaluate a model, we would need to estimate $R(\hat{f}_n)$, which is given by:

$$\mathbb{E}_{X,Y}\, L(Y, \hat{f}_n(X)).^{8}$$

This is not available of course, and a naive estimate of this would be the following:

$$\frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}_n(x_i)). \tag{1.14}$$

Unfortunately, (1.14) is not a good estimate of $R(\hat{f}_n)$. Indeed, recalling its definition, $\hat{f}_n$, minimizes (1.14), leading to underestimate $R(\hat{f}_n)$. For this reason, a more sensible way to proceed is the following: split the data in two part, a training set, Tr, and a test set Te; compute $\hat{f}_n$ by

$$\hat{f}_n = \arg\min_{f \in H} \frac{1}{|\mathrm{Tr}|} \sum_{i \in \mathrm{Tr}} L(y_i, f(x_i));$$

[7] The terminology used here was adapted from Vapnik, 2013, Vapnik, 1992, Hastie, Tibshirani, and Friedman, 2009. For a short introduction see Vapnik, 1999. For the bound of estimation error one needs to evaluate the so-called Vapnick-Cervonenkis dimension of the learner, which measures its complexity: for further details see Vapnik, Levin, and Cun, 1994; for neural network see Koiran and Sontag, 1996.

[8] Since this quantity is computed for a given $\hat{f}_n$, we take the expectation conditioned on a particular sample/training set (this is what is referred as $\mathrm{Err}_{\mathcal{T}}$ in Hastie, Tibshirani, and Friedman, 2009.

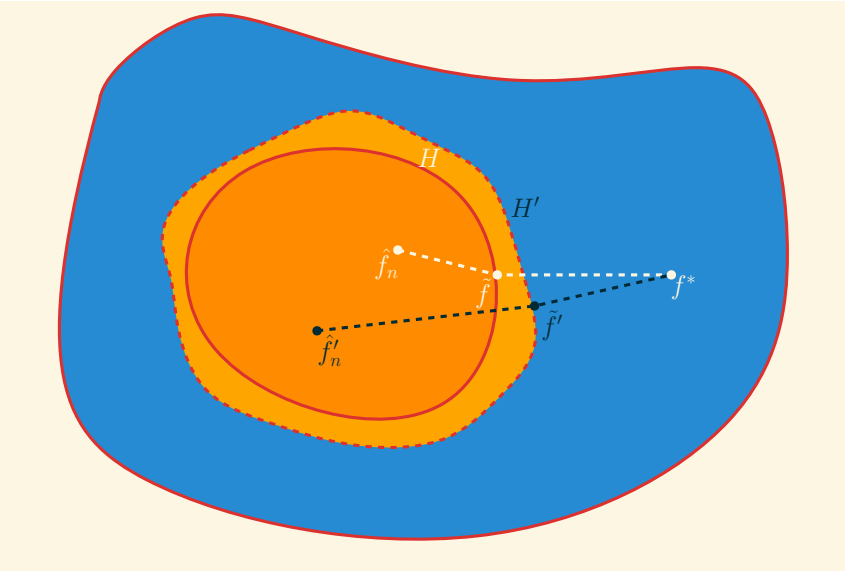


Figure 1.6: Representation of the approximation error and estimation error. Among the space of functions (blue set) the function f^* minimizes R; the function $\tilde{f}$ minimizes the risk R among the functions of the space H (dark orange set), implied by the chosen model; $\hat{f}_n$ minimizes the sample average of the loss function L on the observed sample. The dashed lines represent distances. As the classifier becomes more complex, the space H becomes wider (represented by H', light orange set) and the new solutions are represented by $\tilde{f}'$ and $\hat{f}'_n$: the approximation error (dashed line connecting $\tilde{f}$ and f^*) reduces, while the estimation error (dashed line connecting $\hat{f}_n$ and $\tilde{f}$) increases.

approximate $R(\hat{f}_n)$ with

$$\hat{R}(\hat{f}_n) = \frac{1}{|\text{Te}|} \sum_{i \in \text{Te}} L(y_i, \hat{f}_n(x_i)). \tag{1.15}$$

The idea is to approximate the risk, for a given $\hat{f}_n$ on unseen data, in order to mitigate the underestimation due to overfitting. Note also that (1.15) is an unbiased estimator for $R(\hat{f}_n)$.

A second issue is model selection. Typically, we have a plethora of classification/regression models and each of them has several *hyperparameters*. These are parameters that determines the set H, and thus the function $\hat{f}_n$ obtained by optimization, but are not directly optimized when minimizing $\frac{1}{|\text{Tr}|} \sum_{i \in \text{Tr}} L(y_i, f(x_i))$ (as an example, these are the number of nodes/layers in a neural network; allowed numbers of splits in a classification trees and minimum number of point in each terminal node; penalization parameter in a lasso regression etc.). However, we would like to optimize these "architectural" parameters as well. An approach is to try many different configuration and compute the implied risk for each of them. More formally, let a configuration of hyperparameters denoted as α, we need to rewrite $\hat{f}_n$ considering its dependency on α:

$$\hat{f}_n^{(\alpha)} = \arg\min_{f \in H_\alpha} \frac{1}{|\text{Tr}|} \sum_{i \in \text{Tr}} L(y_i, f(x_i)).$$

Then, ideally we would like to find α^* so that:

$$\hat{R}\big(\hat{f}^{(\alpha^*)}\big) = \arg\min_{\alpha} \hat{R}\big(\hat{f}_n^{(\alpha)}\big).$$

A couple of remarks here: we are evaluating the optimum obtained using the training set for a given α, $\hat{f}_n^{(\alpha)}$, on unseen data to avoid error underestimation; the minimization over all the possible configurations of α is typically extremely hard and computationally unfeasible. A solution to this is to try in a principled way only few of the possible configurations of α and to select the one achieving the minimum value for $\hat{R}\big(\hat{f}_n^{(\alpha)}\big)$. Selecting the best value for α is known as *tuning*. Once the tuning is completed and the best model has been selected, we might ask what the error of this model is. Note that if we were to use the same error used to select the model, we would incur in the same issue above in using the training error (1.14) as a proxy for $R(\hat{f}_n)$: the error used to select α is minimized with respect to the chosen α, so that it would likely underestimate the error on unseen data. To overcome this issue, we split the original dataset in three different samples as shown in Figure 1.3. For convenience let us (re)define the training error, validation error and test error as:

$$e_{\mathrm{Tr}} \coloneqq \frac{1}{|\mathrm{Tr}|} \sum_{i \in \mathrm{Tr}} L(y_i, \hat{f}_n^{(\alpha)}(x_i)); \tag{1.16}$$

$$e_{\mathrm{Va}} \coloneqq \frac{1}{|\mathrm{Va}|} \sum_{i \in \mathrm{Va}} L(y_i, \hat{f}_n^{(\alpha)}(x_i)); \tag{1.17}$$

$$e_{\mathrm{Te}} \coloneqq \frac{1}{|\mathrm{Te}|} \sum_{i \in \mathrm{Te}} L(y_i, \hat{f}_n^{(\alpha)}(x_i)). \tag{1.18}$$

Then, the procedure goes as follows: for a given α, use the training set to compute $\hat{f}_n^{(\alpha)}$ and e_{Tr}; with the so computed $\hat{f}_n^{(\alpha)}$, compute e_{Va} using the validation set; use e_{Tr} and e_{Va} to aid the choice of α; select α^* giving the lowest e_{Va}; using the test set, obtain an unbiased estimator of the risk by computing e_{Te} for $\hat{f}_n^{(\alpha^*)}$. Figure 1.7 provides a typical workflow.[9]

[9] This is just for illustrative purposes. We do not intend to provide an exhaustive listing of cases, or a definitive way to approach the problem. It is a rough indication of a typical workflow and some of the suggestions may not apply in all cases.

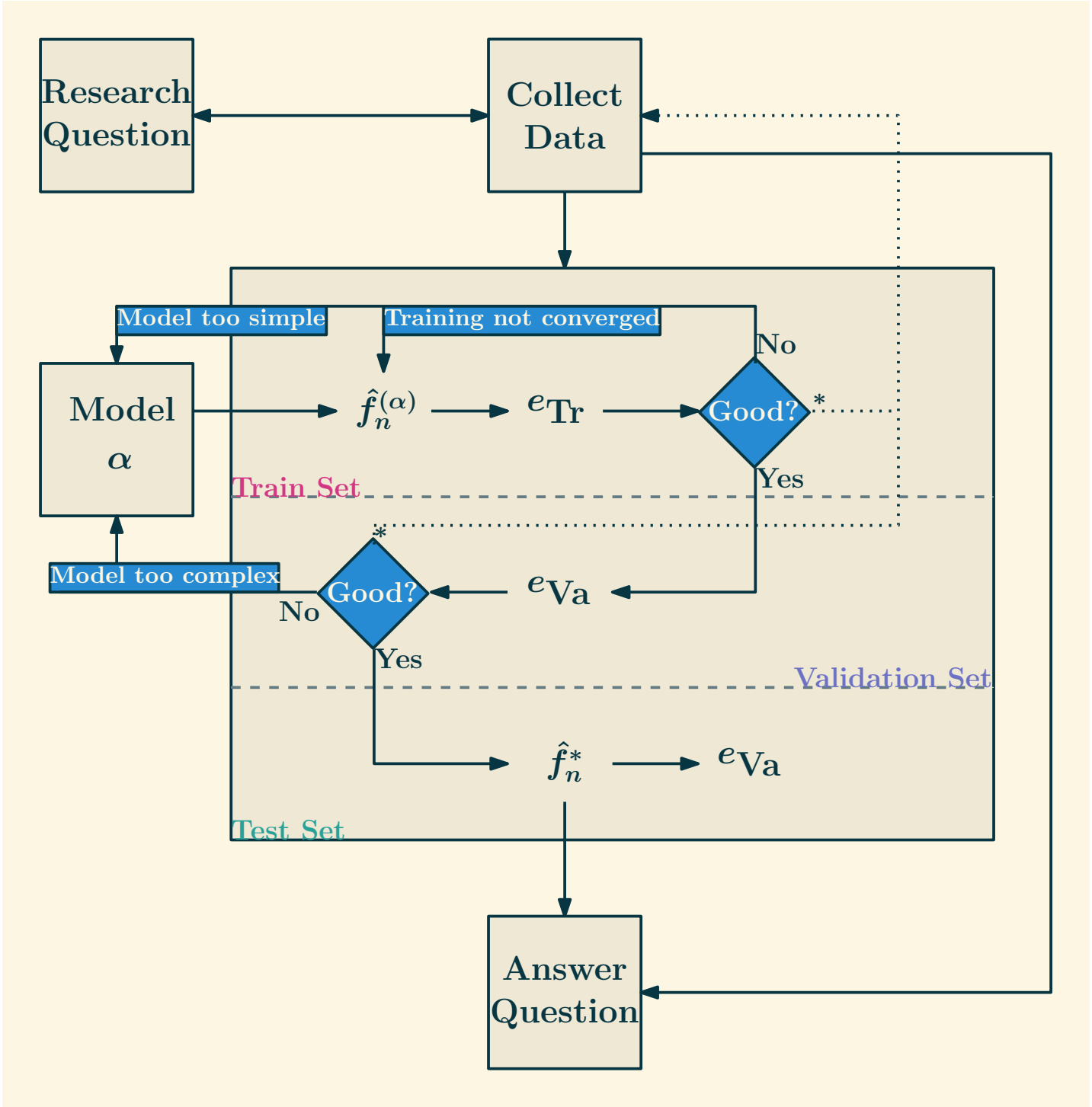


Figure 1.7: Typical workflow. Define a research question (can be inspired by data availability, motivating the double-sided arrow) and collect the needed data. Also, define some model (learning algorithms) in order to exploit data information. Having a model (with an initial architecture α), we can start an iterative procedure to optimize it. Split the data in training, validation and test set (separated by the dashed line). Train/fit the model on training data and compute the training error obtaining $\hat{f}_n^{(\alpha)}$ and e_{Tr}. The "goodness" of the errors is problem specific: in some fields values that would be otherwise be too low might be acceptable; one could use the related literature as a benchmark. Poor values here may depend on failure in training convergence (needs to re-estimate; sometimes requires tuning α to achieve convergence), or the learner may lack complexity to adapt to the data. Upon an acceptable e_{Tr}, use the validation set to estimate the generalization error, e_{Va}. Poor values here are usually due to overfitting (tune α to obtain a "simpler" model). In some cases, data might not contain enough information to obtain good results ($*$-line; collect more data). In trial and error fashion, tune α to achieve the best value for e_{Va}. Repeat to tune other learners (if more than one is considered), and choose the one achieving the best e_{Va}. Calling $\hat{f}_n^*$ the result of this process, we can (optionally) obtain an unbiased estimation of its generalization error using the test set to compute e_{Te}. Finally, we re-train $\hat{f}_n^*$ using the whole data and use it to answer the research question.

There are several ways to perform tuning and model selection. All of them rely on splitting the original data in multiple samples. There is no general principle to decide splitting proportions, however there are some splitting proportions that are more popular than others (see Figure 1.3). Other popular methodologies are *k-fold Cross Validation*, *leave-one-out*, *bootstrapping* and *stratified splitting* (see Hastie, Tibshirani, and Friedman, 2009 and Kohavi, 1995) and various corrections for them (Adler and Lausen, 2009). Before closing this brief introduction, let us stress two important points.

Remark 1.3.1. *Up to now, we implicitly assumed that the loss function L used to compute $\hat{f}_n$ is the same used in the computation of the errors (1.16), (1.17) and (1.18). However, there are some cases where the learning algorithm uses a different loss function, say L'. This is typically due to computational reasons, since differentiability and convexity play an important role in most optimization procedures. Thus, it may happen (especially in classification problems) that the algorithm minimizes a different loss function with respect to the one used to compute the errors that guide validations procedures and models assessment. As an example, the classification trees often minimize either the* Gini *or* Entropy *criteria (Breiman, 2017), while it is common to evaluate model performances using the number of misclassified points, i.e. $L(y_i, f(x_i)) = \mathbb{1}\{y_i \neq f(x_i)\}$. Sometimes, the optimization problem faced by the algorithm is a reformulation of the loss employed to compute the errors. Other times, the two losses coincides (e.g. the squared loss in regression problems). However, the aim is still errors minimization. So, even if the algorithm returns estimates $\hat{f}_n$ minimizing a slightly different loss, we will choose the one minimizing the errors (1.16) and (1.17). In most cases the minimizer of the two different loss function, L and L', coincides.*

Remark 1.3.2. *When splitting the data in train, validation and test set, it is important that the resulting splits are homogeneous. That is, even if with different sample sizes, the set should not contain substantially different information. This might be the case when non-random sampling is used (e.g. we use only individuals with particular characteristics to be in the training set). Or, even when using random sampling, this problem may arise in presence of very unbalanced datasets, where some classes are dramatically under-represented with respect to the others (here stratification may be a better resampling scheme).*

References

Adler, W. and Lausen, B. (2009). "Bootstrap estimated true and false positive rates and ROC curve". In: *Computational Statistics & Data Analysis* 53.3, pp. 718–729.

Aly, M. (2005). "Survey on multiclass classification methods". In: *Neural Netw* 19, pp. 1–9.

Angrist, J. D. and Pischke, J. (2008). *Mostly harmless econometrics: An empiricist's companion*. Princeton university press.

Arlot, S. and Celisse, A. (2010). "A survey of cross-validation procedures for model selection". In: *Statistics surveys* 4, pp. 40–79.

Athey, S. (Jan. 2018). "The Impact of Machine Learning on Economics". In: *The Economics of Artificial Intelligence: An Agenda*. University of Chicago Press, pp. 507–547. URL: http://www.nber.org/chapters/c14009.

Athey, S. and Imbens, G. (2016). "Recursive partitioning for heterogeneous causal effects". In: *Proceedings of the National Academy of Sciences* 113.27, pp. 7353–7360.

Bandiera, O. et al. (2017). *Ceo behavior and firm performance*. Tech. rep. National Bureau of Economic Research.

Banfield, J. D. and Raftery, A. E. (1993). "Model-based Gaussian and non-Gaussian clustering". In: *Biometrics*, pp. 803–821.

Belloni, A., Chernozhukov, V., and Hansen, C. (2014). "High-dimensional methods and inference on structural and treatment effects". In: *Journal of Economic Perspectives* 28.2, pp. 29–50.

Biernacki, C., Celeux, G., and Govaert, G. (2000). "Assessing a mixture model for clustering with the integrated completed likelihood". In: *IEEE transactions on pattern analysis and machine intelligence* 22.7, pp. 719–725.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Boyles, R. A. (1983). "On the Convergence of the EM Algorithm". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 45.1, pp. 47–50. ISSN: 00359246. URL: http://www.jstor.org/stable/2345622.

Breiman, L. (2017). *Classification and regression trees*. Routledge.

Bryant, P. and Williamson, J. A. (1978). "Asymptotic behaviour of classification maximum likelihood estimates". In: *Biometrika* 65.2, pp. 273–281.

Calvano, E. et al. (2018). "Artificial intelligence, algorithmic pricing and collusion". In:

Celeux, G. and Govaert, G. (1992). "A classification EM algorithm for clustering and two stochastic versions". In: *Computational statistics & Data analysis* 14.3, pp. 315–332.

— (1995). "Gaussian parsimonious clustering models". In: *Pattern recognition* 28.5, pp. 781–793.

Coretto, P. and Hennig, C. (2017). *Consistency, Breakdown Robustness, and Algorithms for Robust Improper Maximum Likelihood Clustering*. Tech. rep., pp. 1–39. URL: http://jmlr.org/papers/v18/16-382.html..

Cucker, F. and Smale, S. (2002). "Best choices for regularization parameters in learning theory: on the bias-variance problem". In: *Foundations of computational Mathematics* 2.4, pp. 413–428.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum likelihood from incomplete data via the EM algorithm". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1, pp. 1–22.

Fraley, C. and Raftery, A. E. (1998). "How many clusters? Which clustering method? Answers via model-based cluster analysis". In: *The computer journal* 41.8, pp. 578–588.

— (2002). "Model-based clustering, discriminant analysis, and density estimation". In: *Journal of the American statistical Association* 97.458, pp. 611–631.

Garcìa-Escudero, L. A. et al. (2008). "A general trimming approach to robust cluster analysis". In: *The Annals of Statistics* 36.3, pp. 1324–1345.

Gentzkow, M., Kelly, B., and Taddy, M. (2019). "Text as data". In: *Journal of Economic Literature* 57.3, pp. 535–74.

Goldfarb, A., Gans, J., and Agrawal, A. (2019). *The Economics of Artificial Intelligence: An Agenda*. University of Chicago Press.

Gordon, A. D. (1999). *Classification*. Chapman and Hall/CRC.

Hartigan, J. A. (1975). "Clustering algorithms". In:

Hastie, T., Tibshirani, R. J., and Friedman, J. (2009). *The Elements of Statistical Learning*. 2nd ed. Springer New York. DOI: 10.1007/978-0-387-84858-7. URL: https://doi.org/10.1007/978-0-387-84858-7.

Hathaway, R. J. (1985). "A constrained formulation of maximum-likelihood estimation for normal mixture distributions". In: *The Annals of Statistics* 13.2, pp. 795–800.

Hennig, C. et al. (2015). *Handbook of cluster analysis*. CRC Press.

Ingrassia, S. (2004). "A likelihood-based constrained algorithm for multivariate normal mixture models". In: *Statistical Methods and Applications* 13.2, pp. 151–166.

Ingrassia, S. and Rocci, R. (2007). "Constrained monotone EM algorithms for finite mixture of multivariate Gaussians". In: *Computational Statistics & Data Analysis* 51.11, pp. 5339–5351.

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). "Data clustering: a review". In: *ACM computing surveys (CSUR)* 31.3, pp. 264–323.

Jean, N. et al. (2016). "Combining satellite imagery and machine learning to predict poverty". In: *Science* 353.6301, pp. 790–794.

Kleinberg, J. et al. (2015). "Prediction policy problems". In: *American Economic Review* 105.5, pp. 491–95.

Kohavi, R. (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection". In: *Ijcai*. Vol. 14. 2. Montreal, Canada, pp. 1137–1145.

Koiran, P. and Sontag, E. D. (1996). "Neural networks with quadratic VC dimension". In: *Advances in neural information processing systems*, pp. 197–203.

McLachlan, G. J. (1982). "The classification and mixture maximum likelihood approaches to cluster analysis". In: *Handbook of statistics* 2, pp. 199–208.

McLachlan, G. J. and Krishnan, T. (2007). *The EM algorithm and extensions*. Vol. 382. John Wiley & Sons.

McLachlan, G. J. and Peel, D. (2000). *Finite Mixture Models*. John Wiley & Sons, Inc. DOI: 10.1002/0471721182. URL: https://doi.org/10.1002/0471721182.

McLachlan, G. J. and Rathnayake, S. I. (2015). "Mixture models for standard p-dimensional Euclidean data". In: *Handbook of Cluster Analysis*. Chapman and Hall/CRC, pp. 166–193.

Micci-Barreca, D. (2001). "A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems". In: *ACM SIGKDD Explorations Newsletter* 3.1, pp. 27–32.

Mullainathan, S. and Spiess, J. (2017). "Machine learning: an applied econometric approach". In: *Journal of Economic Perspectives* 31.2, pp. 87–106.

Parkhi, O. M., Vedaldi, A., and Zisserman, A. (2015). "Deep face recognition." In: *bmvc*. Vol. 1. 3, p. 6.

Redner, R. A. and Walker, H. F. (1984). "Mixture Densities, Maximum Likelihood and the Em Algorithm". In: *SIAM Review* 26.2, pp. 195–239. ISSN: 00361445. URL: http://www.jstor.org/stable/2030064.

Rosasco, L. et al. (2004). "Are loss functions all the same?" In: *Neural Computation* 16.5, pp. 1063–1076.

Schwab, K. (Dec. 12, 2015). "The Fourth Industrial Revolution". In: *Foreign Affairs*. URL: https://www.foreignaffairs.com/articles/2015-12-12/fourth-industrial-revolution.

Scott, A. J. and Symons, M. J. (1971). "Clustering methods based on likelihood ratio criteria". In: *Biometrics*, pp. 387–397.

Scrucca, L. et al. (2016). "mclust 5: clustering, classification and density estimation using Gaussian finite mixture models". In: *The R Journal* 8.1, pp. 205–233. URL: https://journal.r-project.org/archive/2016-1/scrucca-fop-murphy-etal.pdf.

Silver, D. et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587, p. 484.

Silver, D. et al. (2017). "Mastering the game of go without human knowledge". In: *Nature* 550.7676, p. 354.

Teoh, E. R. and Kidd, D. G. (2017). "Rage against the machine? Google's self-driving cars versus human drivers". In: *Journal of safety research* 63, pp. 57–60.

Tsoumakas, G. and Katakis, I. (2007). "Multi-label classification: An overview". In: *International Journal of Data Warehousing and Mining (IJDWM)* 3.3, pp. 1–13.

Vapnik, V. (1992). "Principles of risk minimization for learning theory". In: *Advances in neural information processing systems*, pp. 831–838.

— (1999). "An overview of statistical learning theory". In: *IEEE transactions on neural networks* 10.5, pp. 988–999.

— (2013). *The nature of statistical learning theory*. Springer science & business media.

Vapnik, V., Levin, E., and Cun, Y. L. (1994). "Measuring the VC-dimension of a learning machine". In: *Neural computation* 6.5, pp. 851–876.

Wang, K., Ng, A., and McLachlan, G. J. (2018). *EMMIXskew: The EM Algorithm and Skew Mixture Distribution*. R package version 1.0.3. URL: https://CRAN.R-project.org/package=EMMIXskew.

Chapter 2

Scoring and selection of clustering solutions in unsupervised learning

2.1 Introduction

In this chapter we address the problem of the selection of a clustering solution. We are mainly interested in the case where the clusters are reasonably identifiable with elliptic-symmetric density regions. Thus, we address situations in which a cluster is described by its centre, scatter and size.

In this context, there are a variety of choices to take into account when selecting a clustering model. One of the critical modelling aspects, to which it is usually given the most attention by the literature, is that of choosing the "adequate" number of clusters, K. Consider Figure 2.1, at the end of this introduction. The figure shows 5 different clustering solutions for the data represented in the top-left panel. Visualizing the solutions (which may be not feasible in higher dimensional settings), we see that some of them are rather more plausible than others, and a solution can be (relatively) easily selected. Of course, it is clear the need of a systematic approach to perform these choices.

There is a plethora of valid methodologies in the literature to tackle this problem. This particular framework, where clusters are represented by a size, centre and scatter parameters, is easily framed and understood under a model-based clustering approach, introduced in Chapter 1. In this case, there are two main approaches to model selection: information based criteria and criteria testing the number of components via likelihood-ratio type tests (see McLachlan and Peel, 2000 and Fraley and Raftery, 2002). More general criteria, not necessarily implying distributional assumptions as in model-based approaches, pertain to the evaluation of intra-cluster homogeneity and inter-clusters dissimilarity (e.g. Caliński and Harabasz, 1974; Rousseeuw, 1987).

In this chapter, we propose a novel methodology to select clustering solutions. This is a general approach in the sense that it can be used to evaluate any clustering solution where it makes sense to describe clusters in terms of size, centre and scatter quantities. This implies that clustering regions are elliptic-symmetrically shaped, but we do not require any specific assumption regarding distributional aspects for the data generating process.

We propose to evaluate a clustering solution via a scoring function that relates to the quadratic scoring (Hastie, Tibshirani, and Friedman, 2009). This function gives a measure

of the degree of adequacy with which sample points fit into given clusters. The selection of a clustering scheme is based on the evaluation of this scoring function at various solutions produced by any considered method. These solutions are formed estimating a clustering multiple times on bootstrap resamples. This allow us to obtain consistent estimates of the average value of the score at different parameters estimates, and also to take into account its variability.

The proposed procedure has several advantages.

- Even if strongly linked to model-based approaches, it does not actually require distributional assumptions;
- It allows to select many aspects of methods rather than just the number of clusters, K.
- It accounts for the variability that is induced by the implementation of a given clustering strategy (a method). Here we do not restrict to the comparison of clustering methods that are founded on model assumptions (model-based clustering). Although many of the concepts, and the intuitions used in this work are inspired by the model-based clustering literature. This is the reason why model-based clustering was introduced in Chapter 1.

A note on the last bullet. Typically, classical model selection criteria are focussed on selecting the number of components, K. However, there are usually many other choices to be considered. For example, in model-based clustering we reviewed constraints on covariance matrices and different parametrizations (Subsection 1.3.1); also, different initializations methods for the clustering algorithms are possible (McLachlan and Krishnan, 2007). As we will show later in the empirical analysis, our procedure is able to automatically take into account these different choices.

The consistency of this methodology is proven theoretically. We conduct an extensive empirical analysis comparing different clustering selection methods. In the empirical comparison, the proposed method is able to retrieve the true underlying clustering structure of the data

The chapter is so organized. The remaining part of this section (Subsection 2.1.1) reviews some of the methods proposed in the literature to select clustering solutions; Section 2.2 introduces the scoring function and the related methodology we propose; Section 2.3 introduces the proposed resamlping scheme and develops the theoretical analysis; Section 2.4 illustrates the empirical analysis; Section 2.5 concludes the chapter with some final remarks.

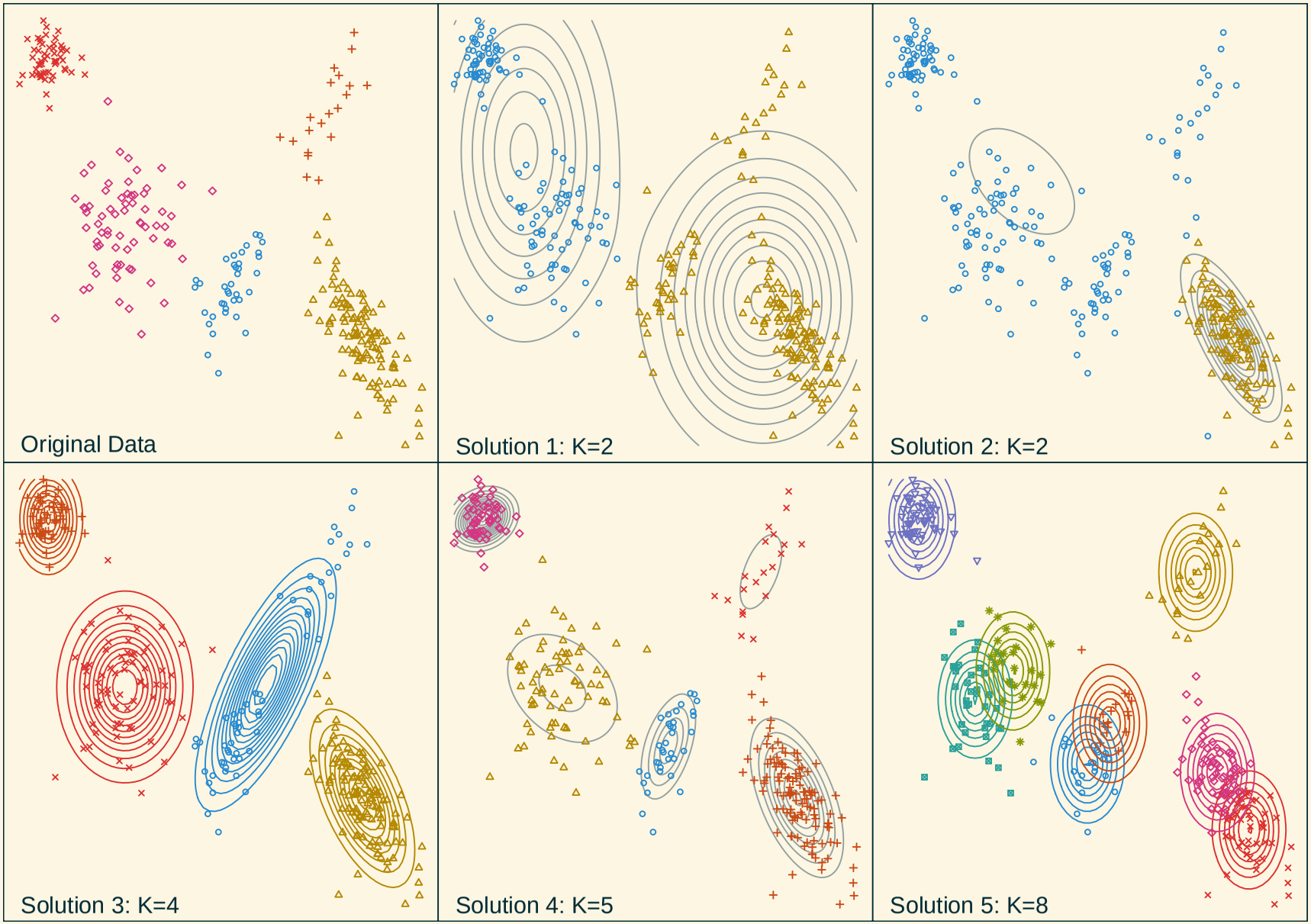


Figure 2.1: Different clustering solutions example. (Top-Left) Original sample made of 300 points from a 5-component t-student mixture model; different colors and shapes of the points highlight the generating component. (Other Graphs) 5 clustering solutions obtained either via Mclust or OTRIMLE R packages. Covariances parametrization and number of clusters K dramatically shape the solution.

2.1.1 Methods from the literature

There are different aspects and approaches in cluster validation. These are summarized in Hennig and Meila, 2015 as:

- techniques to test whether data are clustered in the first place or not;
- indexes defined by means of some objective function to define a measure of cluster quality (these methods are generally independent from the particular clustering approach);
- indexes to compare different clustering solutions (this may be particularly helpful when there is external information to be considered or to evaluate stability of the solutions);
- evaluating the stability of clustering solutions (i.e. different but similar data should be clustered analogously);
- data visualization techniques to aid the clustering process.

Here we will be mainly concerned with indexes to evaluate and select a clustering solution. In what follows, we review some of the existing methodologies for model selection. As briefly mentioned in the introduction, there is a huge number of different criteria. A comprehensive review of these approaches is not in the scope of the thesis. Also, as far as the author knows, there is no exhaustive survey on these. We point at Hennig et al., 2015 and Halkidi, Vazirgiannis, and Hennig, 2015 for a wide survey, although not complete.

Before doing that, we want to highlight a difference with model selection methods in supervised learning: cluster selection is concerned with observed data and not with unobserved "future" data, as it is the case in supervised learning. Thus, most of the methods do not require the construction of a validation set as we did in supervised learning; rather, the quality of the solution is evaluated in-sample. There are exception as we will see later. The conceptual difference is that supervised learning is concerned about predictions, which are meaningful on unseen data. Unsupervised learning is mainly concerned in optimally representing data at hand, by describing some of its unobserved structure or characteristic.

We review the indexes that are going to be used in later analysis. We consider two broad categories of indexes: model-free indexes and model-based indexes. The former, does not require any particular distributional assumption on the data. These indexes takes as input the partition of sample points given by a clustering method.

The second category, instead, explicitly implies a model-based approach, where there are precise distributional assumptions (see Subsection 1.3.1). These methods typically relies on penalized likelihood criteria (hence the model-based framework).

We recall that we restrict ourselves to a case where clustering solutions be fully represented by a sizes, centres and scatters. These clustering solutions are typically generated by location-scale families distributions and can be easily framed into model-based clustering by an appropriate choice of the underlying mixture distribution.

Thus, in what follows, we will consider clustering solutions as a member m of a set of candidate solutions, $\mathcal{M}$, where each member of $\mathcal{M}$ is a parametric description of the clustering structure implied by a certain methods. In particular, since we are interested in situations where the groups have symmetric and elliptical shapes, each element m is determined by:

- $K(m)$, the number of clusters;
- $\theta(m) = \left[\pi_1^{(m)}, \mu_1^{(m)}, \Sigma_1^{(m)}, \ldots, \pi_{K(m)}^{(m)}, \mu_{K(m)}^{(m)}, \Sigma_{K(m)}^{(m)}\right]^T$, a vector of $K(m)$ cluster proportions, centres and scatters for the $K(m)$ clusters;
- $z_{i,k}^{(m)}$, the assignment of point x_i to cluster k.

In what follows, if not otherwise specified, $\mathbb{X}_n$ indicates an observed sample of size n and x_i denotes an element of $\mathbb{X}_n$.

Model-free criteria

We consider two different model-free indexes. The first one was proposed by Caliński and Harabasz, 1974, and it is defined as follows.

$$CHC(m) := \frac{\text{trace}\big((W(m)\big)}{\text{trace}\big(B(m)\big)} \frac{n - K(m)}{K(m) - 1}, \tag{2.1}$$

where

$$n_k = \left| \left\{ x_i \in \mathbb{X}_n : z_{i,k}^{(m)} = 1 \right\} \right|; \quad \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i; \quad \bar{x}_k = \frac{1}{n_k} \sum_{x_i : z_{i,k}^{(m)} = 1} x_i;$$

$$W(m) := \sum_{k=1}^{K(m)} \sum_{x_i : z_{i,k}^{(m)} = 1} (x_i - \bar{x}_k)(x_i - \bar{x}_k)^T;$$

$$B(m) := \sum_{k=1}^{K(m)} n_k (\bar{x}_k - \bar{x})(\bar{x}_k - \bar{x})^T$$

(note, for convenience,we suppressed the dependence of n_k and $\bar{x}_k$ on m). The first ratio in (2.1) is the ratio between the intra cluster variability (within variability), W, and the inter cluster variability (between variability), B. The higher this ratio the better the clustering, because we expect homogeneous clusters (low W) that differ a lot one from each other (high B). The second ratio is a correction term: the variance of B, considering K terms, has $K-1$ degrees of freedom; the variance of W has $n - K$ degrees of freedom.

Note that an increasing number of clusters K, should reduce both W (smaller clusters) and B (closer cluster means due to their increased number). Thus, CHC trades-off these two effects. The CHC criterion does not assume a model for the data generating distribution. The underlying cluster notion is that clusters are well separated group of points having low intra-cluster variance. Of course this homogeneity notion has a model interpretation when we assume that the generating distribution produces spherical clusters as noted in Halkidi, Vazirgiannis, and Hennig, 2015.

The second index we consider is the Average Silhouettes Width (ASW) criterion introduced by Rousseeuw, 1987. As for the CHC criterion, this method measures the intra cluster homogeneity and inter cluster variability, Let $d(x_i, x_j)$ be a proper measure of dissimilarity between

the observed points x_i and x_j, the ASW criterion is defined as follows:

$$ASW(m) := \frac{1}{n}\sum_{i=1}^{n}\frac{b_i - a_i}{\max\{a_i, b_i\}}, \tag{2.2}$$

where

$$\begin{aligned}
n_k &= \left|\left\{x_i \in \mathbb{X}_n : z_{i,k}^{(m)} = 1\right\}\right|; \\
a_i &= \frac{1}{n_k - 1}\sum_{x_j : z_{j,k}=1} d(x_i, x_j) \quad \text{where } k: \ z_{i,k}^{(m)} = 1; \\
b_i &= \min_{l \neq k}\frac{1}{n_l}\sum_{x_j : z_{j,l}=1} d(x_i, x_l) \quad \text{where } k: \ z_{i,k}^{(m)} = 1.
\end{aligned}$$

Therefore, the ASW criterion requires the specification of a dissimilarity measure to assess the underlying homogeneity intra-cluster notion. Thus, a_i is the average dissimilarity of point x_i from points within its assigned cluster; b_i is the dissimilarity of point x_i from points in the cluster to which x_i is most similar to.

Each summand in (2.2) is the *silhouette* of point x_i; it is lower for points that lie at the "borders" of a cluster, i.e. those points that do not clearly belong to one or another cluster (b_i small, a_i big). The idea is that the more the clusters are separated and internally homogeneous (b_i big, a_i small for all points) the higher is the value of ASW.

The ASW statistics in (2.2) measures the average silhouettes across the sample points. As for the CHC criterion this popular statistics does not require any model assumption, although the choice of $d(\cdot, \cdot)$ implicitly formalizes the specific notion of homogeneity that the analyst is pursuing. The Silhouettes Width plot, a graphical display based on the silhouette values, is also a popular display for assessing a data partition (see Rousseeuw and Kaufman, 1990).

Model-based criteria

As mentioned above, these methods hypothesize a particular form for the data distribution. In our case, this is a mixture distribution with components being distributions of a location-scale family.

In principle, the log-likelihood function (1.4) could be used as a measure of cluster quality: the higher the likelihood the better the ability of the clustering solution to represent the data. Thus, the solution maximizing the log-likelihood would be the chosen one. However, this is likely to be a poor solution, since this method tends to overestimate K.

The likelihood of the data will monotonically increase with model the complexity, which is expressed both in terms of number of free parameters (e.g. refer to Table 1.2) and number of components. To see this, imagine a simple case where $\pi_k = \frac{1}{K}$ and $\Sigma_k = I$ for all $k = 1, \ldots, K$. Consider these parameters to be fixed, while only K and the centres are allowed to be chosen in order to optimize the observed likelihood. In this case, only distances of observed points to components' centres, μ_k's, matter. In particular, the closer a point to any centre, the better the likelihood evaluated at the point. Thus, it is always possible to achieve an higher likelihood under $K + 1$ components rather than K. Eventually, this reasoning would select $K = n$ components,

where each component is centred at a sample point.

Define the observed log-likelihood for model m to be:

$$l(m) := \sum_{i=1}^{n} \log \Big(\sum_{k=1}^{K(m)} \pi_k^{(m)} f_k\big(x_i; \mu_k^{(m)}, \Sigma_k^{(m)}\big) \Big), \tag{2.3}$$

where f_k is a density function completely specified by parameter μ_k and Σ_k. According to this criterion, one may select:

$$m^* = \arg\max_{m \in \mathcal{M}} \; l(m)$$

This naive implementation would select solutions with too many clusters. This motivates the introduction of some penalizations for model complexity. Note that in the model-based context where these criteria are used, a member m identifies a model.

The Bayesian Information Criterion (BIC) was introduced by Schwarz, 1978. It is obtained as an asymptotically valid approximation of the posterior probability of model m being the "true" model, given observed data. This posterior probability is also called the *integrated likelihood* of model m (see Fraley and Raftery, 2002). This criterion tends to select the model, among the considered set of models $\mathcal{M}$, with the highest posterior probability of having generated observed data.

The BIC criterion for model m is defined as:

$$BIC(m) := 2l(m) - \nu(m) \log n, \tag{2.4}$$

where $l(m)$ is as in (2.3) and $v(m)$ is the number of free parameters in model m, that needs to be estimated. The BIC in (2.4) increases with the observed likelihood, i.e. where we have a better fit of the underlying density. However, this quantity is penalized by the model complexity, captured by the term $\nu(m)$. This penalization grows with the sample size n at a logarithmic rate. The intuition is that more complex models may achieve an high value of the likelihood because of their excessive fit the data (overfitting). This problem exacerbates when the sample size increases.

Keribin, 1998 showed that, under suitable conditions, this method consistently estimates the correct number of components, K, for mixture densities (an example are Gaussian mixtures with compact parameters space and covariance matrices of the type λI; refer also to Table 1.2). Hence, BIC is widely used in this context (Fraley and Raftery, 1998; also Fraley and Raftery, 2002 gives examples of successful applications).

It must be noted that the BIC tries to select a model, within the considered set of models, that approximate well the underlying data distribution. This might not always be desirable. For example, suppose that the underlying data in generated with very K' asymmetric groups, and that the set of models, $\mathcal{M}$, is made of Gaussian mixture models only, with different number of components. Then, since a mixture of Gaussian with a sufficiently high number of components can approximate any continuous distribution, BIC may easily prefer a model such that $K \gg K'$, in order to fit the underlying data generating process well. Thus, the number of groups for which the BIC would (supposedly) be consistent need not to relate to the number of clusters.

The Integrated Complete Likelihood criterion (ICL) was introduced in Biernacki, Celeux, and Govaert, 2000, and tries to overcome this latter problem of the BIC. This is derived under the same Bayesian conceptual framework underlying the BIC. However, differently from the BIC, this takes into account the ability of a mixture model to assess clustering structure of the data. This is done by constructing an approximation not for the integrated likelihood (as in BIC), but for the integrated complete likelihood. That is, in the derivations of models' posterior probability, the complete likelihood (1.3) is considered instead of likelihood (1.2). Now, this quantity is approximated via a BIC-like approximation, leading to the following criterion:

$$ICL(m) := 2cl(m) - \nu(m) \log n. \tag{2.5}$$

where, to actually compute it, indicator variables z's are estimated via the MAP estimator $\hat{z}$ as in (1.8).

Baudry et al., 2015 provides more insights and theoretical properties for the ICL. The author also suggests that this criterion tends to prefer more separated clusters than the BIC. Indeed, using the decomposition of (1.4) and (1.5) in Subsection 1.3.1, we can rewrite the ICL as:

$$ICL(m) := 2cl(m) - \nu(m) \log n = 2l(m) - \nu(m) \log n + 2h(\theta(m)) = \\ BIC(m) - (-h(\theta(m))).$$

Thus, the ICL is substantially equivalent to the BIC penalized by the estimated mean entropy term $-h(\theta(m))$ (also here the z's should be replaced by MAP $\hat{z}$). The entropy term is non negative and is higher for unclear assignment of point to clusters, which usually happens when there is a high number of overlapping components. This will help to mitigate the overestimation problem discussed for the BIC. However, in case of well separated components, BIC and ICL will tend to choose the same solution.

Another widely adopted criterion is the Akaike Information Criterion (AIC) introduced by Akaike, 1973. The criterion is defined as follows

$$AIC(m) := 2l(m) - 2\nu(m), \tag{2.6}$$

AIC is derived under an information theoretic approach as an asymptotically valid approximation of the expansion of the Kullback-Leibler distance of the candidate model with respect to the (unknown) true data generating process. This criterion selects the model, in the set of considered models, minimizing the expected value (where expectation is taken with respect to all possible generated samples from the true distribution) of the Kullback-Leibler distance with respect to the underlying true data generating process. For a full account on the derivations of AIC and historical notes, see Burnham and Anderson, 2003. Leroux, 1992 shows that, under suitable conditions, AIC does not underestimate the number of components in a mixture model asymptotically, while Celeux and Soromenho, 1996 show its tendency to overestimate number of components in finite samples.

The AIC has essentially the same interpretation of the BIC, in the sense that it is computed as the maximized value of the observed likelihood, penalized for model complexity. However, here the difference is that the penalization does not increase with the sample size n. Thus, especially in large samples, the AIC tends to penalize complex models less than BIC, tending to overestimate the number of components.

This tendency is partially taken into account by the AIC3, proposed by Bozdogan, 1983:

$$AIC3(m) := 2l(m) - 3\nu(m). \tag{2.7}$$

We now review one last criterion, not coming in the form of penalized likelihood. This approach was introduced by Smyth, 2000, who investigated a cross-validation approach to automatically determine the number of clusters in model-based clustering. We briefly mentioned cross-validation in Chapter 1. This is indeed typical in supervised learning and less common in unsupervised learning. The idea is to split the data in two independent part: one is used to fit the mixture model (implied by the model m); the other is used to evaluate the fitted model. The reason of scarce usage of this method in cluster analysis is that we do not have a ground truth available and usually it is not straightforward to evaluate performances (via the misclassification loss in this case) on unseen data.

Smyth, 2000 proposed the negative expected log-likelihood as the base loss function for the cross-validation. The rational is as follow. It is easy to see that the expected log-likelihood is proportional to the Kullback-Liblier loss attained when the formulated model is used to approximate the true data distribution. One may estimate it based on the observed sample using the sample estimate of the unknown model's parameter. However, Akaike, 1973 showed that this would produce such a biased estimate of the loss due to the overfitting trap. The proposal of Smyth, 2000 is to estimate the expected log-likelihood out-of-sample using cross-validation. The idea is that models that performs too well on the data at hand, might be too specific and adapted so that they generalize poorly on unseen data.

A general definition for this criterion is

$$CV(m) := \frac{1}{V}\sum_{i=1}^{V} l\Big(m(\mathbb{X}_{tr}^{(i)}); \mathbb{X}_{te}^{(i)}\Big),$$

where $l(m; \mathbb{X})$ is the log-likelihood function for model m evaluated on data $\mathbb{X}$; $m(\mathbb{X}_{tr}^{(i)})$ indicates that model m's parameters were obtained by estimation on $\mathbb{X}_{tr}^{(i)}$; V is an integer number depending on the particular form of cross-validation used; $\mathbb{X}_{tr}^{(i)}$ and $\mathbb{X}_{te}^{(i)}$ are subsets or $\mathbb{X}_n$ and their construction also depends on the form of cross-validation.

In *10-fold cross-validation*, for example, we randomly shuffle the data, $\mathbb{X}_n$, and then partition it in 10 subsets of equal size:

$$\mathbb{X}_n = \mathbb{X}_\nu^{(1)} \cup \mathbb{X}_\nu^{(2)} \cup \cdots \cup \mathbb{X}_\nu^{(10)}; \quad \forall i \neq j,\ \mathbb{X}_\nu^{(i)} \cap \mathbb{X}_\nu^{(j)} = \varnothing; \quad \nu \approx \frac{n}{10}.$$

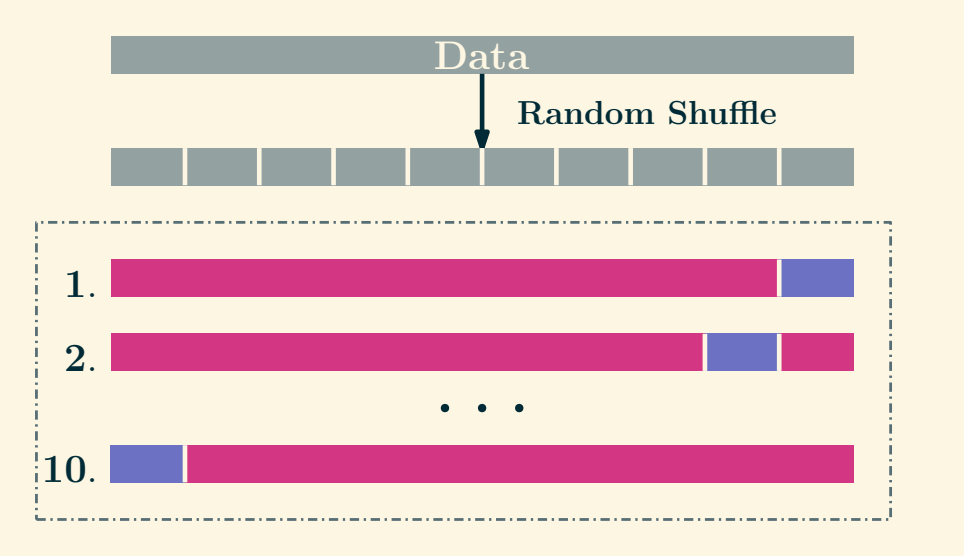


Figure 2.2: Ten-fold Cross-Validation. The data is first randomly shuffled and then partitioned in 10 equal size portions or folds. Each of the fold will be used once as a validation set (purple), while using the remaining 9 folds as a training set (magenta) for model estimation.

Then, $V = 10$ and for each $i = 1, \ldots, V$ we set:

$$\mathbb{X}_{te}^{(i)} := \mathbb{X}_{\nu}^{(i)}; \quad \mathbb{X}_{tr}^{(i)} := \mathbb{X}_n \backslash \mathbb{X}_{te}^{(i)}.$$

That is, the data is split in 10 folds; each fold is used once as test set while the remaining 9 are used as training set. Figure 2.2 gives a graphical representation of the described procedure.

Another type of cross-validation is the so called *Monte Carlo cross-validation*. In this case, we randomly shuffle the data $\mathbb{X}_n$ and partition it in two subsets of sizes γn and $(1-\gamma)n$, where $\gamma \in (0, 1)$ is set by the user. The two subsets constitute the training (tr) and test (te) sets. This procedure is repeated V times independently, obtaining, for $i = 1, \ldots, V$:

$$\mathbb{X}_n = \mathbb{X}_{tr}^{(i)} \cup \mathbb{X}_{te}^{(i)}; \quad \mathbb{X}_{tr}^{(i)} \cap \mathbb{X}_{te}^{(i)} = \varnothing; \quad |\mathbb{X}_{tr}^{(i)}| = \gamma n, \; |\mathbb{X}_{te}^{(i)}| = (1-\gamma)n,$$

and the obtained partitions are independent one another across i. Smyth, 2000 adopts this latter type of cross-validation, using $\gamma = 0.5$ and $V = 100$ (however the author suggests that V in the range 20 to 50 should suffice for many applications). For more details on cross-validation techniques see Arlot and Celisse, 2010.

2.2 Scoring cluster configurations

As stated in Section 2.1 one of the contributions of this Chapter is the introduction of a scoring mechanism based on the quadratic discriminant rule, a popular tool in classification analysis. We assume that there is population distribution F that produces clustered regions of points where each cluster is meaningfully described by a set of size, centrality and scatter parameters. A cluster configuration m is described by

K the number of clusters;

π_k the relative size of the cluster. This is well understood in model-based clustering as mixing proportion. Otherwise this is given by $\frac{n_k}{n}$, where n_k is the number of points in the k-th group, and n is total number of points. As usual $\sum_k \pi_k = 1$;

μ_k the centre of the cluster k.

Σ_k the scatter matrix of cluster k. Usually this is understood as the covariance matrix of the k-th group.

When needed, we make the dependence on a model m explicit.

These parameters have a clear interpretation in model-based clustering. However, they can also be retrieved in other clustering methods. In fact there are methods that, although they do not assume a location-scale model for the underlying clusters' distributions, they produce solutions that can be interpreted in this sense. For example the popular K-means algorithm produces a partition that coincides with the MAP assignment one would obtain under the parameter that maximizes the likelihood function of a mixture-model with K spherical Gaussian components. Therefore, in many cases clusters' size, centres and scatters can be treated as a meaningful description of the clustered regions, and this also applies outside the model-based domain where one assumes a certain location-scale model for the groups' distributions. We now describe and motivate the construction of a scoring function for a clustering solution m that can be described in terms of the overall parametric description

$$\theta^{(m)} := \left[\pi_1^{(m)}, \mu_1^{(m)}, \Sigma_1^{(m)}, \ldots, \pi_{K^{(m)}}^{(m)}, \mu_{K^{(m)}}^{(m)}, \Sigma_{K^{(m)}}^{(m)}\right]^T = \left[\theta_1^{(m)}, \ldots, \theta_{K^{(m)}}^{(m)}\right]^T.$$

We remark that these serves a general descriptions of a clustering solution: while, for convenience, we will refer to these as *parameters*, in principle they do not assume an underlying parametric model.

The proposed scoring mechanism is based on the well-known quadratic discriminant rule (e.g. see Bishop, 2006 or Hastie, Tibshirani, and Friedman, 2009). This rule is usually derived in supervised frameworks, where sample points are assigned to classes via optimal Bayes allocation. This is based on posterior probabilities, computed as the probability that a point came from a particular class, given the observed sample and hypothesized conditional class densities.[1] We will now illustrate how we use this score to evaluate clustering solutions and cluster assignments.

In the case of clustering, given a *fixed solution* m and looking at its clusters as classes, the quadratic score for point x, for a given cluster k is given by (for simplicity we drop the dependence of the parameters on m):

$$q(x, \theta_k) := \log(\pi_k) - \frac{1}{2}\log\left(\det(\Sigma_k)\right) - \frac{(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)}{2}. \tag{2.8}$$

The score is composed of three terms. The third term in (2.8) is the squared Mahalanobis distance of point x from the centre. The second term, $-\frac{1}{2}\log\left(\det(\Sigma_k)\right)$, accounts for the extension of the geometric region covered by the cluster (see Anderson, 2003). These two terms have the same

[1] These rule are usually derived under the assumption of Gaussian densities. However, as we will see later, they are much more general. Also, in practical situation they are documented to perform well even in the absence of the validity of the assumptions Hastie, Tibshirani, and Friedman, 2009.

effect on q, which is increased when the two decrease. q is higher for points close to the centre of small-volume clusters. The first term, $\log(\pi_k)$, accounts for cluster's size and positively affects the score. Note that this is always negative and is smaller for small clusters. Overall, the score $q(x, \theta_k)$ may be viewed as the strength with which point x fit into cluster k. Indeed, for point closer to the centre (accounting for cluster size and variability), q is higher.

In a supervised framework, this score is used to assign new points to classes by the MAP estimator. That is, point x is assigned to class k if $k = \arg\max_k q(x, \theta_k)$. We indicate this as $\hat{z}_{i,k} = 1$. Under some conditions, this assignment is optimal in that it minimizes the misclassification rate, defined as the probability that a point arising from class k is assigned to a wrong class k'. This concept can also be extended to the case of clustering.

Too see this, consider the problem of assigning points to K clusters. Consider the family of densities described by:

$$f_k(x) = \left|\Sigma_k\right|^{-1/2} f_0\Big((x-\mu_k)'\Sigma_k^{-1}(x-\mu_k)\Big), \tag{2.9}$$

Σ_k being square positive definite matrices and f_0 is a strictly decreasing positive function and it is a density in $f_0(x'x)$, $x \in \mathbb{R}^p$. This includes elliptic-symmetric densities characterized by centre and scatter (Anderson, 2003). Assume that $f_k(x)$ represents the distribution of points belonging to the k-th group. That is $f_k(x)$ is the class-conditional density for $k = 1, 2, \ldots, K$. Let π_k be the probability that a point is observed from cluster k, with density, f_k, given by (2.9) parametrized at θ_k. Thus, we are in the case of elliptical-symmetric clusters. As usual, Z are the indicator variables for the generating component. Consider a partitioning rule $r(x)$ used to assign points to clusters; r defines A_k, $k = 1, \ldots, K$, disjoint subset in $\mathbb{R}^p$. Then, the misclassification rate is defined as the complement to 1 of the rate of correct assignments from rule r.

$$L(r(x)) := 1 - \sum_{k=1}^{K} P\{Z = k\}\, P\{r(X) \in A_k | Z = k\} = 1 - \sum_{k=1}^{K} \pi_k \int_{A_k} f_k(x)dx. \tag{2.10}$$

It is possible to show (Velilla and Hernández, 2005) that the optimal Bayes rule, minimizing (2.10), is the rule r^* such that:

$$r^*(x) = k \iff x \in A_k^*; \quad A_k^* := \left\{ x \in \mathbb{R}^p : \pi_k f_k(x) = \arg\max_{j=1,\ldots,K} \pi_j f_j(x) \right\}.$$

Were the true parameters (θ_k) known, the same optimal misclassification rate is achieved by an assignment rule based on the quadratic score q in (2.8), assigning points to the cluster for which they show the highest relative quadratic score. Such a rule is called quadratic discriminant rule (QDA). This result can be showed under an additional assumption on clusters' scatters, which Velilla and Hernández, 2005 argue to be not too restrictive in case of an approximate solution.

Proposition 2.2.1. *Assume that: (i) clusters are generated by a mixture model as in (1.1), with component densities as in (2.9); (ii) the true mixture parameters θ_k are known; either (iii.a) $det(\Sigma_k) = det(\Sigma_j) = c$, $i \neq j$, $i, j = 1 \ldots, j$, or (iii.b) $f_k = \phi_k$ (where ϕ_k is a Gaussian density parametrized at μ_k, Σ_k). Then, the assignment rule $z^*(x) = \arg\max_k q(x, \theta_k)$ (see (2.8)), achieves the optimal misclassification rate as defined in (2.10).*

Proof. First notice that if X is distributed according (1.1), then the conditional distribution of $X|Z = k$ has density $f_k(x)$. That is $f_k(x)$ is a component of the mixture density, and coincides with the class-conditional density.
Then, $A_k^* \left\{x \in \mathbb{R}^p : \pi_k f_k(x) = \arg\max_{j=1,\dots,K} \pi_j f_j(x)\right\}$ achieves the optimal misclassification rate.
Consider (iii.a). Now, writing $y_k' y_k = (x-\mu_k)'\Sigma_k^{-1}(x-\mu_k)$, consider $f_0(y_k' y_k)$ and $\exp\left\{-\frac{1}{2}y_k' y_k\right\}$. These are both monotonically strictly decreasing in $y'y$, so that:

$$y_k' y_k > y_j' y_j \implies \exp\left\{-\frac{1}{2}y_k' y_k\right\} > \exp\left\{-\frac{1}{2}y_j' y_j\right\}, \text{ and } f_0(y_k' y_k) > f_0(y_j' y_j).$$

Thus, for all k, j:

$$\begin{aligned}\pi_k f_k(x) > \pi_j f_j(x) &\iff \pi_k \det(\Sigma_k)^{-\frac{1}{2}} f_0(y_k' y_k) > \pi_k \det(\Sigma_j)^{-\frac{1}{2}} f_0(y_j' y_j) \overset{(iii.a)}{\iff} \\ &\pi_k c^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(y_k' y_k)\right\} > \pi_k c^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(y_j' y_j)\right\} \iff \\ &\log\left(\pi_k c^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(y_k' y_k)\right\}\right) > \log\left(\pi_k c^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(y_j' y_j)\right\}\right)\end{aligned}$$

Now, note that $q(x, \theta_k) = \log\left(\pi_k \det(\Sigma_j)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(y_j' y_j)\right\}\right)$. Thus, the rule $z^*(x) = \arg\max_k q(x, \theta_k)$ produces the partition A_k^*, which is also the optimal one. In case (iii.a) is replaced by (iii.b), the result is trivial since $\pi_k f_k$ in the proof above can be simply replaced by $\pi_k \phi_k$, the logarithm of which is equivalent to the quadratic score, but for a constant term. □

It is important to stress that the hypothesis on clusters' scatters used above can be relaxed, leading to a good approximation of the optimal misclassification rate. Thus, using the true parameters θ_k, a clustering solution assigning point based on the quadratic score q, would minimize the misclassification rate in case of elliptic-symmetric clusters with comparable clusters' volumes.

Of course, the true parameters are not known in practice, and needs to be estimated from the data. In the case of classification, Velilla and Hernández, 2005, show that for consistent estimates of centres and scatter matrices (consistent to the hypothesized true mixture parameters), and similar scatter volume across the classes, then the QDA is (approximately) consistent for families of the type (2.9). Meaning that as, $n \to \infty$, QDA will achieve the optimal misclassification rate. This would in principle extend also to a clustering framework, had we had consistent estimates of the parameters. Unfortunately, here there are more subtleties: unknown number of clusters; unknown class memberships.

Note that the assumption (i) in proposition 2.2.1 on the data generating process being a mixture was only needed to achieve an optimal misclassification rate in those cases. However, the QDA can also be used in cases where these assumptions do not hold. This is frequently the case in practical applications. Hastie, Tibshirani, and Friedman, 2009 document the successes of QDA even in these cases. In particular QDA seems to work well when the data can support at most quadratic boundaries for groups splitting.

Based on the previous observation we restrict our attention on those situations where the quadratic score assignment make sense. Therefore for a given clustering solution $\theta(m)$, we

consider the class memberships indicator defined as

$$\hat{z}_{i,k}^{(m)} = z_k(x_i, \theta^{(m)}) = \begin{cases} 1 & \text{if } k = \arg\max_k, q(x_i, \theta^{(m)}) \\ 0 & \text{otherwise.} \end{cases}$$

Recalling that q also gives a measure of the degree with which each point is accommodated to a cluster, we can evaluate how well a point fits into a clustering solution, m, by what we call the hard scoring (HS):

$$HS(x_i, m) := \sum_{k=1}^{K^{(m)}} \hat{z}_{i,k}^{(m)} q\big(x_i, \theta_k^{(m)}\big). \tag{2.11}$$

In the case of overlapping clusters, it may be sensible not to define an hard assignment of points to clusters. We propose to use a smooth degree of membership of points to clusters, computed as:

$$\hat{\tau}_{i,k}^{(m)} = \tau_k(x_i, \theta^{(m)}) := \frac{\exp\left\{q(x_i, \theta_k^{(m)})\right\}}{\sum_{k=1}^{K} \exp\left\{q(x_i, \theta_k^{(m)})\right\}} \in [0, 1]\,.$$

Note that the τ's are obtained by exponentiation of the score attached to a point to belong to a cluster relative to the summed score that the point achieves across all clusters. Therefore this normalized weights will tell how strongly a point x_i is connected to the k-th group according to $\theta(m)$. Any other positive strictly monotonically increasing transform of $q(\cdot)$ other than $exp(\cdot)$ would have produced a similar goal. However, the $exp(\cdot)$ transform used above allows to connect the scoring mechanism to number of well known likelihood-type quantities that are central in the model-based clustering literature. In this sense this will allow to construct a parallel between solutions $m \in \mathcal{M}$ that have large score, with solution pursued by ML-type methods when one assume a mixture model for the underlying data generating process.

In this case, the extent to which a point x is well accommodated in a clustering solution described by m, is given by a *smooth score* (SS):

$$SS(x_i, m) := \sum_{k=1}^{K^{(m)}} \hat{\tau}_{i,k} q\big(x_i, \theta_k^{(m)}\big). \tag{2.12}$$

Finally, the overall adequacy of a clustering solution on observed data $\mathbb{X}_n$ may be evaluated by the average adequacy of points to clusters. This defines the HSC, SSC criteria:

$$HSC(m) := \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K^{(m)}} \hat{z}_{i,k}^{(m)} q\big(x_i, \theta_k^{(m)}\big); \tag{2.13}$$

$$SSC(m) := \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K^{(m)}} \hat{\tau}_{i,k}^{(m)} q\big(x_i, \theta_k^{(m)}\big). \tag{2.14}$$

One wants to choose an $m \in \mathcal{M}$ so as to maximize quantities (2.13) and (2.14). In both cases, the criterion selects the m for which the points are best accommodated into clusters, on average, according to the quadratic score q. In the case of HSC, only points assigned to a cluster participate to the evaluation of that cluster. In the case of SSC, all the point participate

to the evaluation of all the clusters according to their degree of membership to clusters. The contribution to the overall score of each point is given by the quadratic score that the point achieve in the relative cluster.

A final remark on the proposed scoring functions (2.11) and (2.11). Their different behaviour should be accentuated with overlapping or not well-separated clusters. In these cases, one can expect that the weights τ behave much differently with respect to the indicators z. We expect that the smooth version gives higher relative importance to points closer to the cluster centre and that can be assigned with more confidence. Indeed, in HS (2.15) points are weighted either 1 or 0 and contributes to the scoring by (2.8). In the smooth version SS (2.12), points not only contribute by (2.8) (decreasing moving afar from the centre), by they are also weighted by τ which is decreasing when moving towards other cluster centres.

Connection with likelihood theory

The scores HS and SS introduced above can be shown to have connections ((2.11) and (2.12)) with classical likelihood theory in particular cases. In this section, we will see that, under some assumptions HSC and SSC may be seen as selecting the best model m in terms of the likelihood function. In more general cases, they can be seen as selecting an optimal solution in terms of the Kullback-Leibler distance to the underlying data generating process.

Throughout this section, we call with $\phi_k(x)$ a Gaussian density parametrized at μ_k, Σ_k, were it will be clear from the context where the parameters comes from. We may emphasize the dependency on model parameters writing $\phi_k(x, m) = \phi(x; \mu_k^{(m)}, \Sigma_k^{(m)})$. Also, $c = 2\pi^{-p/2}$, where p is the dimensionality of random vector X (see Subsection 1.3.1). We assume also that $\mathcal{M}$ is such that all the quantities used later are well defined for models in it (this, for example, amounts to non singular scatter matrices).

First, we note that the quadratic score (2.8) has a connection with mixtures of Gaussian distributions in the following sense. Consider a redefinition of the scores HS and SS given by:

$$s_h(x; m) := \sum_{k=1}^{K(m)} \mathbb{1}\left\{k = \arg\max_k \left\{\pi_k^{(m)} \phi_k(x; m)\right\}\right\} \log\left(\pi_k^{(m)} \phi_k(x; m)\right) \tag{2.15}$$

$$s(x; m) := \sum_{k=1}^{K(m)} \frac{\pi_k^{(m)} \phi_k(x; m)}{\sum_{k=1}^{K(m)} \pi_k^{(m)} \phi_k(x; m)} \log\left(\pi_k^{(m)} \phi_k(x; m)\right). \tag{2.16}$$

Then s_h and s preserve the same ordering for solutions m as HS an SS. Thus, we may redefine the criteria HSC and SSC by equivalent formulation in term of Gaussian densities. This result is essentially motivated by the usual derivation of the quadratic score, which is obtained under Gaussianity. Thus, in a sense, HSC and SSC may be seen as evaluating the adequacy of clustering solutions, m, according to the cluster that these solutions would imply under a Gaussian mixture model.

Proposition 2.2.2. *For all $m, m' \in \mathcal{M}$ then:*

$$SH(x, m) \geqslant SH(x, m') \iff s_h(x, m) \geqslant s_h(x, m')$$
$$SS(x, m) \geqslant SS(x, m') \iff s(x, m) \geqslant s(x, m')$$

(refer to equations (2.12), (2.11), (2.15) and (2.16)).

Proof. The following proof is for SS. The proof for HS is analogous. For any m:

$$q(x,\theta_k) = \log(\pi_k) - \frac{1}{2}\log\big(\det(\Sigma_k)\big) - \frac{(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)}{2}$$
$$= \log\Big(\pi_k \det(\Sigma_k)^{-\frac{1}{2}} \exp\big\{-\frac{(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)}{2}\big\}\Big) = \log\Big(\pi_k\phi_k(x)\Big) - \log c. \quad (2.17)$$

Note the equivalence:

$$\frac{\pi_k\phi_k(x_i)}{\sum_k \pi_k\phi_k(x_i)} = \frac{\exp(q(x_i,\theta_k))}{\sum_k \exp(q(x_i,\theta_k))} = \hat{\tau}_{i,k}.$$

Finally, since $\sum_{k=1}^{K^{(m)}} \hat{\tau}_{i,k} = 1$, we have:

$$\sum_{k=1}^{K^{(m)}} \hat{\tau}_{i,k} q\big(x_i,\theta_k^{(m)}\big) = \sum_{k=1}^{K^{(m)}} \Big(\hat{\tau}_{i,k}\log\Big(\pi_k\phi_k(x)\Big)\Big) - \log c = s(x,m) - \log c.$$

Note that $\log c$ depends on the dimensionality of x only and is independent of m. Thus it now follows:

$$SS(x,m) \geqslant SS(x,m') \iff s(x,m) - \log c \geqslant s(x,m') - \log c$$
$$\iff s(x,m) \geqslant s(x,m').$$

This completes the proof for SS. For HS, simply note from the above proof that the indicator variables z implied by the two formulation are equivalent. □

Note also that (2.17) reformulates the score q in term of Gaussian densities and a constant correction term.

The assertion that the scores can be looked as evaluating solutions according to an implicit Gaussian mixture model, can be confirmed in the case that the data are truly generated from Gaussian mixture models. In this case, the scores select $m \in \mathcal{M}$ according to likelihood maximization criteria. This is stated in the next two propositions, which show that, under Gaussianity of the true data generating processes, the solutions selected by SSC and HSC are the same solutions that would maximize the complete version of the data likelihood.

Proposition 2.2.3. *Assume that (i) F is a Gaussian mixture model as in (1.1) with K components; (ii) $K(m) = K$. (iii) $\mathbb{X}_n$ is a sample of size n of i.i.d. random variables from F. The solution $m \in \mathcal{M}$ maximizing the HSC given in (2.13), also maximizes the complete log-likelihood of the data, $cl(\theta)$ (see (1.3)).*

$$\arg\max_{m\in\mathcal{M}} HSC(\theta) = \arg\max_{m\in\mathcal{M}} cl(\theta(m))$$

Proof. We drop the dependence on m in what follows. Simply note that:

$$HSC(m) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\hat{z}_{i,k}q(x_i,\theta_k) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\hat{z}_{i,k}\log(\pi_k\phi_k(x_i)) - \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\hat{z}_{i,k}c = \frac{1}{n}cl(\theta) - c,$$

The last equality is motivated as follows. In $cl(\theta)$ the unknown z are replaced via MAP estimators, $\hat{z}_{i,k}^{MAP}$. These are equal to 1 if $k = \arg\max_k \pi_k\phi_k(x)$, 0 otherwise. However, up to the constant c, $q(x_i,\theta_k)$ is equivalent to the logarithm of $\pi_k\phi_k(x)$ (see (2.17)). Thus, $\hat{z}_{i,k}^{MAP} = 1 \iff k = \arg\max_k \pi_k\phi_k(x) \iff k = \arg\max_k q(x_i,\theta_k) \iff \hat{z}_{i,k} = 1$. Finally, since for any given n, $\arg\max_\theta \frac{1}{n}cl(\theta) - c = \arg\max_\theta cl(\theta)$, the result follows. □

Proposition 2.2.4. *Assume that (i) F is a Gaussian mixture model as in (1.1) with K components; (ii) $K(m) = K$; (iii) $\mathbb{X}_n$ is a sample of size n of i.i.d. random variables from F. The solution $m \in \mathcal{M}$ maximizing the SSC given in (2.14), also maximizes the conditional complete log-likelihood of the data, $\mathbb{E}(cl(\theta)|\mathbb{X}_n)$ (see (1.5)):*

$$\arg\max_{m\in\mathcal{M}} SSC(m) = \arg\max_{m\in\mathcal{M}} \mathbb{E}(cl(\theta(m))|\mathbb{X}_n)$$

Proof. We drop the dependence on m in what follows. Note that, conditioning on the data, the expectation in $\mathbb{E}(cl(\theta)|\mathbb{X}_n)$ is taken with respect to the indicator variables z. As discussed in Subsection 1.3.1, these correspond to the posterior probabilities $\tau'_{i,k} = \frac{\pi_k\phi_k(x_i)}{\sum_k \pi_k\phi_k(x_i)}$. Then $\mathbb{E}(cl(\theta)|\mathbb{X}_n)$ is obtained by simply replacing the z's in $cl(\theta)$ with τ's (see (1.5)). Also recall the equivalence $\tau'_{i,k} = \frac{\pi_k\phi_k(x)}{\sum_k \pi_k\phi_k(x_i)} = \frac{\exp(q(x_i,\theta_k))}{\sum_k \exp(q(x_i,\theta_k))} = \hat{\tau}_{i,k}$. Then consider the following:

$$SSC(m) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\hat{\tau}_{i,k}q(x_i,\theta_k) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\hat{\tau}_{i,k}\log(\pi_k\phi_k(x_i)) - \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\hat{\tau}_{i,k}c = \frac{1}{n}\mathbb{E}(cl(\theta)|\mathbb{X}_n) - c,$$

Finally, since for any given n, $\arg\max_\theta \frac{1}{n}\mathbb{E}(cl(\theta)|\mathbb{X}_n) - c = \arg\max_\theta \mathbb{E}(cl(\theta)|\mathbb{X}_n)$, the result follows. □

Finally, in case the underlying process F, is not Gaussian, the SSC criterion will still have the following appealing interpretation: it pursues a good fitting of the data, trying to avoid overfitting. That is, asymptotically, it selects the model that minimize $\{d_{KL}(f||m) + H(\theta(m))\}$, where the first term is the Kullback-Leibler distance between the data generating process, F, and Gaussian mixture model parametrized at $\theta(m)$; $H(\theta(m))$ is an entropy term. The first term is smaller when the model adapts better to the data. An high fitting, however, may require an excessive complexity of the model, causing overfitting (see Subsection 2.1.1). This is taken into account by the entropy term, which increases when the assignment of points to clusters in not clear: typically, models that are too complex will have many strongly overlapping clusters, which in turns implies unclear points assignments. Note that the adequacy of the fitting is evaluated

in terms of a Gaussian mixture model implied by the parameters of the clustering solution $\theta(m)$. This further clarify in which sense the scores can be seen as evaluating solutions according to implicit Gaussian mixture models. However, we remark that the solutions given by SSC or HSC do not aim to recover nor assume a true model for the data.

This, is stated in the next two proposition. The second shows that, under the additional assumptions that the true data generating process is a Gaussian mixture and that $\mathcal{M}$ contains it, this principle reduces to selecting $m \in \mathcal{M}$ maximizing the expected likelihood in case of well separated clusters.

Proposition 2.2.5. *Assume that (i) F has continuous density f. (ii) Fix $K(m) = K$ (iii) $\mathbb{X}_n$ is a sample of size n of i.i.d. random variables from F. Then, as $n \to \infty$, the solution $m \in \mathcal{M}$ maximizing the smooth score SSC given in (2.14), is such that*

$$\arg\max_{m \in \mathcal{M}} \left\{ \lim_{n \to \infty} SSC(m) \right\} = \arg\min_{m \in \mathcal{M}} \left\{ d_{KL}(f||m) + H(\theta(m)) \right\},$$

where: d_{KL} is the Kullback-Leibler distance between f and a Gaussian mixture model (see (1.1)) parametrized at $\theta(m)$; H is an entropy term defined as $H_n(\theta) = \mathbb{E}\left\{h(\theta)|\mathbb{X}_n\right\} = \sum_{i=1}^{n} \sum_{k=1}^{K} \tau_{i,k} \log(\tau_{i,k})$ and $H(\theta) = \mathbb{E}_X \sum_{k=1}^{K} \tau_k(X, \theta) \log(\tau_k(X, \theta))$, where τ's are defined by (1.6).

Proof. Consider the log-likelihood of the Gaussian mixture model in $\theta(m)$. We define its density with $m(x, \theta(m))$. The log-likelihood is $l_n(\theta(m))$ (compare with (1.1) and (1.4); we drop the dependency on m in what follows). Define the complete log-likelihood as the complete version of l_n, as in (1.5). Than, with the expansions as in Subsection 1.3.1 (see in particular (1.7)), we can expand the Kullback-Leibler distance as (a is a constant; see Burnham and Anderson, 2003):

$$\begin{aligned} d_{KL}(f||m) = a - \mathbb{E}_{X \sim F}\left(\log m(x, \theta)\right) = a - \frac{1}{n}\mathbb{E}_{X \sim F}\left(l_n(\theta)\right) = \\ a - \frac{1}{n}\mathbb{E}_{X \sim F}\left(\mathbb{E}(cl_n(\theta)|\mathbb{X}_n) - H_n(\theta)\right) = c - \frac{1}{n}\mathbb{E}_{X \sim F}\left(\mathbb{E}(cl_n(\theta)|\mathbb{X}_n))\right) - H(\theta) \\ = c - \mathbb{E}_{X \sim F}\left(\mathbb{E}(cl_{n,1}(\theta)|\mathbb{X}_n))\right) - H(\theta). \end{aligned}$$

$cl_{n,1}$ emphasize that we consider a single summand in cl (compare with (1.5)). Furthermore, note that

$$\arg\min_{m \in \mathcal{M}} \left\{ d_{KL}(f||m) + H(\theta(m)) \right\} = \arg\max_{m \in \mathcal{M}} \mathbb{E}_{X \sim F}\left(\mathbb{E}(cl_{n,1}(\theta(m))|\mathbb{X}_n))\right).$$

Finally, using proposition 2.2.4, as $n \to \infty$, by the law of large numbers (e.g. see Bierens, 1996) SSC, for any fixed $m \in \mathcal{M}$, converges to the quantity on the right-hand side:

$$\lim_{n \to \infty} SSC(m) \xrightarrow{p} \mathbb{E}_{X \sim F}\left(\mathbb{E}(cl_{n,1}(\theta(m))|\mathbb{X}_n))\right).$$

Thus, the result follows. □

A similar result also holds for HSC, replacing the expected complete log-likelihood above with the complete log-likelihood ((1.5)) and defining H not taking the expectation over the assignment variables (Z), but replacing them via the MAP estimator.

Now, if we specialize the above to the case where f is also a Gaussian mixture model, with well separated components, we obtain a parallel intuition in terms of maximum likelihood.

Proposition 2.2.6. *Assume that assumptions of proposition 2.2.5 are satisfied. Additionally assume that: (i) f is a Gaussian mixture model with K groups, parametrized at θ_0. (ii) f generates well separated components, in the sense that $P\{x \in \mathbb{R}^p : H(\theta_0) > 0\} < \epsilon$, for some $0 < \epsilon \ll 1$; (iii) $m^* \in \mathcal{M}$ and $\theta(m^*) = \theta_0$. Then, with probability $1 - \epsilon$:*

$$m^* \in \arg\max_{m \in \mathcal{M}} \left\{ \lim_{n \to \infty} SSC(m) \right\} = \arg\max_{m \in \mathcal{M}} \mathbb{E}(f(x, \theta(m))).$$

Proof. First note that by assumptions (ii) and (iii), on a set of probability $1 - \epsilon$, $\arg\min_{m \in \mathcal{M}} \{d_{KL}(f||m) + H(\theta(m))\} = m^*$, since at m^*, a Gaussian mixture model parametrized at $\theta(m^*) = \theta_0$ coincides with f. Thus $d_{KL}(f||m^*) = 0$, and $H(\theta_0) = 0$. Also, by expansion of $l(\theta)$ and $cl(\theta)$ in Subsection 1.3.1, when $H_n(\theta) = 0$, $cl_n(\theta) = l_n(\theta)$. By (ii) this happens with probability $1 - \epsilon$. Thus, within a set with probability $1 - \epsilon$, with the same line of proof of proposition 2.2.5:

$$\mathbb{E}_{X \sim F} \left(\mathbb{E}(cl_{n,1}(\theta(m))|\mathbb{X}_n)) \right) = \mathbb{E}_{X \sim F} \left(l_{n,1}(\theta(m)) \right) = \mathbb{E}_{X \sim F} f(X, \theta(m))$$

where the last equality is motivated by (i). Now the result follows from the convergence of SSC to $\mathbb{E}\left(\mathbb{E}(cl(\theta)|\mathbb{X}_n)\right)$ as in the proof of proposition 2.2.5. □

A similar results also holds for HSC, noting that under the strong separation hypothesis, the indicators z and the posteriors τ, are such that $z(x, \theta_0) \approx \tau(x, \theta_0)$, with high probability, at the true mixture parameters θ_0.

Further insights on HSC and SSC

As shown in the previous Section, cluster configurations that maximizes $HSC(m)$ ((2.13)) or $SSC(m)$ ((2.14)) may have an interpretation in terms of clusterings obtained based on likelihood-type procedures when a mixture model for the underlying data distribution is assumed. As seen previously, the connection holds only in specific cases (e.g. number of groups K fixed and Gaussian class-conditional densities). Here we want to remark that, more generally, the scoring approach proposed in this work does not aim to recover a true underlying generating model, if it one ever exists, but it tries to assess whether for a model m the K groups of points are well fitted into $K(m)$ ellipsoids described by $\theta(m)$. In general, a solution that is highly ranked based on SSC or HSC, may not be related the model m that well represents the underlying true data generating process. To see this let us consider the following example.

Figure 2.3 shows the behaviour of $\mathbb{E}_X\, SSC(m)$ for two datasets generated by a mixture distribution with two components; for each dataset we compare the SSC of two different models:

$$\begin{aligned} m: &\quad K(m) = 1, \quad \pi^{(m)} = 1, \quad \Sigma^{(m)} = I, \quad \mu^{(m)} = \mu/2; \\ m': &\quad K(m') = 2, \quad \pi_k^{(m')} = 0.5, \quad \Sigma_k^{(m')} = I, \quad \mu_1^{(m')} = 0, \ \mu_2^{(m')} = \mu, \end{aligned}$$

where μ is the true centre of the second component of the mixture generating the data. Thus,

the quoted models m and m' are tied to the true data. By moving μ closer to or away from 0, we obtain data with more or less overlapping components. We can then study the behaviour of SSC for different datasets with different distances between clusters' centres. If the distance is larger than a threshold, SSC prefer models that fuse the clusters together. Computing this threshold is not at all straightforward, and depend on the specific distribution of the data (for a given distribution, they can be computed approximately via numerical integration). In the first row (data from Gaussian mixture), model m' coincides with the true mixture density. We find that SSC chooses to represent the mixture with a single component when the amount of overlap of the Gaussian densities is about 30% (shown in Figure 2.3). Beyond this threshold, the entropy term (higher for higher overlaps) does no longer offset the gains obtained by fitting the data with two components. For the shown uniform mixture, one component model is preferred even if the *true* overlap is 0%: the true components' separation needs to be very large for the components to be treated as two groups. Indeed, note that the entropy as per evaluated by SSC is still higher than 0 unless the two uniform densities are not too far apart.

Figure 2.3: SSC behaviour. (First Row) Data generating process: Gaussian mixture model with two spherical and equal components centred at $(0, 0)$ and (μ, μ). (Second Row) DGP: Mixture of uniforms centred at $(0, 0)$ and $(\mu, 0)$. (First Column) Generating mixture at critical distance. (Second Column) Sample from first column's mixtures with models m and m' contours. (Third Column) Value of $\mathbb{E}\, SSC$ for models m and m' at changing μ. At the left of the intersection point (critical distance) SSC prefers m, with one component; at right, model m' with two components is preferred.

2.2.1 Random cluster solutions and scoring

Up to now, throughout the discussion in Section 2.2, we assumed that the models m, elements of $\mathcal{M}$, were a fixed collection of clusters' centre, scatter and size, $\{\theta(m)\}_{m\in\mathcal{M}}$. In practice, we do not usually work with an exogenous, finite list of these. Rather, it is way more typical to obtain clustering solutions as output of clustering or algorithms evaluated on the data. That is, specifying a clustering method m, this takes data, $\mathbb{X}_n$, as input and returns an estimated clustering solution $m(\mathbb{X}_n)$.

For example, imagine that $m' \in \mathcal{M}$ specifies a clustering model obtained by the k-means algorithm, with a pre-specified number of clusters, $K(m')$. Then, we compute a solution on observed data, i.e. $m'(\mathbb{X}_n)$, and obtain the $K(m')$-means partition of $\mathbb{X}_n$. We may then retrieve a configuration of centres, scatters and sizes $\theta(m'(\mathbb{X}_n))$ by computing the sample means, scatters and sizes of the points assigned to each cluster according to solution $m'(\mathbb{X}_n)$.

Thus, we see that the solutions we are willing to score, $m(\mathbb{X}_n)$, are random variables, depending on the underlying data generating process, F, that generates samples, $\mathbb{X}_n$. To simplify the notation, let us call these as $\hat{\theta}_n(m) := \theta(m(\mathbb{X}_n))$. We call $G_{n,m}$ the distribution of $\hat{\theta}_n(m)$. Also, let us emphasize the dependency of SSC and HSC on the sample data and on the collections of centres, scatters and sizes, by writing $S(\mathbb{X}_n, \theta)$ (see (2.13) and (2.14)).

Remark 2.2.1. *From now onwards, S will refer to any of SSC or HSC (or any other scoring criterion that shows conforming structure). Thus, for example (see (2.12))*

$$S(\mathbb{X}_n, \theta) = \int_{\mathbb{R}^p} s(x_i, \theta(m)) dF_n(x), \tag{2.18}$$

where s is either HS or SS (see (2.11) and (2.12)), $\theta = \theta(m)$ and F_n is the empirical cumulative distribution function of the data.

Now, in order to score solutions, $\{m(\mathbb{X}_n)\}$, it would be tempting to estimate these on data and plug them in into the score, so as to evaluate

$$S\Big(\mathbb{X}_n, \hat{\theta}_n(m)\Big).$$

Furthermore, under suitable conditions (like for example consistency of $\hat{\theta}_n(m)$ to a scalar vector $\theta_0(m)$ and regularity conditions for a uniform law of large numbers), we may expect to obtain consistent estimates of the population counterpart:

$$\lim_{n\to\infty} S\Big(\mathbb{X}_n, \hat{\theta}_n(m)\Big) \to \mathbb{E}_X\, s(X, \theta_0(m)).$$

Although, with finite samples, this may be a poor solution. Indeed, similarly to the motivations underlying the AIC (Akaike, 1973) and to the discussions in Subsection 2.1.1, this approach will typically incur into overfitting. That is, when the complexity of the model is higher, the adaptation of the fit to a given sample may be excessive. If we used the same sample information to evaluate the score, we would get good results which are only due to this excessive fit. Imagine computing the score on a sample using a solution that, given a sample, regards each observed point as a cluster. This is of course a poor solution. However, referring to the intuition

on the trade off between fitting and entropy exposed in the previous section, we may achieve, upon evaluating it on the same sample used for estimation, a perfect assignment (0 entropy) and a perfect fitting. This in-sample evaluation of a fitted model does not take into account the introduced variability due to model estimation. This overfitting problem will also clearly emerge from the empirical analysis in Section 2.4.

An ideal solution to this problem would be the following. Suppose that we observed multiple independent samples from F, say $\left\{\mathbb{X}_n^{(b)}\right\}$, for $b = 1, \ldots, B$. Then, we could form multiple clustering solutions $\theta(m(\mathbb{X}_n^{(b)})) = \hat{\theta}_n^{(b)}(m)$. Now we could evaluate:

$$S\Big(\mathbb{X}_n^{(1)}, \hat{\theta}_n^{(2)}(m))\Big).$$

This would solve the overfitting problem. In fact, in the example proposed above, the one-point one-cluster solution would likely score very poorly when evaluated on a different sample than the one used for fitting $\hat{\theta}_n^{(2)}(m)$.

Building on this, we could also take into account the variability of the estimated solutions at different samples. We could then evaluate a clustering model via the following average, less affected by estimates' variability:

$$\frac{1}{B-1}\sum_{b=2}^{I} S\Big(\mathbb{X}_n^{(1)}, \hat{\theta}_n^{(b)}(m))\Big).$$

Under regularity conditions, assuming that $G_{n,m} \xrightarrow[n\to\infty]{} G_m$, then as $n \to \infty$ and $B \to \infty$, this quantity would converge to its population counterpart, that is:

$$\lim_{B,n\to\infty} \frac{1}{B-1}\sum_{b=2}^{I} S\Big(\mathbb{X}_n^{(1)}, \hat{\theta}_n^{(b)}(m))\Big) \to \mathbb{E}_{G_m}\, \mathbb{E}_F\, s(X, \theta_0(m))), \tag{2.19}$$

where G_m is the distribution of $\theta_0(m)$ (possibly a random variable), the limit value of $\hat{\theta}_n(m)$.

Note that the right-hand side of (2.19) computes the average score (under F) at solutions obtained fitting m on full information on F; then these solutions are averaged together. In practical situation, it is often the case that $\theta(m(F)) = \theta_0(m)$, and G_m is a degenerate distribution. In such cases, it would also be possible to estimate a confidence interval for $\mathbb{E}_F\, s(X, \theta_0(m))$, which will allow to better account for variability induced by model estimation.

Of course, this procedure is not feasible in reality. It would require multiple independent samples from F, while we only observe one, namely $\mathbb{X}_n$. We propose a simple resampling procedure to mimic this scheme and estimate the right-hand side of (2.19). That is, instead of using multiple samples from F, we use multiple samples from F_n, the empirical distribution function of $\mathbb{X}_n$. We illustrate our approach in the next section.

2.3 A resample approach to score cluster solutions

In Subsection 2.2.1 we introduced the limitations of an in-sample approach when the solution m is fitted on sample data. Also, we argued that a resampling scheme can be used to cope with this problem. Here we will illustrate our proposal.

We recall that the quantity of interest here is the double expectation on the right-hand side of (2.19)

$$\mathbb{E}_{G_m}\,\mathbb{E}_F\, s(X, \theta_0(m)), \tag{2.20}$$

where $X \sim F$, $\theta_0(m) \sim G_m$ and $\lim_{n\to\infty} G_{n,m} \to G_m$, where $\hat{\theta}_n(m) \sim G_{n,m}$

This is the quantity we are willing to estimate. That is, the average over all realizations of $\theta_0(m)$ of the average average score S (2.18). The outer expectation on G_m allows to take into account the variability introduced due to the estimation of $\theta(m)$; the inner expectation over F allows to score a solution at F. In the case of a degenerate G_m, we are also interested in confidence intervals for $E_F s(X, \theta_0(m))$.

Since multiple independent samples from F are not available (see Subsection 2.2.1), we propose to resample form F_n, the empirical cumulative distribution function (ECDF) of the sample data. The resamples will be used to fit multiple solutions for a certain $m \in \mathcal{M}$. These will be used to approximate the outer expectation in (2.20). The original sample, instead, will be used to approximate the inner expectation. The procedure stated in Algorithm 2 gives the pseudo-code for the estimation of the quantities of interest when a clustering method corresponding to $m \in \mathcal{M}$ is performed on the sample at hand.

Algorithm 2: Resampling Scheme

Input: sample, $\mathbb{X}_n$; model m; scoring function S; a fixed integer B; a fixed $\alpha \in (0, 1)$.
Output: $S_n^*(m)$; $L_n^*(m)$; $U_n^*(m)$.

for $b = 1, \ldots, B$ **do**

- **STEP 1:** extract $\mathbb{X}_n^{*(b)}$ resampling from F_n (ECDF)
- **STEP 2:** fit the solution $\hat{\theta}_n^{*(b)}(m)$ on $\mathbb{X}_n^{*(b)}$
- **STEP 3:** compute $S_n^{*(b)}(m) \leftarrow S(\mathbb{X}_n, \hat{\theta}_n^{*(b)}(m))$

STEP 4: compute $S_n^*(m) \leftarrow \frac{1}{B}\sum_{b=1}^{B} S_n^{*(b)}(m)$

STEP 5: compute

$$H_n^{*(b)}(m) \leftarrow a_n\big(S_n^{*(b)}(m) - S_n^*(m)\big),$$

for an appropriate scaling sequence $\{a_n\}$ (see below).

STEP 6: Compute the empirical quantiles of the root:

$$L_n^*(m) \leftarrow \inf_t \left\{ t : \frac{1}{B}\sum_{b=1}^{B} \mathbb{1}\left\{ H_n^{*(b)}(m) \leqslant t \right\} \geqslant \frac{\alpha}{2} \right\}$$

$$U_n^*(m) \leftarrow \inf_t \left\{ t : \frac{1}{B}\sum_{b=1}^{B} \mathbb{1}\left\{ H_n^{*(b)}(m) \leqslant t \right\} \geqslant 1 - \frac{\alpha}{2} \right\}$$

Specialization of the algorithm in case of bootstrap resampling:
scaling sequence becomes $a_n = \sqrt{n}$. Also Step 1 becomes:

STEP 1b: $\mathbb{X}_n^{*(b)} = \{X_1^*, \ldots, X_n^*\}$, is an i.i.d. sample from F_n, i.e. it is obtained from resampling the original sample with replacement according to the empirical measure.

The procedure is described and motivated as follows (refer to Algorithm 2).

STEP 1 Here we take a sample from F_n, the ECDF of the sample data, according to some resampling scheme. The resampling scheme needs to guarantee that resamples $\mathbb{X}_n^{*(b)}$ are

independent across b's. These resamples will serve as the basis to estimate the outer expectation $\mathbb{E}_{G_m}$ in (2.20). This mimics the idea of observing multiple sample from F, and will be used to form independent fit of $m \in \mathcal{M}$ (compare with Subsection 2.2.1).

STEP 2 Once a resample from F_n has been obtained, we fit the solution $\theta(m)$ on it. This will produce a clustering solution from which we retrieve the usual vector of centres, scatters and sizes $\hat{\theta}_n^{*(b)}(m)$. Due to the resampling scheme, these vectors are independent one another and they are i.i.d. takes from the distribution $G_{n,m}^*$. This distribution will be used to approximate $\mathbb{E}_{G_m}$ in (2.20).

STEP 3 Here we score solutions from STEP 2, obtaining $S_n^{*(b)}(m)$. The scoring is computed according to S (e.g. SSC or HSC) on the sample $\mathbb{X}_n$. Thus, the original sample data is used to approximate the inner expectation, $\mathbb{E}_F$ in (2.19).

(B) It is the number of times that STEP 1, STEP 2, STEP 3 are repeated. The higher the value of B the more precise the approximation of $\mathbb{E}_{G_m}$ in (2.20). However, higher B usually comes with higher computational costs. Typically, the majority of this cost is due to multiple fitting of a certain clustering strategy $m \in \mathcal{M}$.

STEP 4 Once we iterated STEP 1-3 for B times, we compute the sample average of the scores obtained via STEP 3:

$$S_n^*(m) := \frac{1}{B}\sum_{b=1}^{B} S_n^{*(b)}(m). \tag{2.21}$$

Note that this quantity account for both $\mathbb{E}_{G_m}$ (via resamples; see STEP 1 and STEP 2) and $\mathbb{E}_F$ (via the original sample; see STEP 3). This will allow us to consistently estimate (2.20).

STEP 5 As we anticipated, the independence of the resampled quantities, allows us to construct asymptotic confidence intervals for (2.20). Therefore, we compute a rescaled and centred version of the resampled scores $S_n^{*(b)}(m)$,

$$H_n^{*(b)}(m) := a_n\big(S_n^{*(b)}(m) - S_n^*(m)\big). \tag{2.22}$$

The previous quantity is usually called the "root" in the resampling literature. Based on Proposition 2.3.2 the distribution of these quantities can be used to construct the confidence interval in the next step.

STEP 6 Empirical quantiles of (2.22) are computed. The following analysis justifies the use of these quantiles to approximate the $(1-\alpha)$-level confidence interval for (2.22) with $L_n^*(m), U_n^*(m)$. In other words this implements the idea of getting bootstrap confidence interval via the simple "percentile method" approximation.

$$P\left\{S_n^*(m) - \frac{U_n^*(m)}{a_n} \leqslant S(\mathbb{X}_n, \theta_0(m)) \leqslant S_n^*(m) - \frac{L_n^*(m)}{a_n}\right\}.$$

(STEP 1b) Algorithm 2 can be adapted to alternative resampling schemes. When resampling is performed according to the bootstrap idea of Efron, 1979, STEP 1 becomes STEP 1b

and the scaling sequence becomes $a_n = \sqrt{n}$.

In principle, the procedure described above can be used with any resampling algorithm. Bootstrap (Efron, 1979) is one of the most popular and is also the one we adopt here. For simplicity, in the following exposition we will sometimes refer to this as a bootstrap procedure, naming it after the resampling scheme. Nonetheless, this *is not* a standard bootstrap procedure.

Remark 2.3.1. *Bootstrap resampling is only used to obtain independent resamples from F_n (compare with STEP 1b and STEP 2 of algorithm 2). Note that in order to approximate confidence intervals we construct*

$$H_n^{*(b)}(m) := a_n\big(S_n^{*(b)}(m) - S_n^*(m)\big).$$

This is not the standard root employed in typical bootstrap procedures, as resampled quantities are not centred on a consistent in-sample analogue (e.g. see Mammen, 2012); rather they are centred with respect to a quantity computed via the resampling itself. Also, these quantities are defined with respect to two independent random variables (conditioning on F_n). From now onward we stick on the resampling with replacement from the empirical measure, and therefore we call the resamples "bootstrap samples".

Under suitable conditions, we will show that:

$$\lim_{n,B\to\infty} S_n^*(m) = \lim_{n,B\to\infty} \frac{1}{B}\sum_{b=1}^{B} S_n^{*(b)}(m) \to \mathbb{E}_{G_m}\,\mathbb{E}_F\, s(X, \theta_0(m)).$$

We are also able to show that, for the smooth scoring function SS (i.e. s is as in (2.12)), with G_m degenerate, and other regularity conditions that the following limit exists:

$$H_n^{*(b)}(m) := \sqrt{n}\big(S_n^{*(b)}(m) - S_n^*(m)\big) \xrightarrow{d} D.$$

This last result is needed to justify the use of the percentile method in STEP 6 of Algorithm 2, so that

$$\lim_{n,B\to\infty} P\left\{S_n^*(m) - \frac{U_n^*(m)}{a_n} \leqslant S(\mathbb{X}_n, \theta_0(m)) \leqslant S_n^*(m) - \frac{L_n^*(m)}{a_n}\right\} = 1 - \alpha, \tag{2.23}$$

(which can also be shown to be valid for $\mathbb{E}_F\, S(X, \theta_0(m))$).

These quantities may be then used in several ways to compare models and select an element of $m \in \mathcal{M}$. For example, one could select the solution m maximizing (2.21), which leads to a consistent estimate of the desirable target (2.20). However, in order to better account for variability induced by the fit of $m \in \mathcal{M}$ (compare also with Subsection 2.2.1) we advocate to choose using the lower limit of the confidence interval (2.23). This lead us to the introduction of the BHSC and BSSC criteria, defined as (see (2.13) and (2.14)):

$$BHSC(m) := S_n^*(m) - \frac{U_n^*(m)}{a_n}; \quad S = HSC; \tag{2.24}$$

$$BSSC(m) := S_n^*(m) - \frac{U_n^*(m)}{a_n}; \quad S = SSC. \tag{2.25}$$

Then, we choose $m \in \mathcal{M}$ so as to maximize BHSC and BSSC. This will help hedge better against the additional variability introduced by the fit $\hat{\theta}_n(m)$ induced by the implementation of the corresponding clustering strategy. We want to select solutions for which the "worst" scenario (as taken into account by the lower bound of the confidence interval) is the best among all alternatives.

The validity of the procedure, and the desirability of the BHSC and BSSC criteria is strongly documented by the empirical analysis in Section 2.4 and the theoretical developments in Subsection 2.3.1.

2.3.1 Theoretical analysis of the resampling algorithm

In this section, we present the main theoretical results on the bootstrap resampling procedure illustrated in Algorithm 2. The notation used so far needs to be enriched, and we do this in a dedicated section for convenience. We will establish theoretical results using the equivalent scoring functions using Gaussian densities, (2.15) and (2.16). This is done to keep notation as simple as possible. However, resorting to proposition 2.2.2 and (2.17), the same results hold also for HS and SS (see (2.11), (2.12)).

In this section we will prove consistency for the double expectation (2.21), and, in the case of the smooth score (2.16), convergence of the root (2.22) and asymptotic coverage of interval (2.23). Also, a series of other ancillary results are proved.

To keep the focus on the resampling procedure, we move at the end of the section all additional results for the scoring function.

Notation

We adopt the following notation.

$(\Omega, \mathcal{A}, P)$
: Is the underlying probability space. Where Ω is the sample space, ω are elements in Ω and $\mathcal{F}$ is a σ-algebra on the sample space. P is the probability measure defined on the measurable space $\left(\Omega, \mathcal{F}\right)$.

$X(\omega)$
: random variable defined on the space $(\Omega, \mathcal{A}, P)$. $X(\omega) \in \mathbb{R}^p$ for some integer $p > 0$.

F Distribution function of the data generating process:

$$t \in \mathbb{R}^p, \quad F(t) := P\left\{\omega \in \Omega : X(\omega) \leqslant t\right\}.$$

$\mathbb{X}_n$ An observed sample of random variables from F, $\mathbb{X}_n = \{X_1 \dots X_n\}$, $X_i \sim F$

F_n Empirical cumulative distribution function ($ECDF$) of the sample:

$$t \in \mathbb{R}^p, \quad F_n(t) := \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{X_i \leqslant t\}.$$

P^* Is the bootstrap estimator of P.[2] More precisely, for the standard bootstrap, which we are going to analyse in the following, let X be a random vector on the space $(\Omega, \mathcal{A}, P)$, with distribution F. Then, let F_n the ECDF of a sample of size n from F. Then, P^* is the distribution of i.i.d. random vectors each with ECDF given by F_n (see Andrews (2002); more rigorous and general definition is given in Gonçalves and White (2004)). Note that $P^* = P^*_{n,\omega}$, that is, P^* depends on a particular realization ω and on the sample size n. To keep notation as clean as possible, we are going to drop these dependencies. We will specify them when added clarity is needed.

$\mathbb{X}^*, \mathbb{Y}^*$

Indicates bootstrap samples extracted independently one of each other from $\mathbb{X}_n$. Each bootstrap sample is made of independent random resamples with replacement (as for the standard non-parametric bootstrap). The size of the samples will be indicated by a subscript. Thus, for example, $\mathbb{Y}^*_l = \{X^*_1, \dots, X^*_l\}$, where $X^*_i \sim F_n$ for $i = 1, \dots, l$ and $\mathbb{X}^*_h = \{X^{**}_1, \dots, X^{**}_l\}$, where $X^{**}_i \sim F_n$ for $i = 1, \dots, h$; $\mathbb{X}_h \perp \mathbb{Y}_l$. Clearly, each sample extracted this way depends on the sample size n as well. Formally, we should write $\mathbb{X}^*_{n,l} = \{X^*_{n,1} \dots, X^*_{n,l}\}$. However, to simplify notation, we make this dependence implicit and drop the n subscript.

$\mathcal{M}$ A set of candidate strategies m used to represent the data through $\theta(m)$. As usual, each of them implies a fixed $K(m)$ and parameters $\theta(m) = \left((\pi^{(m)}_k, \mu^{(m)}_k, \Sigma^{(m)}_k)_{k=1\dots K}\right)$. These are estimated on data in the sense explained in Subsection 2.2.1. In the present discussion, we will almost always drop the dependence on m. The analysis is conducted for a generic fixed m.

θ^*_n, θ_n, θ

In this section we will typically be interested in the parameters retrieved from the fitting of a certain $m \in \mathcal{M}$. This is a change with respect to previous notation. These coincides with $\hat{\theta}^*_n(m) = \theta(m(\mathbb{X}^*_n))$, $\hat{\theta}_n(m) = \theta(m(\mathbb{X}_n))$, $\theta_0(m) = \theta(m(F))$ (compare with Subsection 2.2.1). Θ defines the space where the parameters can take value for a given $m \in \mathcal{M}$.

S_n, S, s

Scoring function (sample version and expected value, respectively) used to evaluate a $m \in \mathcal{M}$. This is a change with respect to previous notation. We need to emphasize the dependence on sample size here.

$$S_n(\mathbb{X}_n, \theta) := \frac{1}{n} \sum_{x_i \in \mathbb{X}_n} s(X_i, \theta); \qquad S(\theta) := \int_{\mathbb{R}^d} s(x, \theta) dF(x).$$

Here we recall that s is either one of (2.15) or (2.16), unless differently specified (see also the incipit of this section).

B Number of bootstrap replicates. Superscript (b) will be used when needed to indicate a particular bootstrap iteration.

[2] Bootstrap was introduced in Efron, 1979; an accessible detailed survey is in Efron and Tibshirani, 1994; its theoretical properties were fully developed later (see Bickel and Freedman, 1981 and Giné and Zinn, 1990.

n', n''

In the following, we will make use of subsequence arguments. To ease the notation, as in Gonçalves and White, 2004, we adopt the notation n' and n'' that is intended as follows. For a sequence (n), n' indicates a subsequence of (n); further subsequences of n' are denoted with n''. Formally, for a sequence (n), (n_k) is a subsequence of (n), and (n_{k_j}) is a further subsequence of (n_k) (compare with Bierens, 1996–*Theorem 2.7.1* or Billingsley, 2013–*Theorem 20.5*).

The above leads to the following notational changes (refer to Algorithm 2). What was previously denoted as

Old notation	New notation
$S_n^{*(b)}(m) = S(\mathbb{X}_n, \hat{\theta}_n^{*(b)}(m))$	$S_n^{*(b)} = S_n(\mathbb{X}_n, \theta_n^{*(b)})$
$\mathbb{E}_{G_m}\, \mathbb{E}_F\, s(X, \theta_0(m))$	$\mathbb{E}_\theta\, S(\theta)$

Borrowing from Gonçalves and White (2004), we are going to use the following notation: a bootstrap statistic $T^* := T_n(\mathbb{X}_l^*, \omega)$ is said to converge to 0 in probability–P^*, almost sure–P if there exists a set $A \in \mathcal{F}$ such that $P(A) = 1$ and for $\omega \in A$ we have $\lim_{n,l\to\infty} P^*\{\|T^*\| > \epsilon\} = 0$. Similarly, we say the convergence happens in probability–P^*, probability–P if for any $\epsilon > 0$ and $\delta > 0$, $\lim_{n,l\to\infty} P\{\omega : P^*\{\|T^*\| > \epsilon\} > \delta\} = 0$. Note that by a subsequence argument, for any quantity converging in probability–$P^*_{n,\omega}$, probability–P, we can find for any subsequence $n' \in \mathbb{N}$ a further subsequence n'' such that the convergence holds in probability–$P^*_{n'',\omega}$, almost surely–P; this allows us to move back and forth from almost sure convergence and convergence in probability for P. Finally, T^* converges in distribution–P^*, almost surely–P to F if there is $A \in \mathcal{F}$, $P(A) = 1$ such that for $\omega \in A$, and for any $\epsilon > 0$, $n > \bar{n}$, for some $\bar{n} \in \mathbb{N}$: $P\{\sup_t |P^*\{T^* \leqslant t\} - P(F \leqslant t)| > \epsilon\} = 0$.

Asymptotic properties of the resampling procedure

In order to establish consistency properties for (2.21) and (2.23) we need two preliminary results.

The first results establishes the convergence for each single summand of (2.21). This result basically extend the uniform convergence of random function of random variables to the case of bootstrap random variables. The analysis of the bootstrap procedure will be presented as follows: (i) in Proposition 2.3.1 and Proposition 2.3.2 we prove the desired results general assumptions involving s, F, P and P^*; (ii) secondly we prove that some of this assumptions are fulfilled for the hard and the smooth scoring ((2.11) and (2.12)) under fairly weak conditions on the data generating process F.

Proposition 2.3.1. *Let $(\Omega, \mathcal{F}, P)$ be a probability space, and let X be a random variable with distribution F on this space, with $X(\omega) \in \mathbb{R}^p$ for some finite integer p. Let $X_1(\omega), X_2(\omega), \ldots$ be an infinite sequence of independent and identically distributed random variables; $\mathbb{X}_n := \{X_1, \ldots X_n\}$ being the first n terms. Let F_n be the ECDF of $\mathbb{X}_n$; $\mathbb{X}_h^*$, $\mathbb{Y}_l^*$ are two standard bootstrap samples from $\mathbb{X}_n$, of size h and l respectively. Let $\Theta \subseteq \mathbb{R}^d$, for some finite integer d. $S_n : \prod_{i=1}^n \mathbb{R}^p \times \Theta \to \mathbb{R}$ random functions $S : \Theta \to \mathbb{R}$ an almost sure continuous random function of θ. Let $\theta \in \mathbb{R}^p$; $\theta_n := \hat{\theta}(\mathbb{X}_n) \in \mathbb{R}^p$; $\theta_l^* := \hat{\theta}(\mathbb{Y}_l^*) \in \mathbb{R}^p$. Assume that:*

(b.i) for all $l, n = 1, 2, \ldots$, $P(P^*(\theta_l^* \in \Theta)) = 1$, $P(\theta_n \in \Theta) = 1$ *and* $P(\theta \in \Theta) = 1$.

(b.ii) (Convergence of the estimator in conditional probability) for any $\epsilon > 0$, $\delta > 0$ *as* $n, l \to \infty$*:*

$$\lim_{n\to\infty} P\Big\{ \|\theta_n - \theta\| > \epsilon \Big\} = 0$$
$$\lim_{n,l\to\infty} P\Big\{ P^*\big\{ \|\theta_l^* - \theta_n\| > \epsilon \big\} > \delta \Big\} = 0$$

(b.iii) (Uniform convergence of S_n*) for any* $\epsilon > 0$*:*

$$\lim_{n\to\infty} P\Big\{ \sup_{t\in\Theta} |S_n(\mathbb{X}_n, t) - S(t)| > \epsilon \Big\} = 0$$

Then, for any $\epsilon > 0$,$\delta > 0$*:*

$$\lim_{n,l\to\infty} P\Big\{ P^*\big\{ |S_n(\mathbb{X}_n, \theta_l^*) - S(\theta)| > \epsilon \big\} > \delta \Big\} = 0, \tag{2.26}$$

which is a convergence in probability–P^**, probability–*P*.*

Proof. First, consider that assumption (b.ii) implies the convergence of θ_l^* to θ in conditional probability P^*. Indeed:

$$P\Big\{ P^*\big\{ \|\theta_l^* - \theta\| > 2\epsilon \big\} > \delta \Big\} \leqslant P\Big\{ P^*\big\{ \|\theta_l^* - \theta_n\| + \|\theta_n - \theta\| > 2\epsilon \big\} > \delta \Big\},$$

since the event $\|\theta_l^* - \theta_n\| + \|\theta_n - \theta\| > \epsilon$ contains the event $\|\theta_l^* - \theta\| > \epsilon$ for any value of ϵ. Now, due to the convergence of θ_n to θ, for any subsequence n', we can find a further subsequence n'' where the convergence happens almost surely. Since this can be done for all sequences, consider directly the almost sure argument. Thus, we can make the term $\|\theta_n - \theta\|$ as small as we please, say not bigger than ϵ as n grows. Hence, if $n > \bar{n}$ for some integer $\bar{n}$:

$$\begin{aligned} P\Big\{ P^*\big\{ \|\theta_l^* - \theta\| > 2\epsilon \big\} > \delta \Big\} &\leqslant P\Big\{ P^*\big\{ \|\theta_l^* - \theta_n\| + \|\theta_n - \theta\| > 2\epsilon \big\} > \delta \Big\} \\ &= P\Big\{ P^*\big\{ \|\theta_l^* - \theta_n\| > 2\epsilon - \|\theta_n - \theta\| \big\} > \delta \Big\} \leqslant P\Big\{ P^*\big\{ \|\theta_l^* - \theta_n\| > \epsilon \big\} > \delta \Big\}. \end{aligned}$$

We note that the term $\|\theta_n - \theta\|$ is constant when considering the probability P^*. Moreover, the last inequality is justified by the fact that the set $\|\theta_l^* - \theta_n\| > \epsilon$ contains the set $\|\theta_l^* - \theta_n\| > \epsilon'$ if $\epsilon < \epsilon'$. However, the last term of the inequality above goes to 0 by assumption if $n, l \to \infty$. So we have that, for any $\epsilon > 0$:

$$\lim_{n,l\to\infty} P\Big\{ P^*\big\{ \|\theta_l^* - \theta\| > \epsilon \big\} > \delta \Big\} = 0 \tag{2.27}$$

The remaining part of the proof is basically the same as in *theorem 2.6.1* of Bierens (1996).

Consider the following chain of inequalities:

$$P\Big\{P^*\big\{|S_n(\mathbb{X}_n,\theta_l^*)-S(\theta)|>\epsilon\big\}>\delta\Big\}\leqslant$$
$$P\Big\{P^*\big\{|S_n(\mathbb{X}_n,\theta_l^*)-S(\theta_l^*)|+|S(\theta_l^*)-S(\theta)|>\epsilon\big\}>\delta\Big\}\leqslant$$
$$P\Big\{P^*\big\{\sup_{t\in\Theta}|S_n(\mathbb{X}_n,t)-S(t)|+|S(\theta_l^*)-S(\theta)|>\epsilon\big\}>\delta\Big\}\quad w.p.1.$$

The last inequality holds with probability 1 due to assumption (b.i), which ensures θ, θ_n and θ_l^* are in Θ with probability 1.

Consider now any subsequence of (n), n'. For this sequence there is a subsequence, say n_1'' such that assumption (b.iii) holds almost surely. This implies that, there is an integer $\bar{n}_1'$, such that for any $\epsilon'>0$, $\sup_{t\in\Theta}|S_n(\mathbb{X}_n,t)-S(t)|\leqslant\epsilon'$ if $n>\bar{n}_1'$ but for a null set N_2. For the sequence n_1', it is possible to find a further subsequence, n_2'', such that the convergence in (2.27) holds in probability–P^*, almost surely–P. That is, there is a null set N_3 such that for $\omega\in\Omega\backslash N_3$, any $\epsilon>0$: $\lim_{n,l\to\infty}P^*(\|\theta_l^*-\theta\|>\epsilon)\to 0$.

Now consider the set $N_1=\big\{\bigcup_l\bigcup_n N_{1,l,n}^*\big\}\bigcup\big\{\bigcup_n N_{1,n}'\big\}\bigcup N_1''$, where $N_{1,l,n}^*$, $N_{1,n}'$ and N_1'' are the null sets where assumption (b.i) fails to hold for θ_l^*, θ_n and θ respectively. Being the countable union of null sets, $N_1\in\mathcal{F}$ and is null. Let N_4 be the null set where $S(\cdot)$ fails to be continuous, and let $N:=N_1\bigcup N_2\bigcup N_3\bigcup N_4$; being the countable union of null sets in $\mathcal{F}$, N is a null sets, $N\in\mathcal{F}$ and $P(\Omega\backslash N)=1$. For $\omega\in\Omega\backslash N$, and $n,l>\bar{n}'$, we have:

$$P^*\big\{\sup_{t\in\Theta}|S_n(\mathbb{X}_n,t)-S(t)|+|S(\theta_l^*)-S(\theta)|>\epsilon\big\}=$$
$$P^*\big\{|S(\theta_l^*)-S(\theta)|>\epsilon-\sup_{t\in\Theta}|S_n(\mathbb{X}_n,t)-S(t)|\big\}\leqslant$$
$$P^*\big\{|S(\theta_l^*)-S(\theta)|>\epsilon-\epsilon'\big\}.$$

By the continuity of S and the convergence of $\theta_l^*\to\theta$, as $n,l\to\infty$, the last term goes to 0. Thus, the term $S_n(X_n,\theta_l^*)\to S(\theta)$ in probability–P^*, almost surely–P for the sequence n'. Since this argument holds for any sequence n', this implies that the convergence $S_n(X_n,\theta_l^*)\to S(\theta)$ in probability–P^*, in probability–P or equivalently:

$$\lim_{n,l\to\infty}P\Big\{P^*\big\{|S_n(\mathbb{X}_n,\theta_l^*)-S(\theta)|>\epsilon\big\}>\delta\Big\}=0$$

□

Remark 2.3.2. *In general, the rate at which l goes to ∞ is determined as a function of n and depends by the particular result applied to show validity of assumption (b.ii). A typical choice is $l=n$.*

Remark 2.3.3. *Assumption (b.ii) can be replaced by any result stating the convergence of the*

bootstrapped quantity θ_l^*, as for example: for any $\epsilon > 0$, $\delta > 0$,

$$\lim_{n,l\to\infty} P\{P^*\{\|\theta_l^* - \theta\| > \epsilon\} > \delta\} = 0.$$

Now, we move to the second result, showing the convergence in distribution of the quantities (2.22). This convergence will be essential to justify the confidence intervals (2.23). The proof basically relies on the delta method applied to the root (2.22).

Proposition 2.3.2. *Let be assumptions of proposition 2.3.1 be satisfied and assume for convenience that $l = n$. Assume additionally that:*

(b.iv) (bootstrap estimator's convergence in distribution): for any $\epsilon > 0$:

$$\lim_{n\to\infty} P\left\{ \sup_t \left| P^*\{a_n(\theta_l^* - \theta_n) \leqslant t\} - P\{a_n(\theta_n - \theta) \leqslant t\} \right| > \epsilon \right\} = 0, \tag{2.28}$$

for some rate a_n, $a_n \to \infty$ as $n \to \infty$; call T the distribution to which $a_n(\theta_n - \theta)$ converges.

(b.v) (uniform convergence of the first derivative of S_n over θ): $S_n(\mathbb{X}_n, \theta)$ is twice differentiable over θ with uniformly converging first derivatives (over θ, in probability). That is, as $\theta_n \to \theta$ in probability, assume:

$$\nabla_\theta S_n(\mathbb{X}, \theta_n) \xrightarrow[n\to\infty]{p} \nabla_\theta S(\theta).$$

Then, as $n \to \infty$, in distribution–P^*, probability–P:

$$a_n\left(S_n(\mathbb{X}_n, \theta_l^{*(b)}) - \frac{1}{B}\sum_{b=1}^{B} S_n(\mathbb{X}_n, \theta_l^{*(b)}) \right) \xrightarrow{d} \nabla_\theta S(\theta) T + \nabla_\theta S(\theta)\, \mathbb{E}\, T, \tag{2.29}$$

or, if $\mathbb{E}\, T = 0$,

$$a_n\left(S_n(\mathbb{X}_n, \theta_l^{*(b)}) - \frac{1}{B}\sum_{b=1}^{B} S_n(\mathbb{X}_n, \theta_l^{*(b)}) \right) \xrightarrow{d} \nabla_\theta S(\theta) T. \tag{2.30}$$

Proof. The requirement in assumption (b.iv) is equivalent to the following (van der Vaart (1998)):

$$\lim_{n\to\infty} P\{a_n(\theta_n - \theta) \leqslant t\} = T(t)$$
$$\lim_{n\to\infty} P\left\{ \left| P^*\{a_n(\theta_l^* - \theta_n) \leqslant t\} - T(t) \right| > \epsilon \right\} = 0,$$

for all t and any $\epsilon > 0$, for some distribution T. That is $a_n(\theta_n - \theta)$ converges in distribution to T and this can be approximated by the bootstrap distribution of $a_n(\theta^* - \theta_n)$ that converges in distribution–P^*, probability–P to T.

We consider the almost sure–P argument. Here, by a subsequence argument, consider all the

convergence stated above to be almost sure–P. Then consider the following:

$$a_n\Big(S_n(\mathbb{X}_n,\theta_l^{*(b)})-\frac{1}{B}\sum_{b=1}^{B}S_n(\mathbb{X}_n,\theta_l^{*(b)})\Big)=$$
$$a_n\Big(S_n(\mathbb{X}_n,\theta_l^{*(b)})-\frac{1}{B}\sum_{b=1}^{B}S_n(\mathbb{X}_n,\theta_l^{*(b)})+S_n(\mathbb{X}_n,\theta_n)-S_n(\mathbb{X}_n,\theta_n)\Big)=$$
$$a_n\Big(S_n(\mathbb{X}_n,\theta_l^{*(b)})-S_n(\mathbb{X}_n,\theta_n)\Big)-\frac{1}{B}\sum_{b=1}^{B}a_n\Big(S_n(\mathbb{X}_n,\theta_l^{*(b)})-S_n(\mathbb{X}_n,\theta_n)\Big). \quad (2.31)$$

Consider now the expansion of $S_n(\mathbb{X}_n,\theta_l^*)$ around θ_n:

$$S_n(\mathbb{X}_n,\theta_l^*)=S_n(\mathbb{X}_n,\theta_n)+\nabla_\theta S_n(\mathbb{X}_n,\theta_n)^T(\theta_l^*-\theta_n)+$$
$$(\theta_l^*-\theta_n)^T\Delta_\theta S_n(\mathbb{X}_n,\theta_n)(\theta_l^*-\theta_n)+\ldots,$$

where Δ_θ indicates the matrix of second derivatives of S_n. Rearranging the terms and multiplying by a_n yields:

$$a_n\Big(S_n(\mathbb{X}_n,\theta_l^*)-S_n(\mathbb{X}_n,\theta_n)\Big)=\nabla_\theta S_n(\mathbb{X}_n,\theta_n)^T a_n(\theta_l^*-\theta_n)+o_{P*}(a_n),$$

where we indicated by o_{P*} a term that goes to 0 when $n\to\infty$, at a rate a_n in probability–P^*, probability–P (note that this follows from assumption (b.ii) of proposition 2.3.1). Then, as $n\to\infty$:

$$a_n\Big(S_n(\mathbb{X}_n,\theta_l^*)-S_n(\mathbb{X}_n,\theta_n)\Big)=$$
$$\nabla_\theta S_n(\mathbb{X}_n,\theta_n)^T a_n(\theta_l^*-\theta_n)+o_{P*}(a_n)\xrightarrow{d}\nabla_\theta S(\theta)T, \quad (2.32)$$

Note now that as $B\to\infty$, since the bootstrap scheme produces i.i.d. random variables over resamples $*(b)$, the last term in equation (2.31) goes to $\nabla_\theta S(\theta)\,\mathbb{E}\,T$. This is motivated by the strong law of large numbers for i.i.d. variables and (2.32). Thus, by an application of Slustky's theorem applied to the bootstrap probability P^* and again (2.32), as $B,n\to\infty$:

$$a_n\Big(S_n(\mathbb{X}_n,\theta_l^{*(b)})-\frac{1}{B}\sum_{b=1}^{B}S_n(\mathbb{X}_n,\theta_l^{*(b)})\Big)\xrightarrow{d}\nabla_\theta S(\theta)T+\nabla_\theta S(\theta)\,\mathbb{E}\,T=\nabla_\theta S(\theta)T,$$

in distribution–P^*, almost sure–P. The last equality holds only if $\mathbb{E}\,T=0$.

Finally we note that the above is an almost sure argument in P, assumed to hold for any subsequence n'' of a subsequence $n'\in\mathbb{N}$. Thus, for the stated assumptions, the actual convergence is in distribution–P^*, probability–P. □

Remark 2.3.4. *In the proposition above we used $l=n$ for convenience. The above can be probably extended to scenarios where $l=l(n)$ and $n\to\infty$ implies $l\to\infty$. It would be needed that the bootstrapped distribution converges to the same distribution as the sample statistic. We do not pursue this further here.*

Remark 2.3.5. *The last term in equation (2.31) gives an indication of the minimum magnitude of B, which should be high enough to drive down the variance of the term $a_n\Big(S_n(\mathbb{X}_n, \theta_l^{*(b)}) - S_n(\mathbb{X}_n, \theta_n)\Big) = BT_n^{*(b)}$. Note that asymptotically $BT_n^{*(b)}$ are i.i.d. random variables extracted from $\nabla_\theta S(\theta)T$; let us call them Z_b. Thus, assuming the convergence holds in distribution–P^*, almost sure–P (is the convergence is in probability–P, then use a subsequence argument), by Chebyshev's inequality we have:*

$$\lim_{n\to\infty} P^*\Big(|\frac{1}{B}\sum_{b=1}^{B} BT_n^*| \;>\; t\Big) \;=\; P\Big(|\frac{1}{B}\sum_{b=1}^{B} Z_b| \;>\; t\Big) \;=\; P\Big((\frac{1}{B}\sum_{b=1}^{B} Z_b)^2 \;>\; t^2\Big) \;\leqslant\; \frac{\mathbb{E}\,(Z_1)^2}{Bt^2}.$$

Thus, the relative magnitude of B with respect to the variance of Z determines the rate at which the convergence stated in proposition 2.3.2 is to be trust.

Remark 2.3.6. *Note that with the same set of assumptions, it is possible to show that (by expanding $S_n(\mathbb{X}_n, \theta_n)$ around θ):*

$$\lim_{n\to\infty} a_n\Big(S_n(\mathbb{X}_n, \theta_n) - S_n(\mathbb{X}_n, \theta)\Big) \to \nabla_\theta S(\theta)T, \tag{2.33}$$

in distribution–P. Note the extra bias term that is present in (2.29) is missing here, namely $\nabla_\theta S(\theta)\,\mathbb{E}\,T$. When the bias is 0, the bootstrapped quantity (2.29) and (2.33) converges to the same distribution.

To be precise, this result is yet not enough to prove the consistency of (2.22). In particular, note that the term a_n is determined by the hypothesis on the convergence of θ_n and θ_n^*. Were $a_n = \sqrt{n}$ and the required assumptions satisfied, then asymptotic distribution of (2.22) would immediately follow.

Finally, we establish the consistency of (2.21) and (2.23). In order to do so, we also need further conditions on the underlying F and on the scoring function s. These are needed to ensure sufficient smoothness of the scoring function. These issues are analyzed in details at the end of this section. To ease the presentation, we report here the additional assumptions. These pertain to the underlying distribution F and the parameter space Θ.

(s.i) $X \sim F$ *where* F *is such that* $\mathbb{E}\{(XX')^2\} < \infty$*;*

(s.i)* $X \sim F$ *is such that* $\mathbb{E}\{(XX')^4\} < \infty$*.*

(s.ii) $\pi_k > 0$ *for every* k *and* $\sum_{k=1}^{K} \pi_k = 1$*;*

(s.iii) $\|\mu_k\|_2 \leqslant M$ *for some large* M *for every* k*;*

(s.iv) Σ_k *is non singular for every* k*.*

We now show that we can consistently estimate $\mathbb{E}_\theta\, S(\theta)$ with the bootstrapped version. For the moment being, we are going to assume (b.i), (b.ii) and (s.i), (s.ii), (s.iii), (s.iv) hold; also assume $\mathbb{X}_n$ is a sequence of i.i.d. random variables from F. These assumptions ensure that $s(X, \theta)$ is a continuous function in both arguments; Θ is a compact set; $\mathbb{E}(\sup_{t\in\Theta} s(X, t)^2) < \infty$

(by propositions 2.3.6 or 2.3.7). Now, by a straightforward application of *theorem 2.7.5* – Bierens (1996), we have that for any $\epsilon > 0$:

$$P\Big\{ \lim_{n\to\infty} \sup_{t\in\Theta} \Big| \frac{1}{n} \sum_{i=1}^{n} s(X_i, t) - \mathbb{E}\, s(X, t) \Big| > \epsilon \Big\} \equiv$$
$$P\Big\{ \lim_{n\to\infty} \sup_{t\in\Theta} \Big| S_n(\mathbb{X}_n, t) - S(t) \Big| > \epsilon \Big\} = 0. \quad (2.34)$$

Result (2.34) states the almost sure uniform convergence of S_n, which is stronger than that required by assumption (b.iii). Now, with the assumptions above and (2.34), we can apply proposition 2.3.1, yielding the convergence in probability–P^*, probability–P for the considered S_n statistic: for any $\epsilon > 0$, $\delta > 0$, and any $b = 1, \ldots, B$:

$$\lim_{n\to\infty} P\Big\{ P^*\big\{ |S_n(\mathbb{X}_n, \theta_n^{*(b)}) - S(\theta)| > \epsilon \big\} \Big\} = 0.$$

Thus, applying the law of large numbers to the bootstrap probability–P^*, we have the following:

$$\lim_{B\to\infty} \lim_{n\to\infty} \frac{1}{B} \sum_{b=1}^{B} S_n^{*(b)} \to \mathbb{E}\, S(\theta) \equiv \mathbb{E}_\theta\, \mathbb{E}_X\, s(X, \theta), \quad (2.35)$$

where the convergence happens in probability–P^*, in probability–P. This result states that we can consistently estimate (2.20) with its bootstrap counterpart. Next we show the adequacy of the confidence interval.

The following discussion consider s to be specified as (2.12). Under the same set of assumptions, strengthened by (b.iv) and (s.i*) (which automatically implies (b.v) by proposition 2.3.8 similarly to (b.iii)), we can apply Proposition 2.3.2 together with *Lemma 23.3*–van der Vaart (1998), yielding the consistency of the bootstrapped quantiles for the distribution $\nabla_\theta S(\theta) T$ (eventually shifted by a bias term). We indicate these with t_α. More precisely, L_n^* (i.e. $L_n^*(m)$) and U_n^* (i.e. $U_n^*(m)$) as defined in Algorithm 2 converges to L and U, where:

$$L := \inf_t \Big\{ t : P(\nabla_\theta S(\theta) T \leqslant t) \geqslant \frac{\alpha}{2} \Big\} + \nabla_\theta S(\theta)\, \mathbb{E}\, T = t_{\frac{\alpha}{2}} + \nabla_\theta S(\theta)\, \mathbb{E}\, T;$$
$$U := \inf_t \Big\{ t : P(\nabla_\theta S(\theta) T \leqslant t) \geqslant 1 - \frac{\alpha}{2} \Big\} + \nabla_\theta S(\theta)\, \mathbb{E}\, T = t_{1-\frac{\alpha}{2}} + \nabla_\theta S(\theta)\, \mathbb{E}\, T;$$

the convergence occurs as $B \to \infty$, $n \to \infty$, in probability–P^*, in probability–P.

Now, for simplicity, we assume that the convergence stated in proposition 2.3.2 holds in distribution–P^*, almost sure–P; a subsequence argument allows to extend the following to the convergence in probability–P. Thus, along almost all sample sequences, that is, with probability–

P 1, as $B, n \to \infty$:

$$P^*\Big\{S_n(\mathbb{X}_n,\theta) \geqslant S_n^* - \frac{L_n^*}{a_n}\Big\} = P^*\Big\{a_n\Big(\frac{1}{B}\sum_{b=1}^{B} S_n^{*(b)} - S_n(\mathbb{X}_n,\theta)\Big) \leqslant L_n^*\Big\} =$$

$$P^*\Big\{a_n\Big(\frac{1}{B}\sum_{b=1}^{B} {S_n}^{*(b)} - S_n(\mathbb{X}_n,\theta_n)\Big) + a_n\Big(S_n(\mathbb{X}_n,\theta_n) - S_n(\mathbb{X}_n,\theta)\Big) \leqslant L_n^*\Big\} \xrightarrow{p^*}$$

$$P\Big\{\Big(\nabla_\theta S(\theta)\,\mathbb{E}\,T + \nabla_\theta S(\theta)T\Big) \leqslant \nabla_\theta S(\theta)\,\mathbb{E}\,T + t_{\frac{\alpha}{2}}\Big\} = P\Big\{\nabla_\theta S(\theta)T \leqslant t_{\frac{\alpha}{2}}\Big\} = \frac{\alpha}{2},$$

where we used $\xrightarrow{p^*}$ to indicate that the convergence is in probability–P^*, and is motivated by an application of Slutsky's theorem to the bootstrap probability P^*. The same reasoning applies to U_n^*, therefore $S_n(\mathbb{X}_n,\theta)$:

$$P^*\Big\{S_n^* - \frac{U_n^*}{a_n} \leqslant S_n(\mathbb{X}_n,\theta) \leqslant S_n^* - \frac{L_n^*}{a_n}\Big\} \to 1-\alpha,$$

where the convergence is in probability–P^*, probability–P (as $B, n \to \infty$). This result establishes (2.23). Note that by Remark 2.3.6, via a similar expansion, it can be shown that

$$P^*\Big\{S_n^* - \frac{U_n^*}{a_n} \leqslant S(\theta) \leqslant S_n^* - \frac{L_n^*}{a_n}\Big\} \to 1-\alpha,$$

where the convergence is in probability–P^*, probability–P (as $B, n \to \infty$).

Assumptions in Propositions 2.3.1 and 2.3.2 are rather general, some of them can be characterized in terms of easier to interpret conditions involving F, some other are not easy to be established for the clustering algorithm at hand. In the following Propositions (labelled from 2.3.3 to 2.3.8) we show that, under fairly easy to interpret assumptions on data distribution, all sufficient conditions for Propositions 2.3.1 and 2.3.2 are satisfied, except the conditions (b.ii) and (b.iv) about P^*. The latter are different and will be treated at the of the this section.

Properties of the scoring function $s(X,\theta)$

In this section we formally illustrate properties of the considered scoring function s. These were largely used above to show asymptotic properties of the resampling scheme.

Let us recollect here the quantities used in the following.

$$S_n\big(\mathbb{X}_n,\theta\big) := \frac{1}{n}\sum_{i=1}^{n} s(X_i,\theta); \qquad S\big(\theta\big) \;\; := \mathbb{E}_X\, s(X,\theta);$$

$$s(x;m) := \sum_{k=1}^{K(m)} \frac{\pi_k^{(m)}\phi_k(x;m)}{\sum_{k=1}^{K(m)}\pi_k^{(m)}\phi_k(x;m)} \log\big(\pi_k^{(m)}\phi_k(x;m)\big);$$

$$s_h(x;m) := \sum_{k=1}^{K(m)} \mathbb{1}\Big\{k = \arg\max_k\Big\{\pi_k^{(m)}\phi_k(x;m)\Big\}\Big\} \log\big(\pi_k^{(m)}\phi_k(x;m)\big);$$

$$\theta := \Big(\pi_1^{(m)},\dots,\pi_{K(m)}^{(m)},\mu_1^{(m)},\dots,\mu_{K(m)}^{(m)},\Sigma_1^{(m)},\dots,\Sigma_{K(m)}^{(m)}\Big).$$

where $K(m)$ is a finite integer determined by the method $m \in \mathcal{M}$ and $\phi(X; \mu_k, \Sigma_k)$ are (multivariate) normal distributions with centres given by μ_k and covariances given by Σ_k. In the following we will assume some $m \in \mathcal{M}$ is specified and consider a fixed $K = K(m)$. We drop the dependency on m.

Note that due to (2.17) this analysis is also valid for HS and SS scores based on the quadratic score (see (2.11) and (2.11)).

The following three propositions establish basic properties of the scoring functions s and s_h.

Proposition 2.3.3 (continuity)**.** *$s(x, \theta)$ and $s_h(x, \theta)$ as defined above are continuous in x, θ.*

Proof. $s(x, \theta)$ is obviously continuous in x, θ, being the product of continuous functions.

Consider $s_h(x, \theta)$. Discontinuity points might occur when the indicator function switches from one component to another. Without loss of generality, we treat the case where $K = 2$. Consider the following:

$$s_h(x, \theta) = \begin{cases} \log(\pi_1 \phi_1(x)) \text{ if } 1 = \arg\max_{1,2} \pi_i \phi_i(x) \\ \log(\pi_2 \phi_2(x)) \text{ if } 2 = \arg\max_{1,2} \pi_i \phi_i(x) \end{cases} = \max\{\log(\pi_1 \phi_1(x)), \log(\pi_2 \phi_2(x))\}$$

The maximum of continuous functions is continuous. □

Proposition 2.3.4 (Bounded from above)**.** *If $\pi_k \in (0, 1)$ and $\det \Sigma_k > 0$ for all $k = 1, \ldots, K$, $s(x, \theta)$, then $s_h(x, \theta)$ are bounded from above. I.e. $\exists M \in \mathbb{R} : s(x, \theta) < M$ for any $x \in \mathbb{R}^p$.*

Proof. We give the proof for s; s_h follows by the same argument, changing the smooth weight with the indicator variable. For a given number of K, $\pi_k \in (0, 1)$ $\forall k = 1 \ldots K$ (if one of the π_k is equal to 0 we are in a case with $K - 1$ components; if one of the π_k is equal to 1 we are in the case of $K = 1$). Moreover, $\phi_k(x)$ for finite μ_k and non-singular Σ_k is bounded by 0 from below and by $\phi_k(\mu_k)$ from above. As a consequence:

$$\frac{\pi_k \phi_k(x)}{\sum_{k=1}^K \pi_k \phi_k(x)} \in (0, 1); \quad \sum_{k=1}^K \frac{\pi_k \phi_k(x)}{\sum_{k=1}^K \pi_k \phi_k(x)} = 1 \tag{2.36}$$

Consider $\log(\pi_k \phi_k(x))$, this quantity belongs to the interval $(-\infty, \log(\pi_k \phi_k(\mu_k)))$. As a consequence, it is easy to see that:

$$\sum_{k=1}^K \frac{\pi_k \phi_k(x)}{\sum_{k=1}^K \pi_k \phi_k(x)} \log(\pi_k \phi_k(x)) \leqslant \sum_{k=1}^K \log(\pi_k \phi_k(x)) \leqslant \sum_{k=1}^K \log(\pi_k \phi_k(\mu_k)) \leqslant \infty \tag{2.37}$$

However, it is not bounded from below, as it may happen that as $\|x\| \to \infty$, $\frac{\pi_k \phi_k(x)}{\sum_{k=1}^K \pi_k \phi_k(x)} > 0$ and $\log(\pi_k \phi_k(x)) \to -\infty$. □

Proposition 2.3.5 (Bounded in probability)**.** *If $\pi_k \in (0, 1)$ and $\det \Sigma_k > 0$ for all $k = 1, \ldots, K$, $s(x, \theta)$, then $s(X, \theta)$, $s_h(X, \theta)$ are bounded in probability. I.e. for any $\epsilon > 0$, $\exists M \in \mathbb{R}$: $P\{|s(X, \theta)| < M\} \geqslant 1 - \epsilon$.*

Proof. We give the proof for s; s_h follows by the same argument, changing the smooth weight with the indicator variable. We need to show that: $\forall \epsilon > 0$, $\exists M \in \mathbb{R}$: $Pr\{|s(X, \theta)| < M\} > 1 - \epsilon$.

Consider the following (this argument holds for crisp weights as well):

$$|s(X,\theta)| \leqslant \sum_{k=1}^{K} |\log(\pi_k \phi_k(X))|. \tag{2.38}$$

We note that, for the (multivariate) normal distribution, $\forall \epsilon > 0$, $\exists 0 < a_\epsilon < b_\epsilon \in \mathbb{R} : Pr\{\phi_k(X) \in (a_\epsilon, b_\epsilon)\} > 1 - \epsilon$. Thus, since the logarithm is a continuous transformation, we have that, for some $M_\epsilon \in \mathbb{R} :\ 0 < e^{-M_\epsilon} \leqslant a_\epsilon < b_\epsilon \leqslant e^{M_\epsilon} < \infty$:

$$\begin{aligned} 1 - \epsilon < Pr\{\phi_k(X) \in (a_\epsilon, b_\epsilon)\} = Pr\{a_\epsilon < \phi_k(X) < b_\epsilon\} \leqslant Pr\{e^{-M_\epsilon} < \phi_k(X) < e^{M_\epsilon}\} \\ = Pr\{-M_\epsilon < \log(\phi_k(X)) < M_\epsilon\} = Pr\{|\log(\phi_k(X))| < M_\epsilon\}. \end{aligned} \tag{2.39}$$

Thus, $\log(\phi_k(X))$ is bounded in probability (we note that if all π_k are greater than 0, the argument to prove boundedness in probability is exactly the same, taking into account the added constant: $\log(\pi_k \phi_k(x)) = \{\log(\pi_k) + \log(\phi_k(x))\}$). Thus, as absolute sum of random variable bounded in probability, $\sum_{k=1}^{K} |\log(\pi_k \phi_k(X))|$ is bounded in probability and so is $s(X,\theta)$.

□

The following two propositions prove the existence of the second moment of s and s_h, with respect to random variable X, for all possible values of θ. These are crucial, in that they need to establish uniform convergence required by assumption (b.iii), on which Propositions 2.3.1 and 2.3.2 heavily rely. This is an essential regularity condition, and amounts to shape the degree of smoothness required for the scoring function used in the resampling scheme.

Proposition 2.3.6 (Existence of first two moments of s)**.** *If for every value t of θ*

(s.i) $X \sim F$ *where* F *is such that* $\mathbb{E}\{(XX')^2\} < \infty$*;*

(s.ii) $\pi_k > 0$ *for every* k *and* $\sum_{k=1}^{K} \pi_k = 1$*;*

(s.iii) $\|\mu_k\|_2 \leqslant M$ *for some large* M *for every* k*;*

(s.iv) Σ_k *is non singular for every* k *;*

then:

$$\mathbb{E} \sup_{\theta \in \Theta} \left(s(X,\theta)^2\right) < \infty \tag{2.40}$$

Proof. Consider a partition of $\mathbb{R}^p$, $\{A_k\}_{k=1\ldots K}$, where:

$$A_i := \{x \in \mathbb{R}^p : \log(\pi_i \phi_i(x)) \geqslant \log(\pi_k \phi_k(x))\ \forall k \neq i\}; \tag{2.41}$$

Note that due to the continuity of the functions involved, such a partition can always be found

for any value of θ. Then:

$$\mathbb{E}\left(s(X,\Theta)^2\right) = \int_{\mathbb{R}^p}\left(\sum_{k=1}^{K}\frac{\pi_k\phi_k(x)}{\sum_{k=1}^{K}\pi_k\phi_k(x)}\log(\pi_k\phi_k(x))\right)^2 dF(x) \leqslant$$
$$\int_{\mathbb{R}^p}\left(\sum_{k=1}^{K}\log(\pi_k\phi_k(x))\right)^2 dF(x) = \sum_{i=1}^{K}\int_{A_i}\left(\sum_{k=1}^{K}\log(\pi_k\phi_k(x))\right)^2 dF(x) \leqslant$$
$$\sum_{i=1}^{K}\int_{A_i}\left(K\log(\pi_i\phi_i(x))\right)^2 dF(x) \leqslant \sum_{i=1}^{K}\int_{\mathbb{R}^p}\left(K\log(\pi_i\phi_i(x))\right)^2 dF(x) =$$
$$\sum_{i=1}^{K}\int_{\mathbb{R}^p}K^2\log(\pi_i)^2 dF(x) + \sum_{i=1}^{K}\int_{\mathbb{R}^p}K^2\log(\phi_i(x))^2 dF(x) +$$
$$\sum_{i=1}^{K}\int_{\mathbb{R}^p}K^2 2\log(\pi_i)\log(\phi_i(x))dF(x). \quad (2.42)$$

Note that the inequality passing back from A_i to $\mathbb{R}^p$ is due to the positiveness of the integrand (which is a squared function). Having in mind these last three term, we are going to analyse them in turn.

Consider the first term. It is clearly finite if and only if we have a finite number of components K and $\pi_i > 0$ for each i (ensured by assumption (s.ii)).

$$K^2\sum_{k=1}^{K}\int_{A_k}C_k^2 dF(x) \leqslant K^2\sum_{k=1}^{K}\mathbb{E}\,C_k^2 = K^2\sum_{k=1}^{K}C_k^2 =: C < \infty \quad (2.43)$$

Consider the second term. We can rewrite it more explicitly as:

$$\sum_{i=1}^{K}\int_{\mathbb{R}^p}K^2\log(\phi_i(x))^2 dF(x) = K^2\sum_{i=1}^{K}\int_{\mathbb{R}^p}\left(C_i + \frac{-(x-\mu_i)'\Sigma_i^{-1}(-x-\mu_i)}{2}\right)^2 dF(x)$$
$$= K^2\sum_{i=1}^{K}C_i^2 + \frac{K^2}{4}\sum_{i=1}^{K}\mathbb{E}\left(X'\Sigma_i^{-1}X + \mu_i'\Sigma_i^{-1}\mu_i - 2\mu_i'\Sigma_i^{-1}X\right)^2 -$$
$$K^2\sum_{i=1}^{K}C_i\,\mathbb{E}\left(X'\Sigma_i^{-1}X + \mu_i'\Sigma_i^{-1}\mu_i - 2\mu_i'\Sigma_i^{-1}X\right). \quad (2.44)$$

where: $C_i = \log(2\pi^{-d/2}|\Sigma_i|^{-1/2})$. Consider now the second term of the expansion above. Since all the terms involving parameters only are finite due to assumptions (s.iii) and (s.iv), the remaining difficulties involve terms where X appears. Now, the term $\mathbb{E}(X'\Sigma_i^{-1}X)^2$ involves computing the expected value of linear combinations of the components of X up to the fourth power. Let p be the dimension of X. Define a set of integer vectors $I := \{(i_1, i_2, \ldots, i_p) : 0 \leqslant i_j \leqslant 4, \sum_{j=1}^{p} i_j = 4\}$. Let be I^* the set of unique elements of I, and let i^* denote elements in I^*. Based on some algebra, we can arrange the previous term as follows:

$$(X'\Sigma_i^{-1}X)^2 = \sum_{i^*\in I^*}\gamma_{i^*}X_1^{i_1}X_2^{i_2}\ldots X_d^{i_p}, \quad (2.45)$$

where γ_{i^*} is a coefficient depending on i^*. Note that in (2.45) at most four distinct component

of X can be present in each summand. Thus, to have this quantity bounded we need that:

$$\mathbb{E}\, X_i X_j X_l X_m < \infty; \quad \forall i, j, l, m = 1, \dots, p. \tag{2.46}$$

Assuming this condition holds, we can proceed with the rest of the proof. Going back to the second term of (2.44), consider the terms in the brackets. The first term is finite for the same argument as above. The second term, does not depend on X, and the assumptions on Σ_k and μ_k ensure this term is finite as well. The third term requires boundedness of $\mathbb{E}(XX')$ to be finite. However, the existence of these moments is already implied by (2.46). Thus the second term in (2.44) is bounded because of (s.i).

For the third term in (2.44), the reasoning is exactly as above, since the moments conditions required are less restrictive than (and thus implied by) (2.46).

Finally, consider the superior over θ. Since the integrand function is continuous in Θ, and Θ is compact, the superior is equal to the maximum of the integrand function in Θ. Call θ_0 the maximizer:

$$\mathbb{E}\left(\sup_{\theta\in\Theta} s(X,\theta)^2\right) = \mathbb{E}\left(s(X,\theta_0)^2\right) < \infty, \tag{2.47}$$

since the argument above applies to any $\theta \in \Theta$. □

Remark 2.3.7. *Note that the above proof can be adapted to the existence of the q-th moment of s. In this case, the set I used in the proof should be modified as $I := \{(i_1, i_2, \dots, i_p) : 0 \leqslant i_j \leqslant 2q, \sum_{j=1}^p i_j = 2q\}$, and adapt accordingly the requirements in (2.46).*

Proposition 2.3.7 (Existence of first two moments of s_h)**.** *Propositions 2.3.6 carries over on s_h, that is replacing smooth weight with indicator variables.*

Proof. Consider a partition of $\mathbb{R}^p$, $\{A_k\}_{k=1\dots K}$, where:

$$A_k := \{x \in \mathbb{R}^p : \log(\pi_k \phi_k(x)) \geqslant \log(\pi_i \phi_i(x)) \ \forall i \neq k\} \tag{2.48}$$

Then,

$$\begin{aligned}\mathbb{E}(s_h(X,t)^2) = \sum_{k=1}^K \int_{A_k} \log(\pi_k\phi_k(x))^2 dF(x) \leqslant \\ \sum_{k=1}^K \int_{A_k} \log(\phi_k(x))^2 dF(x) = \sum_{k=1}^K \int_{A_k} \left(C_k - \frac{\|x-\mu_k\|^2}{2}\right)^2 dF(x) = \\ \sum_{k=1}^K \int_{A_k} C_k^2 dF(x) - 2\sum_{k=1}^K \int_{A_k} \frac{C_k\|x-\mu_k\|^2}{2} dF(x) + \frac{1}{4}\sum_{k=1}^K \int_{A_k} \|x-\mu_k\|^4 dF(x)\end{aligned} \tag{2.49}$$

where $C_k = -\frac{p}{2}\log(2\pi) - \frac{1}{2}\log(|\Sigma_k|)$.
The rest of the proof is identical to that of proposition 2.3.6. □

The next proposition is the analogous to the above two to show the validity of assumption (b.v). This is needed to ensure regularity conditions once delta method is applied in proposition 2.3.2. This is shown for smooth scoring s only.

Proposition 2.3.8 (Existence of $\nabla_\theta S(\theta)$ second moment)**.** *Let assumptions of proposition 2.3.6 hold. Let assumption (s.i) be strengthened by the following:*

(s.i) $X \sim F$ is such that $\mathbb{E}\{(XX')^4\} < \infty$.*

Then:

$$\mathbb{E}\left(\sup_{t\in\Theta} \|\nabla_\theta s(X,t)\|^2\right) < \infty. \tag{2.50}$$

Proof. Consider the typical components of $\nabla_\theta s(x,t)$:

$$s(x,t) := \sum_{k=1}^{K} \frac{\pi_k\phi_k(x)}{f(x,\theta)} \log(\pi_k\phi_k(x)); \quad \text{call } f(x,\theta) \sum_{k=1}^{K} \pi_k\phi_k(x);$$

$$\frac{\partial}{\partial\pi_k} s(x,t) = \frac{\phi_k(x)}{f(x,\theta)}\Big(\log(\pi_k\phi_k(x)) + 1 - s(x,\theta)\Big); \tag{2.51}$$

$$\frac{\partial}{\partial\mu_k} s(x,t) = \frac{\pi_k\phi_k(x)}{f(x,\theta)}\Big(\log(\pi_k\phi_k(x)) + 1 - s(x,\theta)\Big)\Sigma_k^{-1}(x-\mu_k); \tag{2.52}$$

$$\frac{\partial}{\partial\sigma_{k,i}} s(x,t) = \frac{\pi_k\phi_k(x)}{f(x,\theta)}\Big(\log(\pi_k\phi_k(x)) + 1 - s(x,\theta)\Big)\Big(\frac{1}{\sigma_{k,i}} + \frac{(x_i-\mu_{k,i})^2}{\sigma_{k,i}^2}\Big), \tag{2.53}$$

where, for simplicity, we show the case for diagonal covariance matrices. That is Σ_k, are diagonal, and $\sigma_{k,i}$ indicates the i-th diagonal term in the k-th covariance matrix. This is without loss of generality, since the result can also be shown in the more general case of positive definite variance matrices. In fact, the expansion of the derivative in this latter case includes at most quadratic terms in x and the argument used later does not change. However, the algebra is much more involved, so that it is not easy to visualize the results. Note that the above equations are of the form:

$$\frac{\pi_k\phi_k(x)}{f(x,\theta)}\Big(\log(\pi_k\phi_k(x)) + 1 - s(x,\theta)\Big)g(x,\theta), \tag{2.54}$$

where $g(x,\theta) = \frac{1}{\pi_k}$ for equation (2.51); $g(x,\theta) = \Sigma_k^{-1}(x-\mu_k)$ for (2.52); $g(x,\theta) = \Big(\frac{1}{\sigma_{k,i}} + \frac{(x_i-\mu_{k,i})^2}{\sigma_{k,i}^2}\Big)$ for (2.53). Now it is easy to see that:

$$\mathbb{E}\left(\frac{\pi_k\phi_k(X)}{f(X,\theta)}\Big(\log(\pi_k\phi_k(X)) + 1 - s(X,\theta)\Big)g(X,\theta)\right)^2 \tag{2.55}$$

is bounded if:

$$\mathbb{E}\left(\frac{\pi_k\phi_k(X)}{f(X,\theta)}\Big(-s(X,\theta)\Big)g(X,\theta)\right)^2 < \infty, \tag{2.56}$$

since the boundedness of this term automatically implies the boundedness of all the other integrals arising from the expansion of the square in (2.55) (because this term contains the others). In turn, the boundedness of the term:

$$\mathbb{E}\left(\frac{\pi_k\phi_k(X)}{f(X,\theta)}\Big(-s(X,\theta)\Big)\Big(\frac{(X_i-\mu_{k,i})^2}{\sigma_{k,i}^2}\Big)\right)^2$$

will imply the boundedness of (2.56), because it involves higher order term with respect to X than the others. However, noting that the term $g(X,\theta)$ is always continuous in X, by using the

same line of proof in proposition 2.3.6, we have:

$$\mathbb{E}\left(\frac{\pi_k\phi_k(X)}{f(X,\theta)}\Big(-s(X,\theta)\Big)\Big(\frac{(X_i-\mu_{k,i})^2}{\sigma^2_{k,i}}\Big)\right)^2 \leqslant \mathbb{E}\left(\Big(s(X,\theta)\Big)^2\Big(\frac{(X_i-\mu_{k,i})^2}{\sigma^2_{k,i}}\Big)^2\right) \leqslant$$
$$\sum_{j=1}^{K}\int_{\mathbb{R}^p} K^2\log(\pi_j)^2\Big(\frac{(X_i-\mu_{k,i})^4}{\sigma^4_{k,i}}\Big)dF(x)+$$
$$\sum_{j=1}^{K}\int_{\mathbb{R}^p} K^2\log(\phi_j(x))^2\Big(\frac{(X_i-\mu_{k,i})^4}{\sigma^4_{k,i}}\Big)dF(x)+$$
$$\sum_{j=1}^{K}\int_{\mathbb{R}^p} K^2 2\log(\pi_j)\log(\phi_j(x))\Big(\frac{(X_i-\mu_{k,i})^4}{\sigma^4_{k,i}}\Big)dF(x),$$

where the last inequality is motivated by using the sets A_i as in proposition 2.3.6 (note that here the subscript is changed from i to j, since we are using the subscript $\{k,i\}$ to denote the derivative with the respect to the i-th term of the k-th covariance matrix; note also that the term $\frac{(X-\mu_{k,i})^4}{\sigma^4_{k,i}}$ is multiplied for all the terms in $s(x,\theta)$). Using the same argument of Proposition 2.3.6, the term that involves the higher moments in term of X is:

$$\sum_{j=1}^{K}\int_{\mathbb{R}^p} K^2\log(\phi_j(x))^2\Big(\frac{(X-\mu_{k,i})^4}{\sigma^4_{k,i}}\Big)dF(x).$$

It can be seen from (2.44) that this requires the finiteness of the following expectations:

$$\mathbb{E}\left((X_hX_jX_lX_m)\right)<\infty;\quad \forall h,j,l,m=1,\dots,p,$$

p being the dimension of X. Since this has to happen for any $i=1,\dots,p$, this condition can be ensured by $\mathbb{E}(XX')^4<\infty$. Now, this condition implies that all the terms appearing in $\|\nabla_\theta s(X,\theta)\|^2$ have a finite expectation. Similarly, because these terms are continuous function over Θ compact, the suprema over Θ are actually maxima, and by the assumptions on Θ (see proposition 2.3.6) the expectations of the terms in $\|\sup_{\theta\in\Theta}\nabla_\theta s(X,\theta)\|^2$ are finite. Then by linearity of the integral it follows:

$$\mathbb{E}(\sup_{\theta\in\Theta}\|\nabla_\theta s(x,\Theta)\|^2)\leqslant\mathbb{E}(\|\sup_{\theta\in\Theta}\nabla_\theta s(x,\Theta)\|^2)<\infty.$$

□

In Propositions 2.3.3 to 2.3.8 we have shown, under interpretable conditions on the data generating process, that some of the assumptions required for the consistency of the resampling procedure are fulfilled. These conditions essentially involve the existence of moments of the observable X, and certain requirements for the set Θ containing the solutions $\theta(m)$. More or less we require that the clustering method under study does not output singular clusters, and that none of the $K(m)$ cluster is returned empty. Although central in the proof of Propositions 2.3.1 and 2.3.2, no further insight is possible for conditions (b.ii) and (b.iv) involving P^*. These conditions essentially requires that the behaviour of the algorithm that computes $\hat{\theta}^*(m)$ (the bootstrap version of $\hat{\theta}_n(m)$, is nice enough that its bootstrap probability measure is able to

mimic the true underlying one. Andrews, 2002 shows that, when θ^* is an argmax functional (e.g. an MLE estimator), and the F has density which is sufficiently smooth beyond the second order, then these conditions are satisfied. However, these sort of sufficient conditions involving high order derivatives of the density of the generating model are difficult to check, perhaps except for simple cases that are of limited interest in clustering analysis.

Even if it is interesting to find sufficient conditions for 2.3.1 and 2.3.2 in such cases, this would be still unsatisfactory because usually the functional that maps X's into cluster solutions $\theta(m)$ in practice may be too difficult to frame in terms of well understood mathematical object. To give a toy example suppose that we want to cluster points using the k-means. It has been argued that we can map the k-means output into a well defined $\theta(m)$. Note that, the k-mean problem can be framed in terms of a ML problem, and the corresponding $\theta(m)$ would coincide with an MLE that is an argmax functional of F (Pollard et al., 1981). Now even if one is able to establish (b.ii) and (b.iv) using results as in Andrews, 2001 or Andrews, 2002, won't be a too reassuring guarantee indeed. In fact, it is well known that the k-mean optimization problem is NP-hard, in practice we approximate its solution using an heuristic algorithm, such as the popular Lloyd's algorithm. The latter only guarantees convergence, but whether the optimal k-mean solution is found strongly depends on the initialization, which adds a further level of randomness to $\theta(m)$. Therefore, what we would compute along the bootstrap resamples may not be even close to the object defined by the target functional.

The main message here is that perhaps a sensible practical thing to do is to gain insights on the bootstrap behaviour of $\theta(m)$ by running some experiment. For example, O'Hagan et al., 2018 show empirically that bootstrapping the EM-algorithm approximation of the MLE of Gaussian mixtures parameters produced results that confirm that the bootstrap distribution of the EM solutions is well behaved, although there is no theoretical guarantee for it. Again, in practice (b.ii) and (b.iv) require that the bootstrap distribution of the clustering solutions provided by the method $m \in \mathcal{M}$ is nice enough to be close to its true counterpart. For many practical clustering methods' implementations, we can only hope for this.

2.4 Empirical Analysis

In this section we show numerical experiments that include both real and artificial data, this allow us to test our methodology on a range of sample size, dimensionality and classes overlap. This section so organized: Subsection 2.4.1 presents the real and artificial datasets used for the analysis; Subsection 2.4.2 discusses the different criteria for selecting a clustering solution we put under comparison; Subsection 2.4.3 illustrates the construction of the set $\mathcal{M}$ of clustering methods we use to obtain a clustering solution; 2.4.4 concludes the section showing discussing the empirical results.

2.4.1 Datasets and sampling designs

Dataset are both real and simulated. Figures are collected at the end of this section for convenience.

We note that the sample size is set reasonably small for all data sets relatively to the dimen-

sionality of the data. In fact, sample sizes for the following data sets range from $n = 150$ to $n = 600$, where the dimensionality ranges from $p = 2$ to $p = 10$. This is to put the resampling algorithm under stress.

Subsection 2.3.1 shows a number of statistical properties for the algorithm, however those are asymptotic-type statements, meaning that they are guaranteed to hold in large samples. The numerical experiments here are built to assess the performance in the more realistic case of finite sample size. Moreover, taking bootstrap resamples of the original data for moderate n increases the probability of ties in the resamples. Algorithm 2 uses bootstrap resamples to compute a cluster solution obtained from a given method m a number of times, and this may be problematic here for various reasons: first in the presence of small clusters, these may be under-represented in the resamples; secondly, ties in the resamples increases the probability that spurious solutions are found by the clustering algorithms considered in the following study. These issues have been discussed in O'Hagan et al., 2018. Designing experiments where the ratio n/p is reasonably small, can be considered as a robust check for all the aforementioned issues.

Iris Dataset

The Iris dataset is a famous one. The data was collected by Anderson (1936). Fisher (1936) used it for the first time in a classification problem, in the paper introducing *Linear Discriminant Analysis*. Since then, the dataset has been used in a countless number of papers, to analyse both classification and clustering algorithms, starting a long tradition.

The Iris data collects measurements on three different Iris species, namely *Iris virginica, Iris versicolor and Iris setosa*. Each of the three classes counts 50 observations, for a total of 150. The features measured are the sepal length, sepal width, petal length and petal width. Thus, here $n = 150$ and $p = 4$. Figure 2.5 gives a visual representation of the data along two-dimensional slices of the four dimensions. It is possible to note that two of the classes have a substantial overlap.[3] The classes' overlap makes it difficult for clustering algorithms to split the data in the true groups.

Olive Dataset

The Olive dataset collects data on fatty acids of selected samples of Italian virgin olive oils. This dataset was introduced by Forina and Tiscornia (1982) and Forina et al. (1983), who used the percentage composition of the eight acids to predict the origin region of the oils. [4]

The Olive data is made of $n = 572$ observations, measured on $p = 8$ features each. The grouping of the data is twofold: a coarser one, where oils are classified according to macro regions of origin (Southern Italy, Sardinia and Northern Italy); a finer one, where the macro regions are split in several areas (North Apulia, South Apulia, Calabria, Sicily, Inland Sardinia, Costal Sardinia, East Liguria, West Liguria and Umbria). As discussed in Forina et al. (1983),

[3] This similarity between the two classes is motivated by the interesting theory by Anderson (1936), stating that Iris versicolor is a hybrid between the Iris setosa and the Iris virginica, receiving its genetics for one third by the former and for two thirds by the latter.

[4] The authors mention a previous work where they had previously described the data (Forina and Amarino, 1982). The dataset is made available within the *GGobi* software (Swayne, Lang, and Lawrence, 2006) and accessed through the *R* software via the *pgmm* package (McNicholas et al., 2011).

some of these areas are well separated, namely West Liguria, Costal Sardinia and Inland Sardinia, while some others show a large similarity, like Sicily, Calabria and South Apulia. The data is represented via its principal components in Figure 2.6. Also, Figure 2.7 shows bidimensional plots for some selected acid variables with the finer group classification. It can easily be seen that data comes with an high amount of discreteness. Also, for some areas, the scatters are highly concentrated along some hyperplanes. These aspects of the Olive data typically cause many algorithms to struggle in retrieving the true 9 areas, estimating poor solutions (compare also with the discussion on complexity in Subsection 2.1.1).

Banknote Dataset

Banknote data is made of $n = 200$ observations on both *genuine* and *counterfeit* Swiss 1000-franc banknotes. The notes recorded in the dataset are those from the second banknote series, first issued between 1911 and 1914 and recalled in 1958. The dataset was introduced in Flury and Riedwyl (1988) and was collected to analyse statistical techniques to tell genuine and counterfeit notes apart.

In Flury and Riedwyl (1988), only six measures on one side of the bill were reported, namely: length of the bill (*length*); width measured on the left side (*left*); width measured on the right (*right*); width of margin at the bottom (*bottom*) width of margin at the top (*top*); image diagonal length (*diagonal*). Measurements are in millimetres (see figure 2.8). The data has observations evenly split across the two type of notes: 100 samples of genuine bills; 100 samples of counterfeit ones.[5] Overall, the dataset has $n = 200$ observation in $p = 6$ dimensions. Some of the features, like the bottom margin and the image diagonal, seem to be better discriminating the two classes. The two-by-two scatter-plots are represented in Figure 2.9.

Pentagon5

The pentagon5 data are simulated from a mixture of 5 spherical Gaussian components in a two-dimensional space. The Gaussian components are centred along the sides of an imaginary pentagon centred at the origin, hence the name of the sample design. Each component defines a cluster. The data generating process is represented by the following mixture model density,

$$f(x) = \sum_{k=1}^{5} \pi_k f_k(x); \quad f_k(x) = \phi(x; \mu_k; \Sigma_k),$$

where ϕ is a Gaussian density parametrized at μ and Σ. Components parameters are set at the following values:

$$\Sigma_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \forall k = 1 \dots 5,$$

[5] The sample size is limited because of the process of data acquisition, which was rather involved. Indeed, to magnify the small variability of the measured characteristics, the notes were projected on a wall and characteristics measured on the projections.

$$\pi_1 = 0.2, \quad \pi_2 = 0.35, \quad \pi_3 = 0.35, \quad \pi_4 = 0.05, \quad \pi_5 = 0.05,$$

$$\mu_1 = [0, 5], \quad \mu_2 = [-4.5, -0.5], \quad \mu_3 = [4.5, -0.5], \quad \mu_4 [3, -2.5], \quad \mu_5 = [-3, -2.5].$$

The total sample size is fixed at $n = 300$.

The peculiarity of this dataset arises from components' high unbalancedness, so that both solutions with $K = 3$ or $K = 5$ components may be considered reasonable. As Figure 2.10 shows, with a big enough sample, it should be relatively easy to identify 5 clusters for any methodology. However, when the sample size is small, it might be more sensible to identify only 3 clusters in the data: this is an example where the true density describes a non desirable clustering structure, which depends on the sample size. If groups' within homogeneity is pursued, here for small sample sizes it may be preferable to have 3 groups rather than 5.

t52D

This dataset is a 2-dimensional artificial design, obtained by sampling $n = 300$ points from a mixture of 5 t-student distributions (hence the name that reads t-student; 5 components; 2Dimensions). The model density is as follows:

$$f(x) = \sum_{k=1}^{5} \pi_k f_k(x); \quad f_k(x) = t\left(x; df_k, \mu_k, \frac{\Sigma_k(df_k - 2)}{df_k}\right), \tag{2.57}$$

where t is a multivariate t-student density with df degrees of freedom, location parameter μ, and scale parameter Σ (note that this is corrected so that the resulting covariance is Σ; e.g. see McNeil, Frey, and Embrechts, 2005). Components' parameters are set at the following values:

$$\pi_1 = 0.15, \quad \pi_2 = 0.04, \quad \pi_3 = 0.05, \quad \pi_4 = 0.15, \quad \pi_5 = 0.25,$$

$$df_1 = 10, \quad df_2 = 12, \quad df_3 = 14, \quad df_4 = 16, \quad df_5 = 18,$$

$$\mu_1 = [0, 3], \quad \mu_2 = [7, 1], \quad \mu_3 = [5, 9], \quad \mu_4 = [-11, 11], \quad \mu_5 = [-7, 5],$$

$$\Sigma_1 = \begin{bmatrix} 1 & 0.5 \\ 0. & 1 \end{bmatrix}, \quad \Sigma_2 = \begin{bmatrix} 2 & -1.5 \\ -1.5 & 2 \end{bmatrix}, \quad \Sigma_3 = \begin{bmatrix} 2 & 1.3 \\ 1.3 & 2 \end{bmatrix}, \quad \Sigma_4 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \quad \Sigma_5 = \begin{bmatrix} 2.5 & 0 \\ 0 & 2.5 \end{bmatrix}.$$

This design is composed of 5 clusters as shown in Figure 2.11. Differently from the previous design, there is a mix between rather spherical clusters and elongated ones. When there is few data, the difficulty arise because of the clusters' varying shape, which cause problems for solutions with fixed shape clusters. Some clustering solutions may prefer to join some clusters (solutions with highly elongated clusters) or split some others (solutions with spherical clusters); see also Figure 2.1 which represent some possible solutions for this design.

t510D

This is a slight modification of the t52D design. The t510D is a design in $p = 10$ dimensions. Here, along the first two dimensions, we have the same 5 t mixture distribution as in the previous design. However, on the remaining dimensions, we add 8 noisy features. The additional features are noisy in the sense that they increase the dimensionality of the t52D data without carrying additional group structure. In practice, these 8 noisy features have an unclustered spherical distribution.

Therefore, the model density representing the distribution of t510D, is the same as the density in (2.57), with the difference that now $p = 10$ and the remaining centrality and scatter parameters are set as spherical components. Thus, the density above is modified with respect to its parameters as follows:

$$\mu_k \leftarrow [\mu_{k,1}, \mu_{k,2}, 0, 0, 0, 0, 0, 0, 0, 0]\,; \quad \Sigma_k \leftarrow \begin{bmatrix} \Sigma_k & \mathbf{0} \\ \mathbf{0} & I_8 \end{bmatrix}, \qquad k = 1 \ldots, 5,$$

where I_8 is an eight-dimensional identity matrix.

This produces the design shown in Figure 2.12 via plotting pairs of dimensions. The first two dimension reproduce exactly the t52D design. The added noise, interact with these 2 dimension and produces varying results. On some pairs a different, still visible, clustering appears; this shows sometimes 4 separate components, some others may be regarded as 3 overlapping components. Considering only noisy pairs, these appears clearly unclustered. Overall, this design makes the appropriate clustering choice very unclear, even by human judgement. Several solutions appear reasonable when considering pairs of dimensions.

Finally, this design also uses $n = 300$ sampled points. The limited sample size, especially in this 10-dimensional dataset, makes the clustering task even more difficult.

Uniform

Finally, we include an example of unclustered data. This is obtained by sampling $n = 300$ points from a two dimensional uniform distribution. The sample is shown in Figure 2.4. Many clustering algorithms will still return a clustered solution even in such cases (see Hennig et al., 2015).

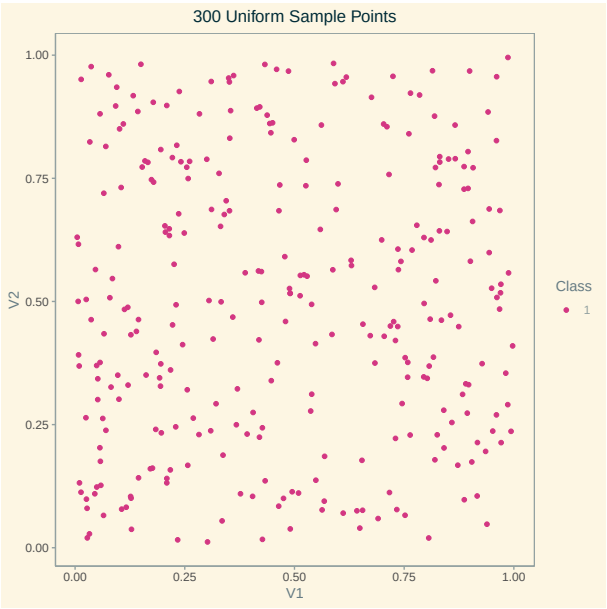


Figure 2.4: Sample from a two dimensional uniform distribution. Data is unclustered.

Table 2.1: Real and artificial designs used for experimental analysis. For each designs, size (n) and data dimensionality (p) is reported. (B) indicates the number of resamples used in applying Algorithm 2. Reasonable amount of groups supported by the data is indicated. Also, the number of Monte Carlo replications of the simulated datasets is reported.

Design	Real/Artificial	n	p	B	Reasonable number of groups K	Number of replicates
Iris	Real	150	4	1000	2 or 3	-
Olive	Real	572	8	1000	3 or 9	-
Banknote	Real	200	6	1000	2	-
Pentagon5	Artificial	300	2	100	3 or 5	10
t52D	Artificial	300	2	100	5	10
t510D	Artificial	300	10	100	4 or 5	10
Uniform	Artificial	300	1	100	1	10

Summary of the designs

Table 2.1 summarizes all the designs exposed so far. Note that the (B) column indicates the number of bootstrap resamples that we take, for the specific design, in applying the proposed Algorithm 2. These are required for resample-type criteria (see next, Subsection 2.4.2). Also, based on their discussion, for each of them we provide a reasonable amount of groups that we can expect the data to support.

We conclude this section with two final remarks.

Remark 2.4.1. *Some of the clustering methods considered for this numerical study (see the next section) are based on the assumption that the underlying clusters have at least approximately a Gaussian shape. Moreover, as highlighted in Section 2.2, the scoring approach proposed here has connections with the likelihood theory when the true underling class conditional distribution is Gaussian.*

The artificial data set considered here introduce some deviation from the normality to check whether the proposed method performs reasonably in situations where the groups have a non-Gaussian elliptical-symmetric distribution.

Remark 2.4.2. *Usually artificial sampling designs are introduced to perform Monte Carlo integration for computing average performance statistics with standard errors. In this study we only perform* 10 *replications the simulation designs. These seems an insufficient number of replicates to compute reliable averages and standard errors on selection criteria.*

Thus, in what follows, we will consider just the first realization for each artificial sample design. Datasets used in the experiment are exactly those shown in Figure 2.4 to Figure 2.12. The other 9 *replicates merely serve as preliminary assessment of the stability of the results. From these, we find that, qualitatively, the analysis given later in unmodified.*

The reason why we do not perform more than 10 *replicates for the simulated designs is because this would be* unfeasible *given the available hardware. This will be clear after the expositions of Subsection 2.4.2 and Subsection 2.4.3. We anticipate that, for resampling methods for each dataset, we fit the clusters at least B times for each of the considered alternative, that is $|\mathcal{M}| = 320$. This is, for simulated designs where $B = 100$, at least* 3200 *re-estimates. Overall,*

these are at least 32000 *re-estimates for just* 10 *replicates, when we would like to have at least* 100 *different replicates for an overall Monte Carlo analysis of the simulated design. This is computationally extremely intensive, and unfortunately poses serious problem to computing time and memory management for the hardware we have access to. Note that in practical situations* $|\mathcal{M}|$ *would be much smaller. Moreover, the resampling method proposed here easily adapts to the toy-parallelization technique to split each bootstrap re-fitting on a node-cpu. With the availability of modern computer clusters, we could have performed much more than* 100 *Monte Carlo replicates, however this was not possible.*

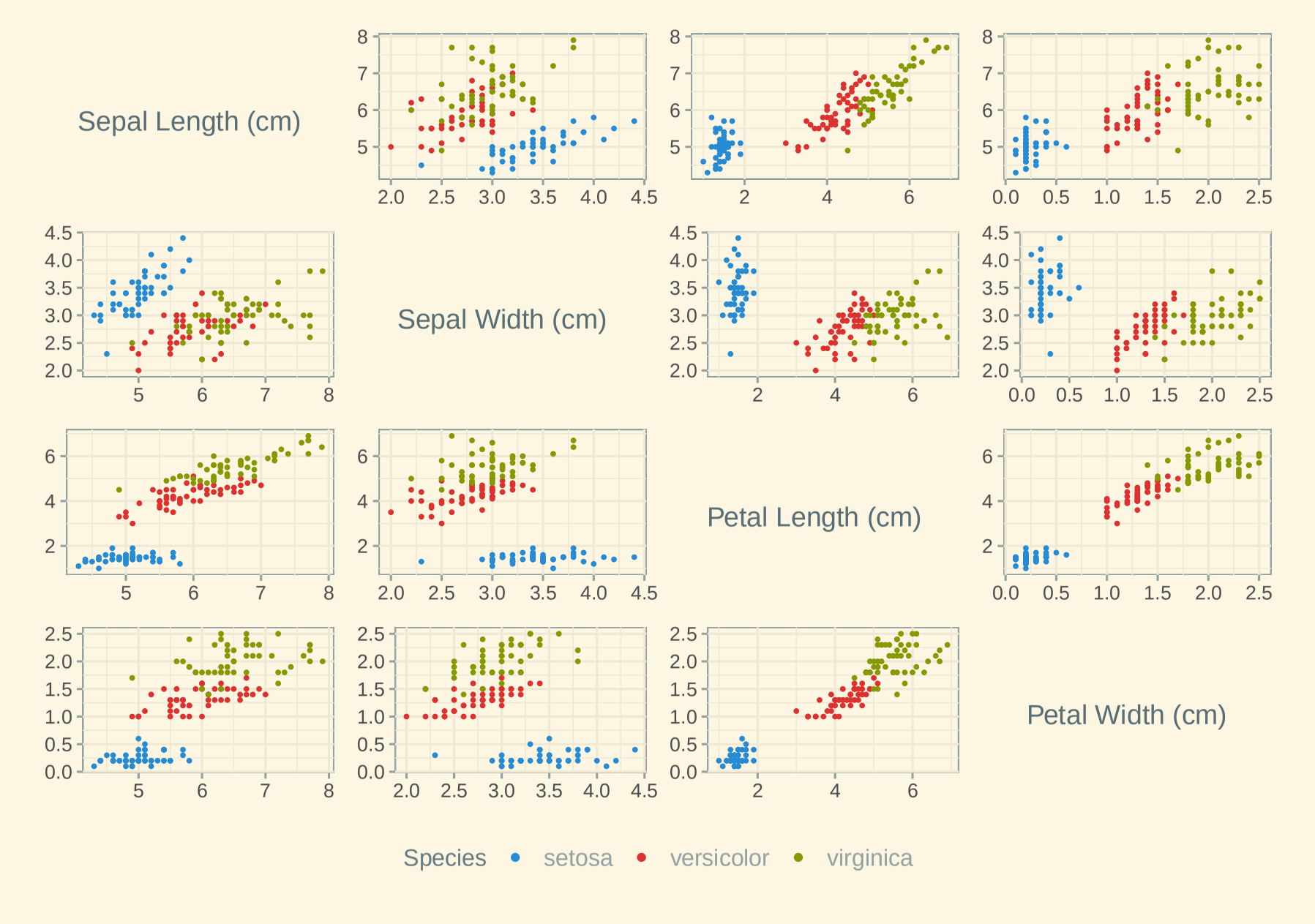


Figure 2.5: A representation of the Iris dataset along the four measured characteristics (names on the diagonal). The species are represented in different colours. It is possible to notice that two of the three classes, versicolor and virginica, exhibit a substantial overlap, which motivates the difficulty in identifying all the three groups using clustering algorithms.

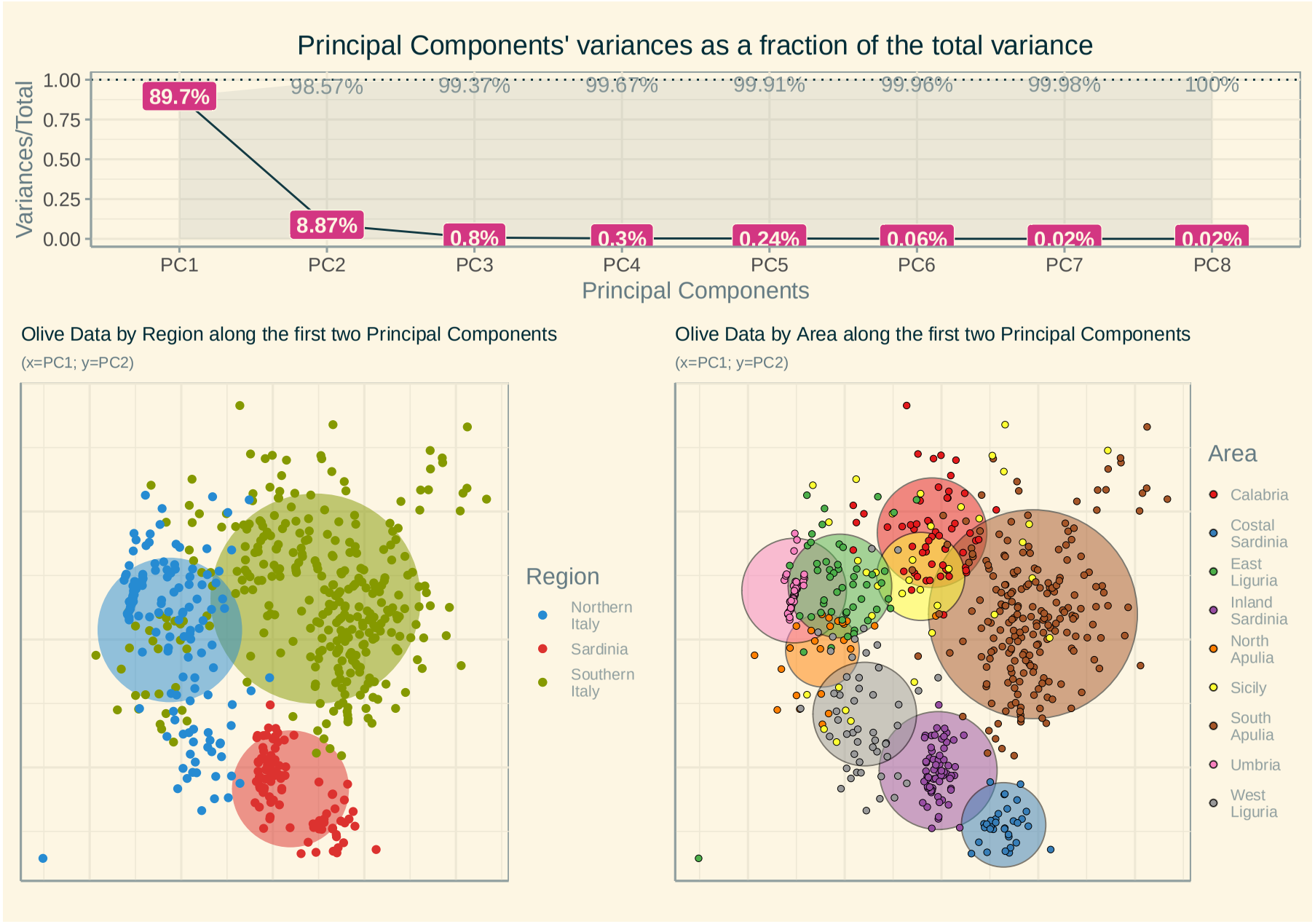


Figure 2.6: Top graph: principal components' explained fraction of total variance. The first two components amount for 98% of total data variability. Bottom-left: Olive data represented by origin region across the first two principal components. There is few overlap between regions. Bottom-right: Olive data represented by origin area across the first two principal components. The graph confirms that some of the areas are well separated (West Liguria, Costal Sardinia, Inland Sardinia) the others exhibits an higher amount of overlap. Circles (centred on class centres) represent the relative amount of classes in the sample.

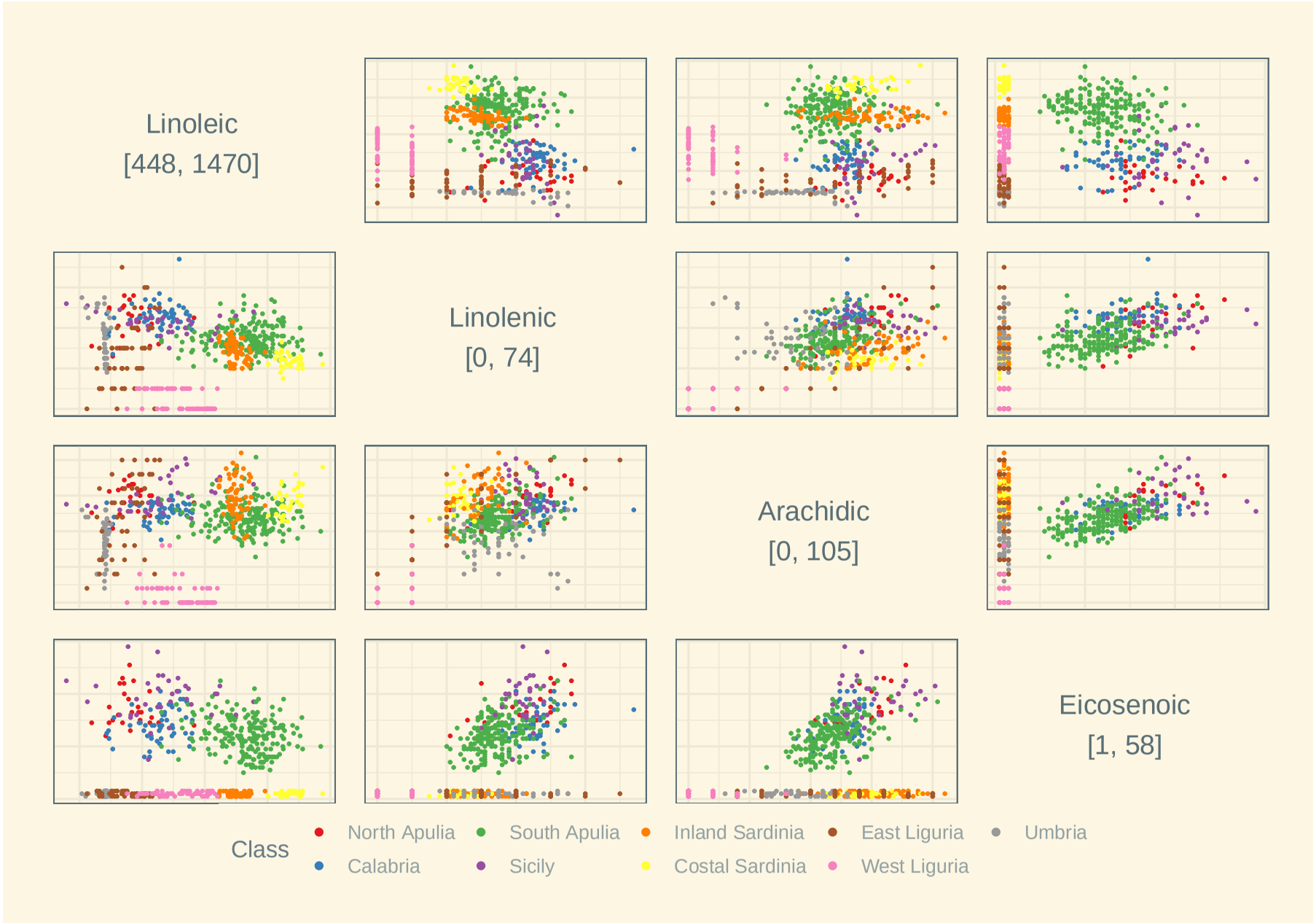


Figure 2.7: Pairs plot of the Olive dataset. Subset of acids variables along the diagonal (ranges in square brackets). From the figure it is easy to see that observations exhibit a high concentration for some values along the shown variables. Also, some areas (e.g. West Liguria, Umbria) present scatters concentrated on hyperplanes.

Figure 2.8: Swiss 1000-franc note, as those collected in the Banknote dataset. First issued in 1911. Data has measures on the back side only: 1) length; 2) left (width); 3) right (width); 4) bottom (margin width); 5) top (margin width); 6) diagonal (image length). Source for bill images: Swiss National Bank, 2019

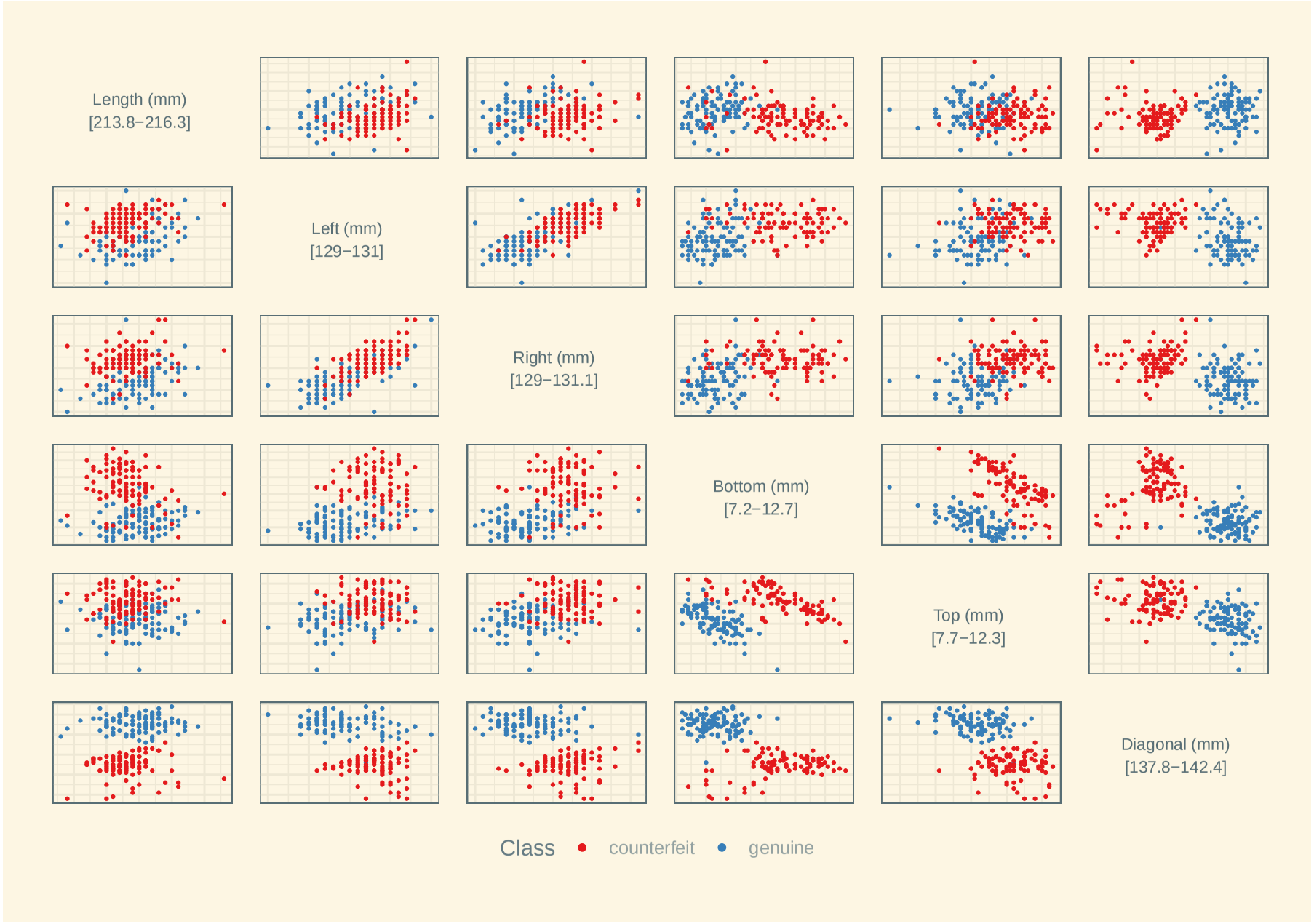


Figure 2.9: Two-by-two scatter-plots of the Swiss Banknotes data. The dimensions represented in each scatter-plot are specified on the main diagonal (range of the characteristic in square brackets). Some of the dimensions, like image diagonal length, seem to be able to distinguish the two classes well.

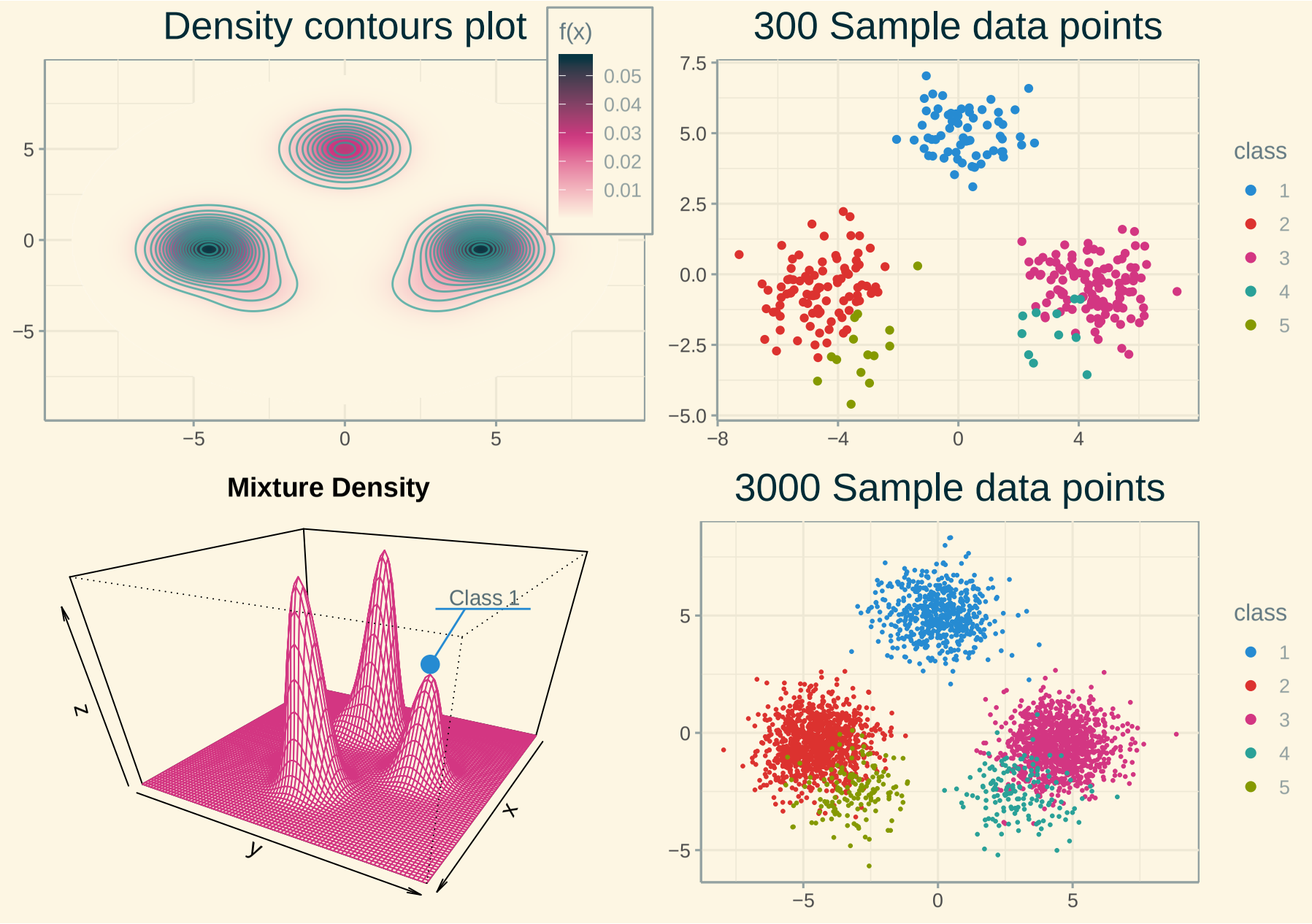


Figure 2.10: Top-left: contours plot of the mixture density $f(x)$. Darker areas indicate an higher value of the density. Lines represent the density levels; Bottom-left: a 3-dimensional representation of the top-left plot. The value of f is represented on the z-axis. Note that the plot is rotated to ease the visualization. The top mixture component is indicated by the blue dot; Top-right: 300 random sampled data points from f; Bottom-right: 3000 random sampled data points from f. The top mixture component is well separated from the others, while the other four exhibit a pairwise overlap. Overall, the components are well separated.

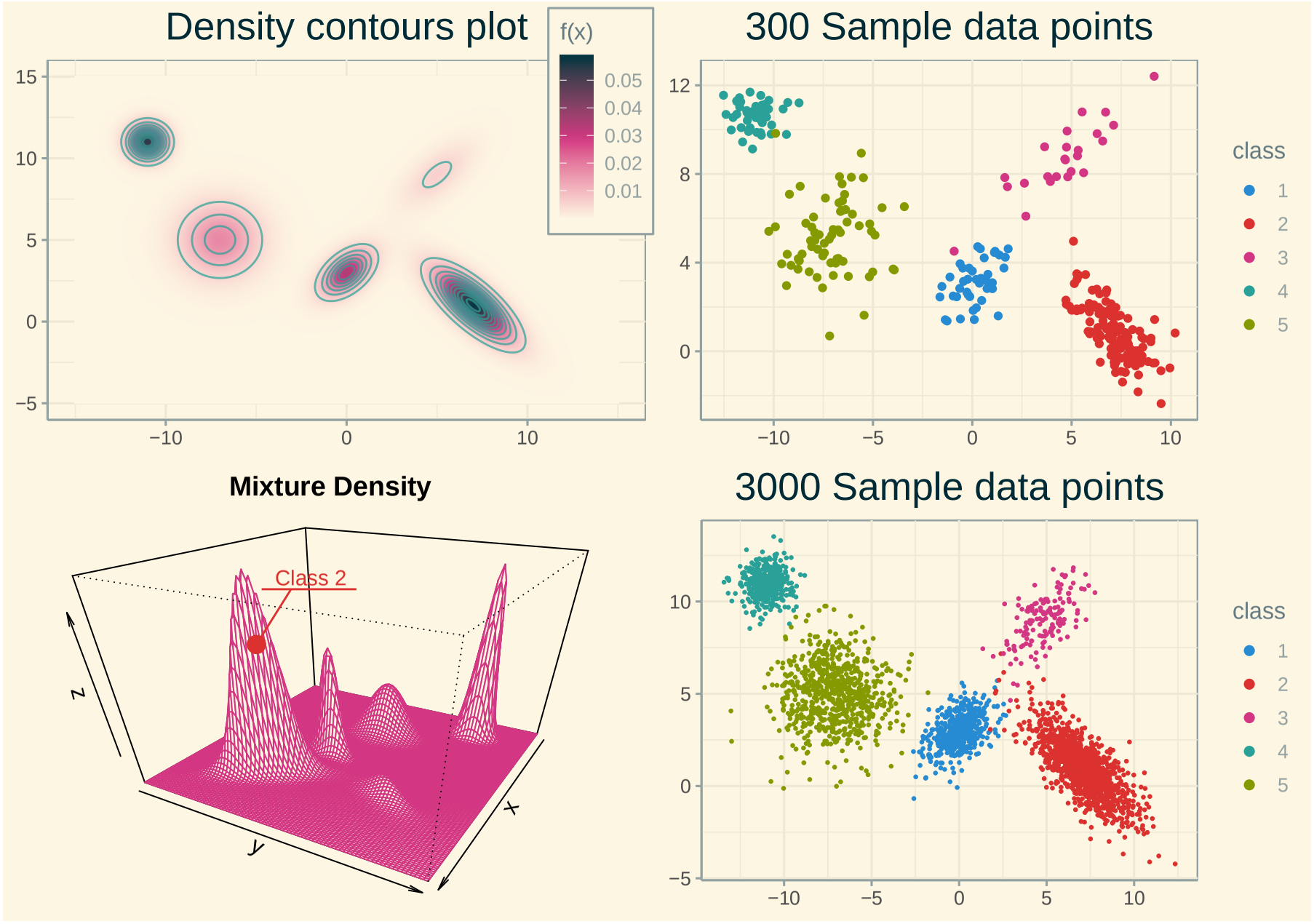


Figure 2.11: Top-left: contours plot of the mixture density $f(x)$. Darker areas indicate an higher value of the density. Lines represent the density levels; Bottom-left: a 3-dimensional representation of the top-left plot. The value of f is represented on the z-axis. Note that the plot is rotated to ease the visualization. The component closest to the bottom-right corner is represented by the red dot; Top-right: 300 random sampled data points from f; Bottom-right: 3000 random sampled data points from f. The two leftmost components seems to be easily separable from the others. The remaining three exhibits more of an overlap and some clustering schemes might prefer to join or split them further.

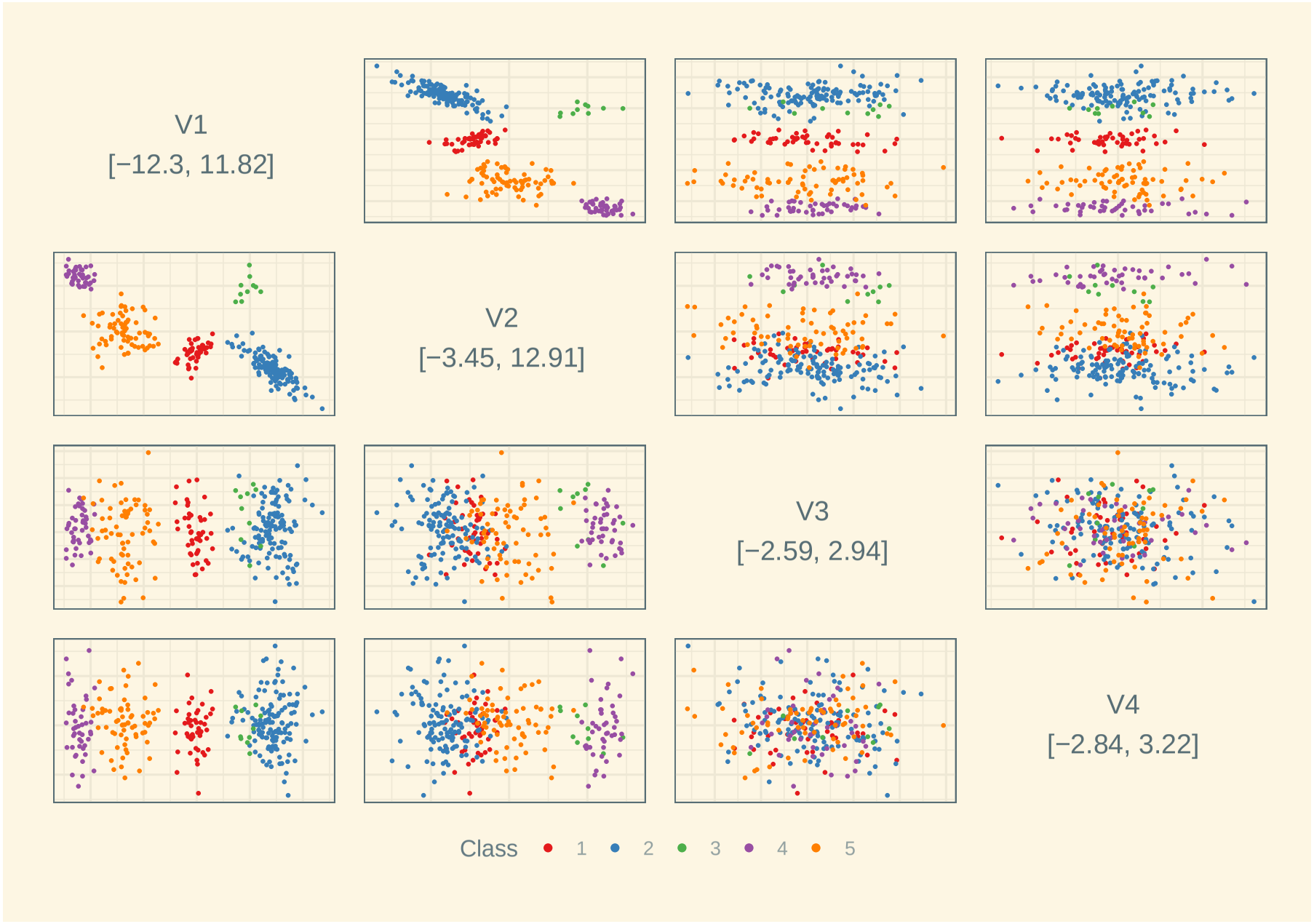


Figure 2.12: Pairs plot of the first 4 dimension of the t510D design. Showing remaining dimensions is uninformative. The first two dimensions (V1, V2) reproduce the 5t2D design; along this pairs 5 clusters appears the most reasonable choice. Pairs (Vj,Vi), with $j = 1, 2$, $i = 3, 4$ reproduce the effect of interacting the clustered dimension with the noise: here we can still see that a clustering might be found with either 3 or 4 components (e.g. (V2,V3) and (V1,V4)). The noise dimensions appear clearly unclustered.

2.4.2 Clustering solutions under comparison

In this section we define the set $\mathcal{M}$ of considered methods. These methods will be used to compare criteria to select clustering solutions.

In this Chapter we were interested in cases in which it is reasonable to assume that data can be approximately described in terms of elliptical-symmetric density regions (Section 2.1). In such cases, we also saw that (Subsection 2.1.1) mixture model-based clustering, as described in (Subsection 1.3.1), based on elliptic-symmetric families (e.g. (2.9)), is a natural choice.

Among possible approaches proposed in the literature, we decided to resort to Gaussian mixture models. There are other extremely valid proposals that we could have considered here, as for example Student-t mixture model (e.g. see Peel and McLachlan, 2000, McLachlan and Peel, 2000), which for some of the designs described above (e.g. Olive or t510D; see Table 2.1) may also be more suitable.

However, we chose to adopt Gaussian mixture models because there is an higher availability of well-established software packages that implement differently the model estimation. In particular, various implementations treat differently some modelling parameters. This allows us to analyse the ability of our proposed methodology to take into account these differences, being able to compare and select clustering solutions obtained by similar models implemented in different way. This is an advantage of our methodology (as also discussed in Section 2.1), since comparing different implementations of the same model is something that cannot be usually done explicitly using classical state-of-the-art criteria (see Subsection 2.1.1).

In this analysis we consider the implementation of Gaussian mixture models via two popular packages in *R*. These are the Mclust package (Scrucca et al., 2016) and the OTRIMLE package (Coretto and Hennig, 2017b).

Mclust implements model-based clustering based on Gaussian-mixture models. The package allows to specify the number of components and the parametrizations of the covariance matrices reviewed in Table 1.2.[6] Also, in model estimation, Mclust implements a Bayesian regularization of the covariance matrices as proposed in Fraley and Raftery, 2007. It is not possible for the end user to control it. Also, the package does not allow to choose the initial partition.

Overall, we consider 140 model estimated via Mclust. These are obtained considering mixtures with K components, for $K = 1, 2, \ldots, 10$; for each number of component K, we consider all the 14 parametrizations reviewed in Table 1.2. These constitute a set of methods $\mathcal{M}^{MC}$. Thus, for example, an element of this set contains the following information: number of components, K; covariance parametrization; and estimation method. In $\mathcal{M}^{MC}$, the computational method for the mixture parameters estimation is always the EM algorithm as implemented in Mclust. For example:

$$m: \quad (K(m) = 5); \quad (\Sigma_k = \lambda I,\ k = 1, \ldots, 5); \quad \text{(Mclust EM)}$$

The OTRIMLE package also allows to estimate Gaussian mixture models.[7] This package allows to specify the mixture's number of components. Differently from Mclust, this package

[6] The 14 parametrizations shown in table Table 1.2 are named in *mclust* with the following acronyms (in order): {EII, VII, EEI, VEI, EVI, VVI, EEE, EVE, VEE, VVE, EEV, VEV, EVV, VVV}.

[7] This package also allows to estimate noise proportion in the data; this feature is not used here.

implements a regularization for the covariance matrices via the eigenratio constraint (see Subsection 1.3.1; Coretto and Hennig, 2017a). The value for the constraint can be set by the user. Also, OTRIMLE allows the user to choose different initial partitions to initialize the EM algorithm.

Overall, we consider 180 estimation settings for the OTRIMLE. These are obtained considering mixtures with K components, for $K = 1, 2, \ldots, 10$; for each number of components K, we consider the following values of the eigenratio constraint $\gamma = \{1, 5, 10, 100, 1000, 10000\}$. For each combination of (K, γ) we consider three different initialization methods, $I = \{1, 2, 3\}$.[8] These constitute a set of methods $\mathcal{M}^{OT}$. Thus, for example, an element of this set contains the following information: number of components, K, eigenratio constraint, γ; initialization I, and estimation method. In $\mathcal{M}^{OT}$, the estimation method for the mixture parameters is always the EM algorithm as implemented in OTRIMLE. For example:

$$m: \quad (K(m) = 2); \quad (\gamma = 10); \quad (I = 1); \quad (\text{OTRIMLE EM})$$

The final set of considered methods to obtain a clustering solution is $\mathcal{M} = \mathcal{M}^{MC} \cup \mathcal{M}^{OT}$. For each element of m we will estimate the parameters of the implied Gaussian mixture model on data, obtaining $m(\mathbb{X}_n)$. The estimation is performed according to the estimation method (taking into account restrictions and initializations) specified by m. From these we obtain $\theta(m(\mathbb{X}_n))$, that is used in computing the selection criteria.

Remark 2.4.3. *The reason why we compare solutions based on Mclust against solutions based on OTRIMLE is because this allows for an interesting comparison where different implementations of the same estimation method (i.e. the EM approximation of the MLE for Gaussian mixture models), introduce different method's tuning to manage similar modelling aspects. In fact, both OTRIMLE and Mclust allow to control the overall relative discrepancy of the groups' shape, and to impose a lower bound for the within-cluster variance (covariance regularization).*

However, they do it in a dramatically different way. Mclust does not allow the user to control the lower bound for the within-cluster variance, but it allows to choose between a number of covariance matrix parametrizations to control the relative discrepancy in cluster shapes. Although the effect of choosing different covariance model finally controls the relative difference in cluster shapes, originally the covariance model selection has been introduced to control the dimensionality of the underlying mixture model parameter space. Note that for the Mclust approach going from one covariance model to another does not produce a continuous path in the implied parametrization.

On the other hand, the OTRIMLE also allows to control similar aspects of the clustering, but it does it in a completely different way. The eigenratio constraint parameter controls the relative discrepancy between cluster shapes in a continuous way, so that one goes from imposing all equal spherical shapes ($\gamma = 1$) to arbitrary shapes ($\gamma = +\infty$). Moreover, the eigenratio constraint imposes a lower bound to the within-cluster variances, although the link between the two things is not direct.

[8] We choose between OTRIMLE's default method, k-means, and pam. For further reference see Coretto and Hennig, 2017b.

For the Mclust parametrization one can compute degrees of freedom, ν, of each model specification, this is not trivial for the OTRIMLE approach. It can be argued that, for a fixed K, an OTRIMLE solution computed with $\gamma = 100$ corresponds to an higher degree of model complexity compared to the case where $\gamma = 3$, although in practice it is not possible to quantify what is the magnitude of the difference in the implied model complexity. Furthermore, when we jointly vary both K and γ, it is not totally clear how to order the relative model complexity (see also considerations in the next Section).

Summarizing, for each dataset considered in Subsection 2.4.1 we compare a total of 320 clustering solutions. Of these, 140 are obtained using the Mclust package, and 180 are obtained by using the OTRIMLE package.

2.4.3 Selection methods under comparison

For each dataset described in Subsection 2.4.1, we compare 15 different criteria to select a clustering solution. We may distinguish three main set of these: in-sample criteria, cross-validation criteria, (bootstrap) resample criteria.

In the following, we will refer to the sample data as $\mathbb{X}_n$. Also, $\mathcal{M}$ refers to the set of methods described in Subsection 2.4.2. The criteria are described as follows.

In-sample criteria

For each $m \in \mathcal{M}$, estimate $m(\mathbb{X}_n)$, call $\hat{\mathcal{M}}$ the set of estimated solutions. Then, the in-sample criteria evaluates $\hat{m}_n \in \hat{\mathcal{M}}$ on $\mathbb{X}_n$ and choose the best solution as follows. Note that some criteria can only be computed for solutions obtained by methods in $\mathcal{M}^{MC}$. Call this subset of estimated solutions $\hat{\mathcal{M}}^{MC}$. Indeed, these criteria require the number of estimated parameters ν, only available for parametrizations available in Mclust (see Remark 2.4.3).

CHC Selects the solution maximizing the CHC implied by m (see 2.1):

$$m_{\text{CHC}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}} CH(\hat{m}_n).$$

ASW Selects the solution maximizing the ASW implied by m (see 2.2):

$$m_{\text{ASW}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}} ASW(\hat{m}_n).$$

LogLk Selects the solution maximizing the log-likelihood implied by m (see 2.3):

$$m_{\text{LogLk}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}} l(\hat{m}_n).$$

HSC Selects the solution maximizing HSC as implied by m (see 2.13):

$$m_{\text{HSC}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}} HSC(\hat{m}_n).$$

SSC Selects the solution maximizing SSC as implied by m (see 2.14):

$$m_{\text{SSC}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}} SSC(\hat{m}_n).$$

AIC (Mclust only) Selects the solution maximizing the AIC implied by m (see 2.6):

$$m_{\text{AIC}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}^{MC}} AIC(\hat{m}_n).$$

AIC3 (Mclust only) Selects the solution maximizing the AIC3 implied by m (see 2.7):

$$m_{\text{AIC3}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}^{MC}} AIC3(\hat{m}_n).$$

BIC (Mclust only) Selects the solution maximizing the BIC implied by m (see 2.4):

$$m_{\text{BIC}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}^{MC}} BIC(\hat{m}_n).$$

ICL (Mclust only) Selects the solution maximizing the ICL implied by m (see 2.5):

$$m_{\text{ICL}} = \arg\max_{\hat{m}_n \in \hat{\mathcal{M}}^{MC}} ICL(\hat{m}_n).$$

Cross-validation criteria

These criteria are computed via a ten-fold cross validation (e.g. see Arlot and Celisse, 2010; also compare Subsection 2.1.1). The procedure is as follows. Randomly shuffle observations of $\mathbb{X}_n$; partition in ten (roughly) equal parts:

$$\mathbb{X}_n = \mathbb{X}^{(1)}_{\sim n/10} \cup \mathbb{X}^{(2)}_{\sim n/10} \cup, \ldots, \cup \mathbb{X}^{(10)}_{\sim n/10}$$

(where all the set unions are disjoint unions). For each $m \in \mathcal{M}$, let $\hat{m}^{(i)}$ be $m(\mathbb{X}_n \backslash \mathbb{X}^{(i)}_{\sim n/10})$ (method fitted on all but i-th partition's data). Then, score $\hat{m}^{(i)}$ on the i-th partition. Select the model maximizing the average:

$$m^*_s = \arg\max_{m \in \mathcal{M}} \frac{1}{10} \sum_{i=1}^{10} \sum_{x \in \mathbb{X}^{(i)}_{\sim n/10}} s(x, \hat{m}^{(i)}). \tag{2.58}$$

Eventually, re-estimate the selected m^* on sample data $\mathbb{X}_n$, obtaining:

$$m_{\text{10CVs}} = m^*_s(\mathbb{X}_n).$$

Criteria differ in the scoring function s used in (2.58).

10CVLogLk Replace s in (2.58) with the mixture density implied by the model:

$$s \leftarrow \log(f(x; m));$$
$$m_{\text{10CVLogLk}} = m_s^*(\mathbb{X}_n).$$

where $f(\cdot; m)$ is (1.1) parametrized at parameters $\theta(m)$.

10CVHS Replace s in (2.58) with HS (2.11):

$$s \leftarrow HS(x; m);$$
$$m_{\text{10CVHS}} = m_s^*(\mathbb{X}_n).$$

10CVSS Replace s in (2.58) with SS (2.12):

$$s \leftarrow SS(x; m);$$
$$m_{\text{10CVSS}} = m_s^*(\mathbb{X}_n).$$

Bootstrap resampling criteria

These criteria make use of the proposed resampling procedure extensively described in Subsection 2.2.1 (see also Algorithm 2).

BHSC Takes the solution maximizing BHSC (see (2.24)) and eventually re-estimate the selected model on data $\mathbb{X}_n$:

$$m^* = \arg\max_{m \in \mathcal{M}} BHSC(m);$$
$$m_{\text{BHSC}} = m^*(\mathbb{X}_n).$$

BSSC Takes the solution maximizing BSSC (see (2.25)) and eventually re-estimate the selected model on data $\mathbb{X}_n$:

$$m^* = \arg\max_{m \in \mathcal{M}} BSSC(m);$$
$$m_{\text{BSSC}} = m^*(\mathbb{X}_n).$$

BLogLk This applies the same procedure described in Algorithm 2 replacing S with the log-likelihood, l, implied by model m (compare with (2.3); how this should be done is detailed in Algorithm 2 where s should be replaced by $\log f(x; m)$, which is the density (1.1) implied by m; compare also with (2.25)).

$$BLogLk(m) := S_n^*(m) - \frac{U_n^*(m)}{a_n}; \quad S = l;$$
$$m^* = \arg\max_{m \in \mathcal{M}} BLogLk(m);$$
$$m_{\text{BLogLk}} = m^*(\mathbb{X}_n).$$

All the described criteria above will ultimately select a clustering method's setup. In the

next section, we discuss the results of the empirical analysis comparing them. Before moving on, we stress two points regarding some in-sample criteria and the cross-validation criteria.

Remark 2.4.4. *Although the original idea of Smyth (Smyth, 2000) was to cross-validate a measure of risk given by the negative expected log-likehood, here we extended the cross-validation approach to the scoring measures proposed in this work.*

Regarding the selection methods based on information-theoretic indexes (AIC, AIC3, BIC and ICL), these are only compared for clustering methods in $\mathcal{M}^{MC}$, that correspond to the Mclust setups. In fact these information theoretic quantities, as discussed in Subsection 2.1.1, requires that the model complexity can be quantified in terms of underlying model parameters. As shown in Remark 2.3.6 this not possible for the OTRIMLE setups

2.4.4 Results

In this section, we show the results on the selection criteria exposed in Subsection 2.4.2. We report the experimental results on the 3 real and 4 simulated datasets (see Table 2.1 and Remark 2.4.2). For each of them we repeated the analysis described in Subsection 2.4.2 obtaining a list of 15 selected methods, $\mathcal{M}^* = \{m_{\text{CH}}, \dots, m_{\text{BLogLk}}\}$.

In order to make comparisons, we evaluated each of the selected solutions against the ground true partition, which is normally not available in practical applications. For the real datasets, this is given by the original classes the points belong to. For the simulated datasets, we assume that the "true" partition is given by assigning points to the mixture component which generated them. Call the ground true partition m_{true}. Then, for each of the solutions $m \in \mathcal{M}^*$, we computed external validation indexes on the couple (m, m_{true}). The indexes computed are the Adjusted Rand Index (ARI), the misclassification rate (CE) and the Variation of Information (VI). These are reviewed in Section 2.6. We present CE and VI as $1 - CE$ and $-VI$, so that all of the three indexes indicate higher similarity of the solutions when they take higher values.

Furthermore, it could happen that, in some settings, none of the criteria was able to select a good solution, while this was indeed among the considered 320 methods, $\mathcal{M}$. To account for this, we also compute the best possible achievable misclassification rate within the considered $\mathcal{M}$. This is the closest solution to the ground truth, in the list of candidates $\mathcal{M}$, that is

$$m_{closest} = \arg\max_{m \in \hat{\mathcal{M}}} \{1 - CE(m, m_{\text{true}})\},$$

(where $\hat{\mathcal{M}}$ is the set of methods estimated on the original data).

Comparisons of models in $\mathcal{M}^*$ and m_{closest} are shown in Table 2.2 and Table 2.3 for real and simulated datasets respectively. Tables show for each model in $\mathcal{M}^*$ the values of the external indexes. We also report them for m_{closest}. An arrow highlights the best models in $\mathcal{M}^*$ in term of $1 - CE$. We note here that, at least in our experiments, all the three external indexes achieve (almost always) the same identical ranking for the considered solutions.

Before commenting the results, let us also introduce the visualization method we propose. This is a compact way to visualize, for any of the criteria described in Subsection 2.4.2, the values achieved by all of method in $\mathcal{M}$. We think these simple plot may be very useful especially when there are a lot of solutions to compare. Figures 2.13 and 2.13 give an example.

To construct these plots, we need to (approximately) order the models in terms of increasing complexity. In light of the discussion in Subsection 2.4.2 (in particular, see Remark 2.4.3), it is possible to find an approximate ordering by sorting clustering methods based on the parameters they receives as input, in a hierarchical fashion: we first order on inputs that are expected to have an higher impact on the complexity.

As an example, for methods in both sets $\mathcal{M}^{MC}$ and $\mathcal{M}^{OT}$, we first sort them according to increasing number of mixture components K. Indeed, this parameter is expected to have the highest impact on model complexity. Secondly for $\mathcal{M}^{MC}$, for each value of $K' = 1, \dots, 10$, we order methods with $K(m) = K'$ by the number of parameters ν they require to estimate. In this case, this is a straightforward solution to account for model complexity. For $\mathcal{M}^{OT}$ we order, within methods with K' components, by increasing values of γ (eigenratio constraint). Finally, we further order, within methods with the same (K', γ), by initialization methods. Here we simply fix an arbitrary order for these.

Even if it is true that this ordering may not be monotonically increasing in model complexity (e.g. a solution with $K = 7$ and unrestricted covariances is likely more complex than a solution with $K = 8$ and spherical covariances), the idea is to decide a ranking for the inputs that the clustering methods receive. Then, a hierarchical ordering can be pursued based on the effect that the values of each input have on model complexity.

Figures 2.13 and 2.13 show the values of the LogLk (2.3) and BSSC (2.25) criteria for all the 320 methods considered. They are constructed as follows. On the x-axis we list on the left the $\mathcal{M}^{MC}$ and $\mathcal{M}^{OT}$ on the right. The two are ordered as described above. The green line is the SSC (2.14). The blue line is the bootstrap average (2.21) and the shaded are its confidence intervals as shown in (2.23), so that the lower bound of this is the BSSC criterion (2.25). The magenta line is drawn in correspondence of m_{BSSC}.

In the graph, overall, we can see that as the models becomes more complex, the fitting of the underlying data becomes better (green line goes up). However, some of this improved fitting is going to be due to excessive adaptation to the data. This is shown by the blue line lying below the green one, helping to visualize the overfitting problem (extremely evident for Iris, Banknote and t510D data). Also, the confidence intervals for the bootstrapped quantity tend to become wider for more complex models. This is expected: models overfitting the data are likely to have high variance due to the overfit with respect to minimal changes in the data. The upward trending behaviour of the green line in these graphs is exactly what penalizations *a la* BIC or ICL try to cope with: penalization will pull the green line down as models become more complex; this will ultimately help select a model that does not overfit. In this case, this behaviour can be visualized explicitly.

Overall, this kind of graphs seems appealing for their informativeness: they allow to compactly visualize results from different models and algorithms. Similar graph can be obtained for the other criteria. Discussing the results based on all the graphical displays of the 15 selection methods for each of the 7 designs, would have made the analysis rather difficult. The pictures cannot show the details of the winning solution. Moreover comparing the 15 selection methods in a single plot for each dataset is not feasible. However, here we want to stress again that visualization like Figure 2.13 and Figure 2.14 are valuable representation of the selection mech-

anism. In fact, as discussed before, they show how the variance introduced by the unnecessary model complexity enters in action, and, for resample procedures, contrary to what offer existing methods they give a clear picture of the uncertainty of the solutions considered in the list of candidates.

We now discuss in detail the results on each dataset. Performances will be discussed in term of the external validation indexes mentioned above.

Olive data

Refer to Table 2.2(a) and Figure 2.13(a).

As seen in Subsection 2.4.1, this data has two possible ground truths: 3 or 9 classes. However, since most of the criteria select solutions closer to the latter classification, we considered this as the true partition. We immediately note that m_{closest} is a model with 8 groups. This means that even if models with the true number of components were considered, it is likely not possible to retrieve the ground truth via Gaussian mixture models.

As we were anticipating, this data poses serious difficulties for most classical methodologies. We note that almost all in-sample methods select likely overfitted models, which show poor performances. This is not true for the ICL that seems to over-penalize, choosing a solution with 6 groups, similarly to cross-validated criteria. However, this solution does not seem optimal neither in the sense of 9 true classes nor in the sense of 3.

An exception are the ASW and CHC scores, which seem to aim to retrieve the 3 true classes partition.

Finally, the bootstrap resampling scheme proves to be the top performing solution. The chosen partition has 8 groups reaching a misclassification rate very close to the best possible score.

In this case, Figure 2.13(a), is showing a very noisy pattern. This is likely due to the fact that models with highly restricted covariances perform poorly on the data: they can not fit well the data scatters, which we showed to be very concentrated on hyperplanes (see Figure 2.7).

Iris data

Refer to Table 2.2(b) and Figure 2.13(b).

In-sample non-penalized criteria perform poorly, preferring an excessive fit of the data. On the other hand, the penalty of ICL, BIC seems too strong in this context.

On the Iris data both bootstrap and cross-validation works well. This probably helps to disentangle the two overlapping classes. Also, the overlap causes the ASW to identify 2 spherical components rather than three.

Furthermore, we see a clear overfitting path in the graph, which exacerbate after $K = 3$, and the increasing variability for more complex model.

Banknote data

Refer to Table 2.2(c) and Figure 2.13(c).

The banknote dataset appears to be of a different nature than the other dataset. Here all methods performs quite poorly in retrieving the two groups except for AWS and CHC finding the optimal partition. Typically, a 3 groups partition is selected. This is not totally unreasonable: by looking at the last column of Figure 2.9, the third cluster is used to account for the more variable counterfeit notes, departing from the core. In this case the second best solution is given by BHSC and BSSC.

Not-penalized in-sample criteria also selects two group. However, these are the worst possible groups as the validation indexes show. As usual, this is motivated by overfitting (here due to unrestricted covariances) as can be seen clearly from the graph.

Pentagon5 design

Refer to Table 2.3(a) and Figure 2.14(a).

In the pentagon5 data, several methods finds an equally valid partition. However, this is always a 3 components partition, while the true data has 5 true groups. Indeed, the closest partition to the truth m_{closest} is a one with 5 groups.

As expected, most algorithms fuse together the overlapping components (refer to Figure 2.10). Here the overlap is so significant that this seems a reasonable solution. Also, other selected partition with 5 groups are not partitioning the data as good as 3 groups solutions.

It is interesting to observe the path on the graph. This is the unique case where the green line does not lie always above the blue one. In this case, we attribute this to the high variability of the estimates and to the greater impact that the resampling scheme has on data information, due to the two extremely under-represented mixture components (here bootstrap resamples may easily not include any point from the under-represented components).

t52D design

Refer to Table 2.3(b) and Figure 2.14(b).

The t52D is also an easy design for most criteria: even if the best criteria are 10CVHS, 10CVSS, BHSC, BSSC, also AIC3, BIC, ICL and ASW select substantially equally valid solutions.

t510D design

Refer to Table 2.3(c) and Figure 2.14(c).

The design t510D is interesting in that there seems to be no consensus at all from the model selection criteria. Only 10CVSS and BSSC were able to find surprisingly good partitions. BSSC also select the best possible solution. This design was expectedly hard due to the noise components and the small sample size.

Note also the profound different behaviour of the green and blue lines in the graph: resampling here plays a fundamental role in finding the optimal solution.

Uniform design

Refer to Table 2.3(d) and Figure 2.14(d).

Finally, in the uniform setting only the ICL, 10CVSS and BSSC were able to correctly avoid to partition the data.

Interestingly, if not at all penalized, SSC returns 8 clusters. This confirms the overestimating behaviour of this statistic and that a penalization is required. Resampling can also be seen as to provide a sort of penalization.

Also, it appears that the entropy-type built-in penalization of SS (see (2.12)) makes this method superior in this case, since, for other criteria, neither bootstrap nor cross-validation worked (compare also with the discussion on the differences between the smooth and the hard score in Section 2.2).

Table 2.2: Real dataset results. Panels show the selected methods in $\mathcal{M}^*$ (second column) for the 15 selection criteria (column one). Second columns report methods settings (O for OTRIMLE; M for MCLUST). Validation indexes are reported for the selected solution (last three columns). Line outside table shows the indexes computed for the closest partition to the truth, m_{closest}.

(a) Olive dataset

	Method	Selected	ARI	1-CE	-VI
	Samp. LogLk	M, $K = 10$, VVV	0.54	0.71	-1.25
	HSC	M, $K = 10$, VVV	0.54	0.71	-1.25
	SSC	M, $K = 10$, VVV	0.54	0.71	-1.25
	AIC	M, $K = 10$, VVV	0.54	0.71	-1.25
	AIC3	M, $K = 10$, VVV	0.54	0.71	-1.25
	BIC	M, $K = 10$, VVE	0.59	0.80	-1.11
	ICL	M, $K = 6$, VVV	0.78	0.79	-0.90
	ASW	O, $K = 2$, $\gamma = 5$	0.29	0.44	-2.27
	CHC	M, $K = 3$, EII	0.42	0.52	-2.27
	10CV LogLk	M, $K = 9$, VEE	0.65	0.81	-1.00
	10CV HS	M, $K = 6$, VVV	0.79	0.79	-0.90
	10CV SS	M, $K = 6$, VVV	0.79	0.79	-0.90
→	BLogLk	M, $K = 8$, VVV	0.86	0.88	-0.74
→	BHSC	M, $K = 8$, VVV	0.86	0.88	-0.74
→	BSSC	M, $K = 8$, VVV	0.86	0.88	-0.74
	Closest	M, $K = 8$, EVE	0.88	0.90	-0.65

(b) Iris dataset

	Method	Selected	ARI	1-CE	-VI
	Samp. LogLk	O, $K = 10$, $\gamma = 10000$	0.44	0.54	-1.79
	HSC	O, $K = 10$, $\gamma = 10000$	0.44	0.54	-1.79
	SSC	O, $K = 10$, $\gamma = 10000$	0.44	0.54	-1.79
	AIC	M, $K = 9$, VEV	0.37	0.49	-1.90
→	AIC3	M, $K = 3$, VEV	0.90	0.96	-0.32
	BIC	M, $K = 2$, VEV	0.57	0.67	-0.67
	ICL	M, $K = 2$, VEV	0.57	0.67	-0.67
	ASW	M, $K = 2$, VII	0.57	0.67	-0.67
	CHC	M, $K = 3$, EII	0.73	0.89	-0.76
→	10CV LogLk	O, $K = 3$, $\gamma = 100$	0.90	0.97	-0.32
→	10CV HS	O, $K = 3$, $\gamma = 100$	0.90	0.97	-0.32
→	10CV SS	O, $K = 3$, $\gamma = 100$	0.90	0.97	-0.32
→	BLogLk	O, $K = 3$, $\gamma = 100$	0.90	0.97	-0.32
→	BHSC	O, $K = 3$, $\gamma = 100$	0.90	0.97	-0.32
→	BSSC	O, $K = 3$, $\gamma = 100$	0.90	0.97	-0.32
	Closest	M, $K = 3$, EEE	0.94	0.98	-0.26

(c) Banknote dataset

	Method	Selected	ARI	1-CE	-VI
	Samp. LogLk	O, $K = 2$, $\gamma = 10000$	0.26	0.41	-2.14
	HSC	O, $K = 2$, $\gamma = 10000$	0.26	0.41	-2.14
	SSC	O, $K = 2$, $\gamma = 10000$	0.26	0.41	-2.14
	AIC	M, $K = 6$, EVE	0.40	0.55	-1.47
	AIC3	M, $K = 6$, EVE	0.40	0.55	-1.47
	BIC	M, $K = 3$, VVE	0.84	0.91	-0.43
	ICL	M, $K = 3$, VVE	0.84	0.91	-0.43
→	ASW	M, $K = 2$, EII	1	1	0
→	CHC	M, $K = 2$, EII	1	1	0
	10CV LogLk	M, $K = 4$, VVE	0.68	0.73	-0.70
	10CV HS	M, $K = 4$, VVE	0.68	0.73	-0.70
	10CV SS	O, $K = 3$, $\gamma = 10$	0.86	0.92	-0.37
	BLogLk	M, $K = 6$, EEE	0.47	0.60	-1.31
	BHSC	O, $K = 3$, $\gamma = 10$	0.86	0.92	-0.37
	BSSC	O, $K = 3$, $\gamma = 10$	0.86	0.92	-0.37
	Closest	M, $K = 2$, EII	1	1	0

Table 2.3: Simulated dataset results. Panels show the selected methods in $\mathcal{M}^*$ (second column) for the 15 selection criteria (column one). Second columns report methods settings (O for OTRIMLE; M for MCLUST). Validation indexes are reported for the selected solution (last three columns). Line outside table shows the indexes computed for the closest partition to the truth, m_{closest}.

(a) pentagon5 design

	Method	Selected	ARI	1-CE	-VI
	Samp. LogLk	O, $K = 10$, $\gamma = 1000$	0.44	0.60	−1.66
→	HSC	M, $K = 3$, VVV	0.86	0.92	−0.39
→	SSC	M, $K = 3$, VVV	0.86	0.92	−0.39
	AIC	M, $K = 5$, EII	0.85	0.91	−0.56
	AIC3	M, $K = 5$, EII	0.85	0.91	−0.56
	BIC	M, $K = 5$, EII	0.85	0.91	−0.56
→	ICL	M, $K = 3$, EVE	0.86	0.91	−0.39
→	ASW	M, $K = 3$, EII	0.86	0.92	−0.39
→	CHC	M, $K = 3$, EII	0.86	0.92	−0.39
	10CV LogLk	M, $K = 5$, EVI	0.86	0.92	−0.55
→	10CV HS	M, $K = 3$, EVE	0.86	0.92	−0.39
→	10CV SS	M, $K = 3$, EVE	0.86	0.92	−0.39
	BLogLk	O, $K = 5$, $\gamma = 1$	0.85	0.91	−0.56
→	BHSC	M, $K = 3$, EVE	0.86	0.92	−0.39
→	BSSC	M, $K = 3$, EVE	0.86	0.92	−0.39
	Closest	M, $K = 5$, EEE	0.87	0.93	−0.38

(b) t52D design

	Method	Selected	ARI	1-CE	-VI
	Samp. LogLk	O, $K = 10$, $\gamma = 10000$	0.74	0.79	−0.89
	HSC	O, $K = 9$, $\gamma = 10000$	0.84	0.91	−0.50
	SSC	O, $K = 9$, $\gamma = 10000$	0.84	0.91	−0.50
	AIC	M, $K = 10$, VVV	0.76	0.83	−0.86
	AIC3	M, $K = 5$, VVE	0.95	0.97	−0.28
	BIC	M, $K = 5$, VVE	0.95	0.97	−0.28
	ICL	M, $K = 5$, VVE	0.95	0.97	−0.28
	ASW	M, $K = 5$, EEE	0.96	0.98	−0.22
	CHC	M, $K = 6$, VEV	0.80	0.89	−0.50
	10CV LogLk	O, $K = 6$, $\gamma = 5$	0.88	0.94	−0.38
→	10CV HS	O, $K = 5$, $\gamma = 5$	0.96	0.99	−0.17
→	10CV SS	O, $K = 5$, $\gamma = 5$	0.96	0.99	−0.17
	BLogLk	O, $K = 6$, $\gamma = 5$	0.88	0.94	−0.38
→	BHSC	O, $K = 5$, $\gamma = 10$	0.96	0.99	−0.17
→	BSSC	O, $K = 5$, $\gamma = 10$	0.96	0.99	−0.17
	Closest	M, $K = 5$, EEV	0.97	0.99	−0.17

(c) 5t10D design

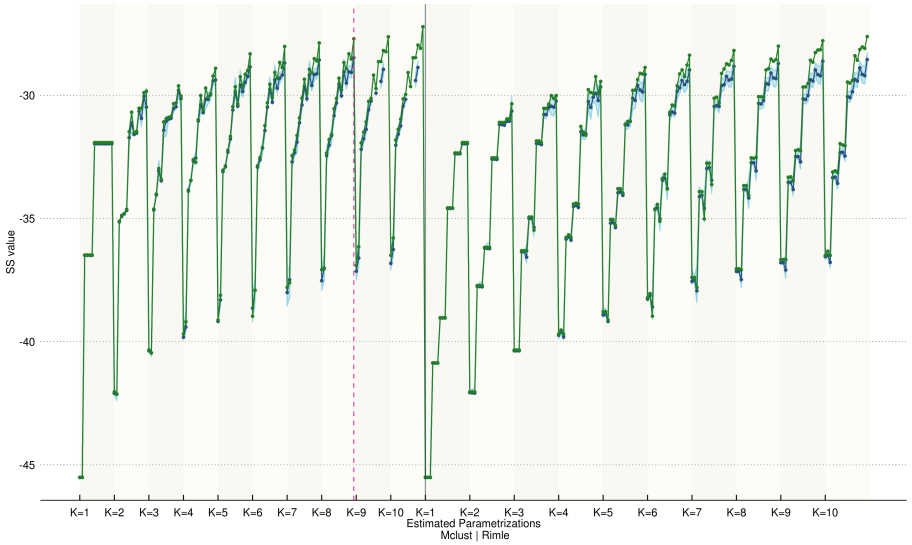
	Method	Selected	ARI	1-CE	-VI
	Samp. LogLk	O, $K = 10$, $\gamma = 10000$	0.51	0.61	−1.26
	HSC	O, $K = 10$, $\gamma = 10000$	0.51	0.61	−1.26
	SSC	O, $K = 10$, $\gamma = 10000$	0.51	0.61	−1.26
	AIC	M, $K = 9$, EVE	0.73	0.80	−1.21
	AIC3	M, $K = 8$, VII	0.65	0.72	−0.80
	BIC	M, $K = 6$, VII	0.75	0.79	−0.50
	ICL	M, $K = 6$, VII	0.75	0.79	−0.50
	ASW	M, $K = 2$, EII	0.54	0.68	−1.03
	CHC	M, $K = 2$, EII	0.54	0.68	−1.03
	10CV LogLk	O, $K = 8$, $\gamma = 1$	0.60	0.64	−0.94
	10CV HS	O, $K = 6$, $\gamma = 1$	0.73	0.79	−0.59
	10CV SS	O, $K = 5$, $\gamma = 1$	0.97	0.99	−0.17
	BLogLk	O, $K = 8$, $\gamma = 5$	0.57	0.66	−1.00
	BHSC	O, $K = 8$, $\gamma = 5$	0.57	0.66	−1.00
→	BSSC	O, $K = 5$, $\gamma = 5$	0.98	0.99	−0.14
	Closest	O, $K = 5$, $\gamma = 5$	0.98	0.99	−0.14

(d) Uniform design

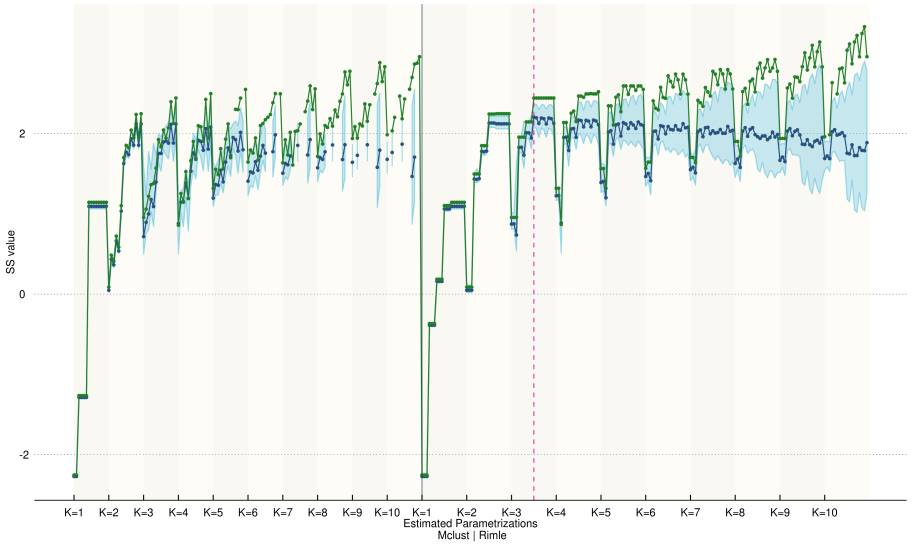
	Method	Selected	ARI	1-CE	-VI
	Samp. LogLk	O, $K = 10$, $\gamma = 1000$	0	0.14	−3.24
	HSC	O, $K = 10$, $\gamma = 1000$	0	0.14	−3.24
	SSC	M, $K = 8$, VVV	0	0.31	−2.96
	AIC	M, $K = 10$, VVI	0	0.20	−3.11
	AIC3	M, $K = 6$, VVE	0	0.24	−2.50
	BIC	M, $K = 5$, VVE	0	0.34	−2.22
→	ICL	M, $K = 1$, EII	1	1	0
	ASW	M, $K = 3$, EEV	0	0.36	−1.58
	CHC	M, $K = 9$, EII	0	0.16	−3.13
	10CV LogLk	M, $K = 7$, VEE	0	0.24	−2.63
	10CV HS	O, $K = 5$, $\gamma = 100$	0	0.31	−2.25
→	10CV SS	M, $K = 1$, EEI	1	1	0
	BLogLk	M, $K = 10$, VVI	0	0.20	−3.11
	BHSC	O, $K = 9$, $\gamma = 10000$	0	0.20	−3.03
→	BSSC	M, $K = 1$, EEI	1	1	0
	Closest	M, $K = 1$, EII	1	1	0

Figure 2.13: Real data. (Blue-line) Bootstrapped value of SS (2.21). (Shaded light blue region) SS bootstrap confidence interval (10% level); the lower bound of this is BSSC (2.25). (Green line) SSC (2.14); this correspond to the in-sample counterpart of the blue line.

(a) Olive data; selected mclust, $K = 8$, cov=VVV



(b) Iris data; selected rimle, $K = 3$, $\gamma = 100$



(c) Banknote data; selected rimle, $K = 3$, $\gamma = 10$

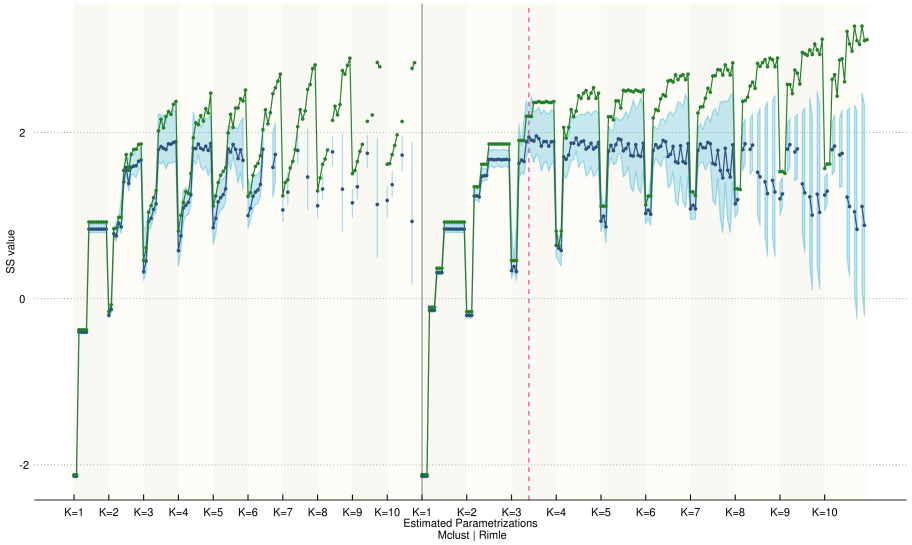
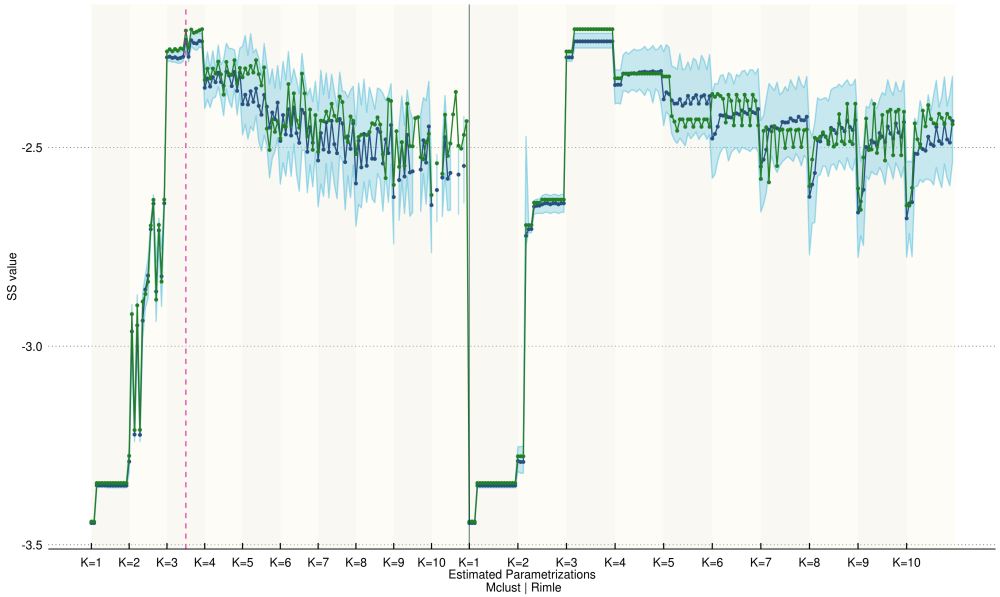
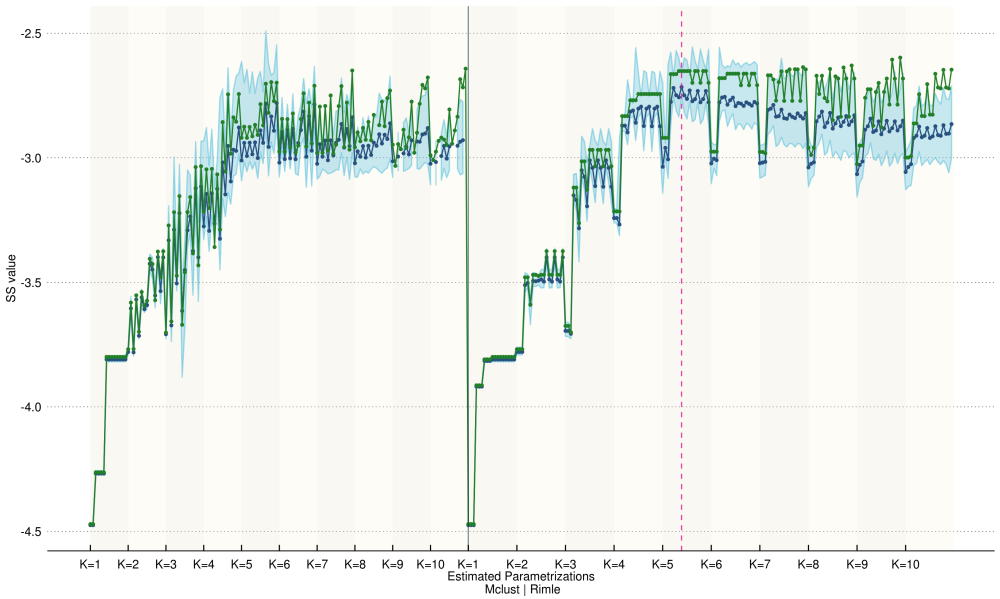


Figure 2.14: Simulated designs. (Blue-line) Bootstrapped value of SS (2.21). (Shaded light blue region) SS bootstrap confidence interval (10% level); the lower bound of this is BSSC (2.25). (Green line) SSC (2.14); this correspond to the in-sample counterpart of the blue line.

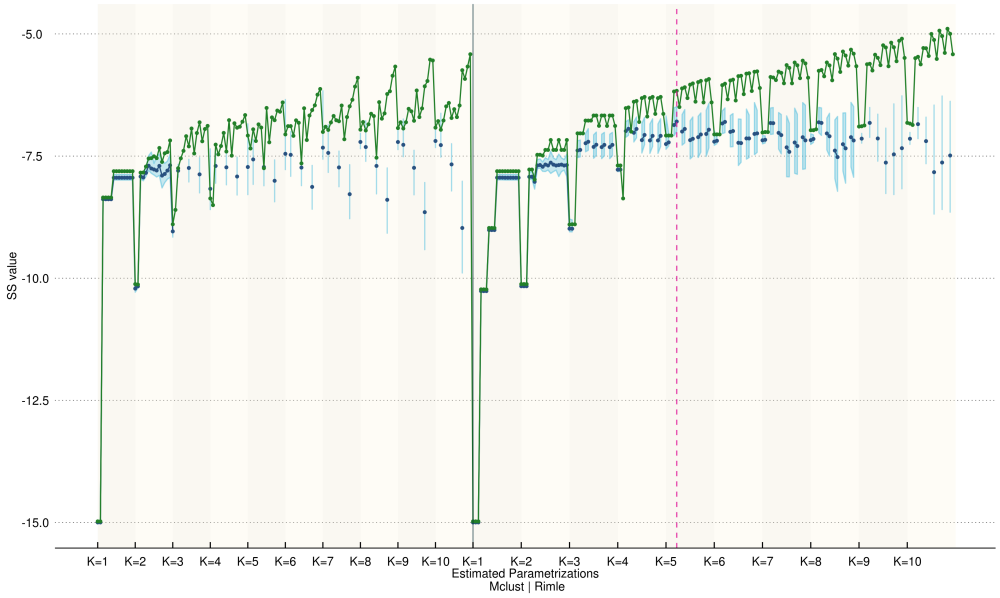
Pentagon5; selected mclust, $K = 3$, cov=EVE



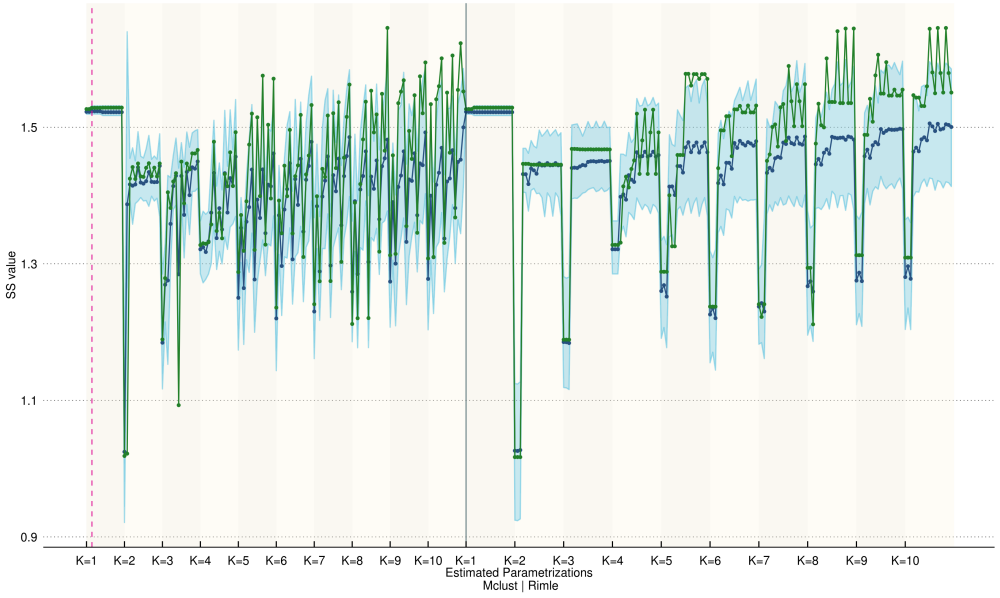
t52D; selected rimle, $K = 5$, $\gamma = 10$



t510D; selected rimle, $K = 5$, $\gamma = 5$



Uniform; selected mclust, $K = 1$, cov=EEI



2.5 Conclusions

In this Chapter we introduced a new method for comparing cluster solutions that may arise from performing different algorithms. We formalized the set of candidate cluster solutions, $\mathcal{M}$, as a set of parametric descriptions of the obtained partitions in terms of size, centrality e and scatter parameters. We introduced scoring functions that assigns a score to each element of $m \in \mathcal{M}$. The scoring is based on the quadratic discriminant function, and we showed its strong relationships to the likelihood theory for Gaussian mixture models. Despite this connection the scoring proposed here is appropriate to evaluate clustering solutions produced by class-conditional distributions that belong to a larger elliptical-symmetric family.

Based on the analysis of the empirical results we can recommend the smooth scoring 2.12 to the hard scoring 2.11. The reason why the smooth scoring approach reports better performance is because it can better manage overlapping groups. We also proposed a bootstrap-based resampling method to estimate the average score of a member $m \in \mathcal{M}$, and to construct confidence intervals for it. Based on the bootstrap estimate we proposed several strategy to perform a selection of a solution from $\mathcal{M}$.

Theoretical guarantees for the resampling method have been derived, and the overall performance of the proposed methodology have been assessed in finite samples via numerical experiments. The results have shown that the proposed method always selects solutions close to the optimal. While some of the well-established competing methods sometimes have fallen dramatically, the proposed method always provided a reasonable answer, if not the best. An additional advantage of our methods is that it provides an inbuilt tool for assessing the uncertainty related to the choice of a member of $\mathcal{M}$. This allows to assess the variance introduced by each clustering strategy under comparison.

The main drawback of our proposal is that it is computationally involved. Each member $m \in \mathcal{M}$ has to be recomputed a large B number of times. To ease the computational workload, we could run two iteration of the procedure: the first with a reduced number of bootstrap resamples, B', can be used to exclude $m \in \mathcal{M}$ which are clearly poor performing; the second iteration with bootstrap replicates B will carefully select the best model; $B' \ll B$. Also, further speed up could be obtained via the k-step bootstrap estimator described in Andrews, 2002. However, this might be hard to apply and its not clear whether it would keep the necessary variability of the estimates providing good results for the bootstrap.

2.6 Appendix: external validation indexes

In this appendix, we review the three indexes used in the empirical section to compare clustering solutions. These indexes are usually referred as *external validation* indexes, and are not used to select one clustering solution over the others. Rather, they are meant to compare a selected solution with external clustering information. In general, these indexes provide a measure of the dissimilarity between two different partitions. Thus, the information that will be compared among two clustering solutions, given by models m and m', is based on the partitions of points into clusters given by $z^{(m)}$ and $z^{(m')}$ (where we remind that these are the indicator variables indicating points' assignment to clusters implied by m and m' respectively). A comprehensive review on these methodologies is given in Meila, 2015, from which we borrow for the following discussion.

Let us define some quantities that will be useful in the exposition. Assume that we want to compare two clustering models (or their solutions) m and m'. Then, we denote with $k = 1, \dots, K(m)$ clusters defined by m and with $k' = 1, \dots, K(m')$ clusters defined by m'. Also, n_k will be the number of points in cluster k for model m:

$$n_k = \left| \left\{ x_i \in \mathbb{X}_n : z_{i,k}^{(m)} = 1 \right\} \right|$$

(analogously for $n_{k'}$); $n_{k,k'}$ will denote the number of points assigned to cluster k under m and to cluster k' under m':

$$n_{k,k'} = \left| \left\{ x_i \in \mathbb{X}_n : z_{i,k}^{(m)} z_{i,k'}^{(m')} = 1 \right\} \right|.$$

First we define the misclassification error (ME). The misclassification error between two partitions is generally understood as the proportion of mismatched assignments. However, cluster labels are not meaningful, i.e. cluster labelling is just a convenient way to identify the groups, but a label switching may imply exactly the same solution. Furthermore, we try to obtain a measure that is invariant with respect to the sample size n. Thus, define this criterion as:

$$ME(m, m') := 1 - \frac{1}{n} \max_{\pi} \sum_{k=1}^{K(m)} n_{k,\pi(k)}, \tag{2.59}$$

where we assume, without loss of generality, that $K(m) \leqslant K(m')$ and π is a mapping $\pi : \{1, \dots, K(m)\} \to \{1, \dots, K(m')\}$. Thus, we first need to find the best matches of clusters in m to clusters in m' (note that the numerosity of the clusters need not to be the same), and then we count the proportion of misclassified points. In this sense, the misclassification error has a probabilistic interpretation as the probability of disagreeing clustering labels on data points given the best possible label correspondence (Meila, 2015). It is easy to see that this index ranges in $[0, 1]$.[9]

Another popular index to compare clustering partitions is the adjusted Rand index (ARI). It is a method correcting the Rand index to cope with some of its drawbacks. This is is one of

[9] Additional notes on *ME*: it is a metric; it can be computed in polynomial time; it is a *local* and *additive* index (see Meila, 2015).

the oldest indexes to compare partitions; it was introduced by Rand, 1971 and is defined as:

$$Rand(m, m') := \frac{N_{11} + N_{00}}{n(n-1)/2},$$

where (following Meila, 2015):

N_{11} number of pairs of point belonging to the same cluster under both m and m', that are points x_i and x_j such that $z_{i,k}^{(m)} = z_{j,k}^{(m)} = z_{i,k'}^{(m')} = z_{j,k'}^{(m')} = 1$, for some k and k'.

N_{00} number of pairs of point belonging to different clusters under both m and under m', that are points x_i and x_j such that $z_{i,l}^{(m)} = z_{j,h}^{(m)} = z_{i,l'}^{(m')} = z_{j,h'}^{(m')} = 1$, for some $h \neq l$ and $h' \neq l'$.

N_{10} number of pairs of point belonging to the same cluster under m and to different clusters under m', that are points x_i and x_j such that $z_{i,k}^{(m)} = z_{j,k}^{(m)} = z_{i,l'}^{(m')} = z_{j,h'}^{(m')} = 1$, for some k and $h' \neq l'$.

N_{10} number of pairs of point belonging to the different clusters under m and to the same cluster under m', that are points x_i and x_j such that $z_{i,h}^{(m)} = z_{j,l}^{(m)} = z_{i,k'}^{(m')} = z_{j,k'}^{(m')} = 1$, for some $h \neq l$ and k'.

Note that the total number of pairs is $n(n-1)/2$.

The problem with this index is that it is not uniformly ranging in $[0, 1]$ and will be usually pushed away from 0, since N_{00} is usually very large. To cope with this type of issues, Hubert and Arabie, 1985, proposed the following corrected version:

$$ARI(m, m') := \frac{Rand(m, m') - \mathbb{E}\, Rand(m, m')}{1 - \mathbb{E}\, Rand(m, m')}, \tag{2.60}$$

where the expected value is the value of the index for two independent clustering, i.e. the index computed as if the two partitions for m and m' were obtained at random.[10] The adjusted index is bounded by 1, and may assume negative values. This correction does not ensure that the index ranges linearly in the unit interval nor that the index is comparable for different models.[11] An higher value for the ARI indicates higher concordance of the two clustering m and m' and it is equal to 1 for perfect overlap.

The last criterion we use is the Variation of Information index, which was proposed in Meila, 2007. This index is a based on an information theoretic approach. Intuitively, this criterion captures "*the amount of information lost and gained from changing to clustering* m *to clustering* m' (Meila, 2007). Consider the following quantities:

$P(k)$ this is the probability that a point falls in cluster k under model m. Empirically, this is defined as $P(k) = \frac{n_k}{n}$ (analogously we define $P(k')$ for model m').

$H(m)$ is the entropy associated with model m; it is a measure of the uncertainty of the clustering and it is defined as $H(m) = -\sum_{k=1}^{K(m)} P(k) \log(P(k))$. The higher its value the more well balanced clusters there are.

[10] The hypothesis of independent partitions basically assumes an hypergeometric distribution for the confusion matrix with elements $n_{k,k'}$. Also, for an exact expression of the expectation term, see Meila, 2015.

[11] Additional notes on ARI: it is not local nor additive.

$P(k, k')$ Similarly to $P(k)$, this is the joint probability of a point falling in clusters k and k' under m and m' respectively. It is computed as $P(k, k') = \frac{n_{k,k'}}{n}$

$H(k, k')$ is the joint entropy defined as $H(m, m') = \sum_{k,k'} P(k, k') \log(P(k, k'))$.

$I(m, m')$ is the mutual information. It is defined as

$$I(m, m') = \sum_{k=1}^{K(m)} \sum_{k'=1}^{K(m')} P(k, k') \log(P(k, k')).$$

It can be intuitively described in the following way: the uncertainty of the assignment of a point under model m' is given by $H(m')$. If we knew the assignment of the point under m, by how much does this information reduce the uncertainty of assignment under model m'? Averaging this over all points we obtain $I(m, m')$. Indeed, note that for two different models m, m' carrying no information one for the other (i.e. $P(k, k') \approx P(k)P(k')$), so that $I(m, m') \approx 0$.

Then the criterion is defined as:

$$VI(m, m') := H(m) + H(m') - 2I(m, m') = \\ \big(H(m) - I(m, m')\big) + \big(H(m') - I(m, m')\big) = 2H(m, m') - H(m) - H(m'). \quad (2.61)$$

(the above are all equivalent formulations). This index is always non-negative, and is minimized at 0 for two equal clusterings. This criterion enjoys several desirable properties, which are detailed in Meila, 2007. Among the others, it is worth noticing that it is a metric on the space of clusterings.[12]

[12] Additional properties for VI: does not depend directly on the sample size n; it is local and additive; it is bounded from above by $\log n$; bounds exists for certain clustering configuration and are well understood.

References

Akaike, H. (1973). "Information theory and an extension of the maximum likelihood principle". In:

Anderson, E. (1936). "The Species Problem in Iris". In: *Annals of the Missouri Botanical Garden* Vol. 23.No. 3, pp. 471–483. URL: https://www.jstor.org/stable/pdf/2394164.pdf?seq=1#page_scan_tab_contents.

Anderson, T. W. (2003). *An Introduction to Multivariate Statistical Analysis*. Wiley-Interscience. ISBN: 9780471360919.

Andrews, D. W. (2001). "Higher-order improvements of a computationally attractive k-step bootstrap for extremum estimators". In: *Cowles Foundation Discussion Paper No. 1230R*.

— (2002). "Higher-order improvements of a computationally attractive k-step bootstrap for extremum estimators". In: *Econometrica* 70.1, pp. 119–162.

Arlot, S. and Celisse, A. (2010). "A survey of cross-validation procedures for model selection". In: *Statistics surveys* 4, pp. 40–79.

Baudry, J. P. et al. (2015). "Estimation and model selection for model-based clustering with the conditional classification likelihood". In: *Electronic journal of statistics* 9.1, pp. 1041–1077.

Bickel, P. J. and Freedman, D. A. (1981). "Some Asymptotic Theory for the Bootstrap". In: *The Annals of Statistics*.

Bierens, H. J. (1996). *Topics in advanced econometrics: estimation, testing, and specification of cross-section and time series models*. Cambridge University Press.

Biernacki, C., Celeux, G., and Govaert, G. (2000). "Assessing a mixture model for clustering with the integrated completed likelihood". In: *IEEE transactions on pattern analysis and machine intelligence* 22.7, pp. 719–725.

Billingsley, P. (2013). *Convergence of probability measures*. John Wiley & Sons.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bozdogan, H. (1983). *Determining the Number of Component Clusters in the Standard Multivariate Normal Mixture Model Using Model-Selection Criteria*. Tech. rep. Illinois Univ. at Chicago Circle Dept. of Quantitative Methods.

Burnham, K. P. and Anderson, D. R. (2003). *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. Springer.

Caliński, T. and Harabasz, J. (1974). "A dendrite method for cluster analysis". In: *Communications in Statistics-theory and Methods* 3.1, pp. 1–27.

Celeux, G. and Soromenho, G. (1996). "An entropy criterion for assessing the number of clusters in a mixture model". In: *Journal of classification* 13.2, pp. 195–212.

Coretto, P. and Hennig, C. (2017a). *Consistency, Breakdown Robustness, and Algorithms for Robust Improper Maximum Likelihood Clustering*. Tech. rep., pp. 1–39. URL: http://jmlr.org/papers/v18/16-382.html..

— (2017b). *otrimle: Robust Model-Based Clustering*. R package version 1.1.

Efron, B. (1979). "Bootstrap Methods: Another Look at the Jackknife". In: *The Annals of Statistics* 7.1, pp. 1–26. DOI: 10.1214/aos/1176344552. URL: https://doi.org/10.1214/aos/1176344552.

Efron, B. and Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC press.

Fisher, R. A. (1936). "The use of Multiple Measurments in Taxonomic Problems". In: *Annals of Eugenics*. URL: http://rcs.chemometrics.ru/Tutorials/classification/Fisher.pdf.

Flury, B. and Riedwyl, H. (1988). *Multivariate Statistics. A practical approach*. Chapman and Hall.

Forina, M. and Amarino, C. (1982). *title not available*.

Forina, M. and Tiscornia, E. (1982). "Pattern recognition methods in the prediction of {I}talian olive oil origin by their fatty acid content". In: *Annali di Chimica* 72.January 1982, pp. 143–155.

Forina, M. et al. (1983). "Classification of olive oils from their fatty acid composition". In: *Food Research and Data Analysis* January 1983, pp. 189–214.

Fraley, C. and Raftery, A. E. (1998). "How many clusters? Which clustering method? Answers via model-based cluster analysis". In: *The computer journal* 41.8, pp. 578–588.

— (2002). "Model-based clustering, discriminant analysis, and density estimation". In: *Journal of the American statistical Association* 97.458, pp. 611–631.

— (2007). "Bayesian regularization for normal mixture estimation and model-based clustering". In: *Journal of classification* 24.2, pp. 155–181.

Giné, E. and Zinn, J. (1990). "Bootstrapping general empirical measures". In: *The Annals of Probability*, pp. 851–869.

Gonçalves, S. and White, H. (2004). "Maximum likelihood and the bootstrap for nonlinear dynamic models". In: *Journal of Econometrics* 119.1, pp. 199–219.

Halkidi, M., Vazirgiannis, M., and Hennig, C. (2015). "Method-independent indices for cluster validation and estimating the number of clusters". In: *Handbook of Cluster Analysis*. Chapman and Hall/CRC, pp. 616–639.

Hastie, T., Tibshirani, R. J., and Friedman, J. (2009). *The Elements of Statistical Learning*. 2nd ed. Springer New York. DOI: 10.1007/978-0-387-84858-7. URL: https://doi.org/10.1007/978-0-387-84858-7.

Hennig, C. and Meila, M. (2015). "Cluster analysis: an overview". In: *Handbook of Cluster Analysis. CRC Press, Taylorand Francis Group*, pp. 1–20.

Hennig, C. et al. (2015). *Handbook of cluster analysis*. CRC Press.

Hubert, L. and Arabie, P. (1985). "Comparing partitions". In: *Journal of classification* 2.1, pp. 193–218.

Keribin, C. (1998). "Consistent estimate of the order of mixture models". In: *Comptes Rendus De L Academie Des Sciences Serie I-Mathematique* 326.2, pp. 243–248.

Leroux, B. G. (1992). "Consistent estimation of a mixing distribution". In: *The Annals of Statistics*, pp. 1350–1360.

Mammen, E. (2012). *When does bootstrap work?: asymptotic results and simulations*. Vol. 77. Springer Science & Business Media.

McLachlan, G. J. and Krishnan, T. (2007). *The EM algorithm and extensions*. Vol. 382. John Wiley & Sons.

McLachlan, G. J. and Peel, D. (2000). *Finite Mixture Models*. John Wiley & Sons, Inc. DOI: 10.1002/0471721182. URL: https://doi.org/10.1002/0471721182.

McNeil, A. J., Frey, R., and Embrechts, P. (2005). *Quantitative Risk Management: Concepts, Techniques, and Tools*. Princeton University Press. ISBN: 0691122555.

McNicholas, P. D. et al. (2011). *pgmm: Parsimonious Gaussian Mixture Models*. R package version 1.0.

Meila, M. (2007). "Comparing clusterings—an information based distance". In: *Journal of multivariate analysis* 98.5, pp. 873–895.

— (2015). "Criteria for comparing clusterings". In: *Handbook of cluster analysis*. Chapman and Hall/CRC, pp. 640–657.

O'Hagan, A. et al. (2018). "Investigation of parameter uncertainty in clustering using a Gaussian mixture model via jackknife, bootstrap and weighted likelihood bootstrap". In: *Computational Statistics*, pp. 1–35.

Peel, D. and McLachlan, G. J. (2000). "Robust mixture modelling using the t distribution". In: *Statistics and computing* 10.4, pp. 339–348.

Pollard, D. et al. (1981). "Strong consistency of k-means clustering". In: *The Annals of Statistics* 9.1, pp. 135–140.

Rand, W. M. (1971). "Objective criteria for the evaluation of clustering methods". In: *Journal of the American Statistical association* 66.336, pp. 846–850.

Rousseeuw, P. J. (1987). "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis". In: *Journal of computational and applied mathematics* 20, pp. 53–65.

Rousseeuw, P. J. and Kaufman, L. (1990). "Finding groups in data". In: *Hoboken: Wiley Online Library*.

Schwarz, G. (1978). "Estimating the dimension of a model". In: *The annals of statistics* 6.2, pp. 461–464.

Scrucca, L. et al. (2016). "mclust 5: clustering, classification and density estimation using Gaussian finite mixture models". In: *The R Journal* 8.1, pp. 205–233. URL: https://journal.r-project.org/archive/2016-1/scrucca-fop-murphy-etal.pdf.

Smyth, P. (2000). "Model selection for probabilistic clustering using cross-validated likelihood". In: *Statistics and computing* 10.1, pp. 63–72.

Swayne, D. F., Lang, D. T., and Lawrence, M. (2006). *GGobi Manual*. Tech. rep. URL: http://www.ggobi.org/docs/manual.pdf.

Swiss National Bank (2019). *Second banknote series (1911)*. https://en.wikipedia.org/w/index.php?title=LaTeX&oldid=413720397. [Online; accessed 15-July-2019].

van der Vaart, A. W. (1998). *Asymptotic Statistics*. Cambridge: Cambridge University Press. ISBN: 9780511802256. DOI: 10.1017/CBO9780511802256. URL: http://ebooks.cambridge.org/ref/id/CBO9780511802256.

Velilla, S. and Hernández, A. (2005). "On the consistency properties of linear and quadratic discriminant analyses". In: *Journal of Multivariate Analysis* 96.2, pp. 219–236.

Chapter 3

Supervised Learning: Employees-to-tasks assignment in Labor Economics

3.1 Introduction

In this chapter we deal with the problem of jobs allocation. More precisely we investigate how firms assign or should assign employees to tasks. In other words, how employers select the right (wo)man for a given task, job or position. This is a relevant problem in Labor Economics and has several implications. As we will see, this is one of the drivers of productivity dispersion.

Our contribution is twofold. As we will argue, the problem of allocating employees to tasks may be treated as a classification problem. We propose to infer an optimal allocation rule from observed data via machine learning (ML) algorithms. The advantage of our approach has an advantage with respect to other methodologies (e.g. sorting; see Eeckhout, 2018) in that it does not need to model explicitly criteria with which the worker allocation should be performed. These are learned implicitly from data.

We propose several indexes to evaluate observed allocations. Based on these indices we can evaluate how well a firm allocated its workforce with respect to a benchmark allocation. In our case, the benchmark will be the ML inferred optimal allocation. These criteria are general, in the sense that they can be used whenever a criterion of suitability of worker to tasks is available.

We conduct our analysis using a huge database matching employers and employees information. This is the LISA database from Statistic Sweden. While classical methodologies from Econometrics can not be directly applied to this estimation task, machine learning methodologies allow efficient exploitation of these type of data.

The following sections are organized as follows. The rest of this section motivates the problem from an Economics perspective. Section 3.2 defines the problem of interest and frames it in a supervised learning fashion. Section 3.3 introduces measures evaluating the quality of employees-to-tasks allocations. Section 3.4 reviews the methodology we use for the estimation task and Section 3.5 presents the data used in this analysis. Finally Section 3.6 illustrates the empirical analyses, and Section 3.7 gives final comments.

3.1.1 Economic motivation

In this section, we will briefly review the motivations that make the problem we address in this chapter interesting from an Economic point of view. The relevant questions we are willing to answer are:

- whether or not there exists a regular pattern in how a firm assigns an employee to a particular job/task/position, upon observing his/her characteristics;
- how could be assessed whether or not some firms are better than others in allocating their workforce;
- whether or not firms' allocation of workforce impacts on their performances (e.g. profitability, productivity per employee).

The employees-to-tasks assignment is not new in economics. Furthermore, it typically has several implications, and there are different perspectives in the literature to look at it.. The last point above is an example of what we mean. For example, typically workers' allocation is used to explain (partially) productivity dispersion and this is also the perspective with which we look at the employees assignment problem.

It must be noted that there is an extensive literature investigating the factors determining productivity dispersion across firms. The latter has many drivers, and different approaches in Economics try to motivate it under different perspectives. Syverson, 2011 gives a review taking into account several of these approaches. Labour economists partially explain the productivity dispersion with human capital quality. This is also supported by thorough empirical studies, made possible by the increasing availability of rich databases, usually matching information on both employer and employees. For example, Abowd et al., 2005 define different measures of workers' skill and determines positive relationships between skill and productivity using data from the U.S. Census Bureau. Fox and Smeets, 2011 also explain productivity dispersion with input quality using data on all Danish citizens for the time span 1980-2001. Quoting their findings:

> *Input quality is one of perhaps many factors that contribute to productivity dispersion.* [...] *our results suggest that productivity mostly represents some attribute of a firm that cannot easily be bought and sold on the market for inputs. Possibilities include management quality, business strategy, the appropriate use of new technologies, and heterogeneous production technologies.* (Fox and Smeets, 2011)

A slightly different literature is more focussed on task assignment and managerial positions and practices. Early works, like Rosen, 1982, study the allocation of workers to positions (divided in production, supervision and management). Costa, 1988 investigates the assignment of managerial tasks in a two period framework. More recently, Lazear, Shaw, and Stanton, 2015 put the emphasis on the positive impact of managerial quality on workers' productivity. Adhvaryu, Kala, and Nyshadham, 2019 study reassignment of workers to tasks (operated by managers) in case of productivity shocks. Bloom et al., 2019 analyse Census Bureau survey data, showing the positive

effect on firm performances (measured in different ways, like productivity and profitability) of structured manager practices.

Yet another perspective is to look at employers employees matching. It is important to consider how firms and workers should be matched in the first place (Lazear and Oyer, 2007), and also the effects of team working and how workers should be matched with each other (Lazear and Oyer, 2007. These problems are generally known as *sorting* problems. Eeckhout, 2018 gives an extensive and rigorous review of these models for labour markets.

Summarizing some of the main points arising from the literature, we observe that:

- human capital quality matters in determining productivity.
- How employers should assign workers to tasks is an interesting problem, and has non trivial solutions. Also, this has a role in determining firms' productivity.
- Dispersion in firms' productivity has many different drivers, some of which depends on unobservable factors.

In this study, we adopt a supervised learning approach in determining allocations of employees to tasks (which we will refer to as *allocation rule*). As far as the author knows, this approach has not been used yet in this context. We are interested in inferring an allocation rule and in measuring the quality of firms' assignments. More precisely, we want to asses how well a firm assigns its workforce to the needed tasks. Our approach is model-free (*not assumptions-free*).

The methodology is briefly described as follows. Using data on matched employers and employees, we select a subset of firms and try to infer an *allocation rule* (a function/mapping from observables to tasks) from this subset. The inferred rule would then be used as benchmark in order to measure the extent by which other firms depart from this optimal assignment. Assuming the validity of such a rule, this measure could be useful to capture and quantify the effects on firms' productivity (but potentially also other performance indicators) of unobservable characteristics that relates to work organization (such as those mentioned in Fox and Smeets, 2011). Also, the inferred rule could be used to suggest possible assignment or reallocation for firms' workforce. This methodology has the advantage of being model-free and data driven. In what follows, we will discuss the proposed methodology and the underlying assumptions in details.

3.2 Predicting Job Allocation

In this section, we illustrate the conceptual framework of allocation of workers to tasks as intended in our study and we will also provide some intuitions on the validity of this approach.

3.2.1 Job Assignment Rule framework

In this section, we illustrate how we frame the problem of estimating employees-to-tasks allocation as a supervised learning problem. For convenience, we first recollect the notation we use in the this chapter. The second part of this section introduces the general framework.

General notation and setup

Let us fix the notation used in the following. In doing so, we assume the existence of the objects specified.

$(\Omega, \mathcal{A}, P)$ An appropriate underlying probability space. Where Ω is the sample space, ω are elements in Ω and $\mathcal{A}$ is a σ-algebra on the sample space. P is the probability measure defined on the measurable space $\big(\Omega, \mathcal{A}\big)$.

$\mathcal{X}$ set of workers/employees. An employee (element of the set) will be denoted by x, a vector of employee's features (e.g. education, age, sex, ...); $x = \left[x^{(1)}, x^{(2)}, \ldots, x^{(d_1)}\right]^T$ (d_1 fixed integer). An employee is perfectly identified with his/her characteristics. x_i denote employee i. Random variable on Ω, taking value in $\mathcal{X}$ are denoted with X.

$\mathcal{Z}$ set of firms. A firm (element of the set) will be denoted by z, a vector of firm's features (e.g. productivity, total assets, size, age ...); $z = \left[z^{(1)}, z^{(2)}, \ldots, z^{(d_2)}\right]^T$ (d_2 fixed integer). Firm k features are denoted as z_k. Random variable on Ω, taking value in $\mathcal{Z}$ are denoted with Z.

$\mathcal{Y}$ A set of possible task/jobs/positions in the market. $\mathcal{Y} = \{1, 2, \ldots, |\mathcal{Y}|\}$ is assumed here to be a discrete set with $|\mathcal{Y}|$ elements. An element of this set is typically denoted with j (job). Random variable on Ω, taking value in $\mathcal{Y}$ are denoted with Y; realization of Y are denoted by y.

S Set of job allocation rules. These are mappings s of the type $s : \mathcal{X} \times \mathcal{Z} \to \mathcal{Y}$. $f \in S$ denotes the optimal assignment rule (see next section). f_j for $j = 1, \ldots, |\mathcal{Y}|$ model the conditional probabilities: $P(Y = j|X, Z) = f_j(X, Z)$, where, with a slight abuse of notation, P also indicates the joint probability induced by the random variables X and Z on the space $\big(\Omega, \mathcal{F}, P\big)$.

top-firms The set of firms assumed to make use of an optimal workforce allocation rule, f. We also refer at this set of firms as the *learning set*. Typically, these firms are chosen to satisfy some criterion.

j'; j^* Will indicate the observed job allocation and the optimal job allocation (according to the optimal allocation rule f). Additional subscripts will indicate the individual and the firm (in this order): e.g. $j'_{i,k}$ indicates the observed task assigned to employee x_i in firm z_k. Note that possible jobs are $j = 1, \ldots, |\mathcal{Y}|$.

p'; p^* Optimal allocation rule's conditional probabilities (given worker characteristics and firms characteristics) for observed allocation j' and optimal allocation j^*. E.g.:

$$p^* := P(Y = j^*|X, Z).$$

Additional subscripts will indicate the individual and the firm (in this order). That is $p^*_{i,k} = P(Y = j^*|x_i, z_k)$.

Job allocation as a classification task

Consider the following informal argument. A firm z_k, needs to employ a resource/worker x_i. The question is: given firm characteristics z_k and employee's observable characteristics x_i, to which task would employee i be most suited for? We would like to be able to assign a task to individual x_i in firm z_k to satisfy any given optimality criterion.

Remark 3.2.1. *We are interested in allocating pairs of (x_i, z_k) to tasks, jobs or positions. Precisely, we want to find the best position for individual i working in firm k in order to satisfy some optimality criterion. Moreover, for the pair employee-firm (x_i, z_k), the choice regarding the allocation of employee i is taken by the employer, i.e. firm k.*

Conceptually, this is very different from the employees-employees matching or employees-firms matching, as framed in the literature on sorting problems (Eeckhout, 2018).

Note that it may be desirable to include characteristics for both firm and individual in the allocation decision. For example, firms in different industries might want to allocate differently similar individuals, i.e. $x_{i_1} \approx x_{i_2}$, because some tasks may be less relevant in one firm than the other. Moreover, two different individuals with different set of characteristics may well be allocated to different tasks by similar firms, i. e. $z_{k_1} \approx z_{k_2}$. This is formalized as follows.

For jobs $j = \{1, \ldots, |\mathcal{Y}|\}$ in the economy, we assume there are functions f_j^0 such that $f_j^0 : \mathcal{X} \times \mathcal{Z} \to \mathbb{R}$. These functions give for each pair of individuals and firms characteristics a score for each job. Also, we assume that these functions respond to some optimality criterion. That is, if $f_j^0(x_i, z_k) > f_{j'}^0(x_i, z_k)$, then the individual is taken to be better suited for job j rather than j', because she/he is, for example, more productive doing j. This defines the best possible assignment as $f^0(X, Z) := \arg\max_j f_j^0(X, Z)$. Finally, we assume that f^0 gives rise to the random variable Y:

$$Y = \arg\max_j \left\{ f_j^0(X, Z) + \sigma_j \right\}, \tag{3.1}$$

where $\{\sigma_j\}$ are suitable random variables on the same probability space $(\Omega, \mathcal{F}, P)$. Here, σ_j's capture the noise in the allocation. These are meant to take into account some variability with respect to a deterministic allocation rule (once conditioned on values of X and Z). Examples of this variability are: the employee expressed a strong personal taste for a particular job so as to influence employer's decision; the optimal task may be unavailable at the moment of hiring; misreport in evaluating candidate characteristics etc.

In practice f^0 is not known. However, we observe realizations of Y. From these, we would like to retrieve f^0 or at least an allocation rule that mimics it. Furthermore, we are not only interested in estimating the optimal allocation, but also the "confidence" with which the assignment was done. In other words, was the assignment a clear cut? Are there other equally valid allocations? In principle, there may be some x and z for which the indication for a particular job given by $f_j^0(x, z)$ is so strong that it dominates any possible noise $\{\sigma_j\}$, and some others x and z for which this indication is weak, so that the optimal allocation can be easily off-set even by a minimal amount of noise. For example, employees not particularly suited for any task would be more easily assigned to any vacancy. We would like to identify those cases in order to assess the extent to which employers detach from the gold standard given by f^0.

One way to assess this is by modelling the conditional probability of job allocation given the observables, i.e. $P(Y|X,Z)$:

$$P(Y_i = j|x_i, z_k) =: f_j(x_i, z_k). \tag{3.2}$$

In fact, we expect that the higher the adequacy of the employee x_i in firm z_k for job j, the higher would be $f_j(x_i, z_k)$. To see this, imagine $f_j^0(x_i, z_k) = \infty > f_{j'}^0(x_i, z_k)$, for $j \neq j'$ and $P(|\sigma_j| < \infty) = 1$ for all j, then $f_j(x_i, z_k) = 1$. These conditional probabilities will also be used to define the *optimal allocation rule*, which is our target rule:

$$j^* = f(x_i, z_k) := \underset{j\in\{1,\dots,|\mathcal{Y}|\}}{\arg\max}\ f_j(x_i, z_k). \tag{3.3}$$

This gives us a way to indirectly evaluate choices that would be made according to f^0. In fact, were there no noise at all, Y would perfectly resemble choices made according to f^0. When the noise variation is small, which may be a reasonable assumption, Y closely reflects f^0.

Remark 3.2.2. *We assume that there is a systematic, deterministic way to assign employees to jobs, upon knowing relevant characteristics, in order to satisfy some optimality criterion. This is expressed as f^0. It is unrealistic that in real world we could ever observe such a function. At most, we may hope to observe what may be regarded as sensible choices in workforce allocation, based on f^0. These choices (being human choices) are affected by some variability and we will indicates them via f. Then, we are interested in retrieving this allocation rule f, that is different from f^0. However, the two will be close when the noise $\{\sigma\}$ is relatively small. Therefore, we are interested in f.*

Knowing f_j, and thus f, we can then evaluate actual task assignments made by firms as follows. Based on firm's observed characteristics z'_k, worker's characteristics x'_i, and the actual job in which x'_i is employed, $(y_{i,k} = j')$, consider

$$\begin{aligned}
&j^* = f(x'_i, z'_k) = \underset{j\in\{1,\dots,|\mathcal{Y}|\}}{\arg\max}\ f_j(x'_i, z'_k);\\
&f_{j*}(x'_i, z'_k) = \underset{j\in\{1,\dots,|\mathcal{Y}|\}}{\max}\ f_j(x'_i, z'_k);\\
&j' = \text{actual task assignment for } (x'_i, z'_k);\\
&f_{j'}(x'_i, z'_k) = \text{conditional probability for the actual assignment } j'.
\end{aligned}$$

Then, $\mathbb{1}\{j^* = j'\}$ answers to the question whether the optimal task allocation was selected by the employer, and $f_{j*}(x'_i, z'_k) - f_{j'}(x'_i, z'_k)$ returns a measure of how far apart is the actual choice j' from what would have been optimal the optimal j^* according to the allocation rule f.

In practice, the function f and the implied f_j's, our targets, are not available. We propose to estimate them from data. Repeating ourselves, we refer to f as the optimal allocation rule; also f_j's are optimal, where the optimality is in the sense described above. Note that the problem of estimating f, can be framed exactly via risk minimization as treated in Subsection 1.3.2. Once a loss function L is defined, we can proceed to the estimate of f using observed data $\{(y, x, z)\}$ by loss minimization. This was described in (1.13). Moreover, in Section 3.4 we give a review of the methodologies we use to tackle this estimation problem.

One final subtlety needs to be taken into account before moving on. Up to now, we assumed that Y is based on f^0. However, in reality it is way more plausible that there are multiple rules, s^0, by which firms allocate workers. This generates multiple *different* random variables similar to Y. Let us call as *top-firms* the population of firms that base their workforce allocation on the optimal rule f^0. *Non top-firms* denotes firms using other allocation rules, which we assume not to be optimal in the sense that workforce allocation based on these rules do not satisfy the desired optimality criterion. The two populations are denoted with $\mathcal{T}$ and $\mathcal{N}$ respectively. Note that the conditional probabilities we are willing to estimate (of the type (3.2)), vary according to the population considered. Because of the different decision processes of top-firms and non top-firms, we have that the true relationship $P(X, Z, Y)$ is different in top-firms set $\mathcal{T}$ and the non top-firms set $\mathcal{N}$. This is because the random variable Y is different in the two populations. Top-firms' f^0 give rise to the allocations $Y_{\mathcal{T}}$; non top-firms' s^0's (there may be differences across them) lead to the allocations $Y_{\mathcal{N}}$. Thus, we write:

$$P(Y_{\mathcal{T}} = j|X, Z) = f_j(X, Z) \neq P(Y_{\mathcal{N}} = j|X, Z) = s_j(X, Z).$$

Now, before proceeding with estimation, is important to filter out $Y_{\mathcal{T}}$, as this will allow us to estimate and retrieve the optimal allocation rule f. Was this not possible, our estimates will make use of a variable Y that reflects multiple non-optimal decision rules. This would lead to a not well defined target and to an estimated allocation rule which does not reflect the optimal one.

Typically, we will assume that such a split in top and non top firms is possible. So that, if not explicitly stated, we drop the subscripts on Y, and write $P(Y_{\mathcal{T}} = j|X, Z) = P(Y = j|X, Z) = f_j(X, Z)$.

This splitting, basically consists in the definition of training data. We mentioned previously that the functions f's optimize some criterion. Assume that, *ceteris paribus*, firms productivity[1] has among its drivers the ability of the employer to properly assign its workforce to tasks (e.g. Lazear and Oyer, 2007). We could then assume that firms exhibiting the highest productivity levels (among other comparable firms) are those that have an optimal employees-to-tasks allocation strategy. Once identified, we use data on these *top-firms* to learn how they allocate workers. We do this on employers-employees matched data, by estimating a classification model using top-firms' characteristics and their employees' characteristics as features variables, and the observed job/task assignments as the outcome variable (see Subsection 3.5.1. The classification model would then return the estimates of the target f and f_j's, which we call $\hat{f}$ and $\hat{f}_j$'s.

Finally note that the this approach is essentially model-free (see also discussion in the next section) in the sense that we do not need to be explicit on how the assignment affects productivity nor specific assumption on the form of f^0 (for example compare with Eiselt and Marianov, 2008).

The estimation of an optimal allocation rule f is a trivial task and for this reason we would like to check that the estimates we retrieve are actually capturing a systematic optimal workforce allocation rule. This is unfortunately hard to do in classical ways. In fact, measures to evaluate the goodness of fit as described in Subsection 1.3.2 cannot be used directly. Even if we were

[1] In principle, replacing productivity with other firms' performance measures does not affect this reasoning.

able to achieve high scores for $\hat{f}$ in the top-firms set, where we estimate it, this would not automatically allow to assess the ability of $\hat{f}$ to capture the optimal rule f. This is because the filtering of top-firms from non top-firms is endogenous to the analysis. For this reason there is no way to assess in a rigorous way whether or not the top-firms set actually contains only firms adopting the allocation rule f. Furthermore, these measures can not be used to assess the performances of $\hat{f}$ on non top-firms set either. High scores for the measures would indicate that f actually generalizes well to this set. However, we do not want it do generalize well on non top-firms! Non top-firms are assumed to use other allocation rule s, so that we would like to see deviations from f in this set.

Nonetheless, it is still possible to partially check the efficacy of the estimated $\hat{f}$ via its connection with the criterion for which f is an optimal allocation rule. Say we are interested in an allocation rule that maximizes productivity. Then, let productivity be a non constant function of workforce allocation and other observable, O, and unobservable U characteristics:

$$\text{productivity} = g(f(X, Z), U, O) \tag{3.4}$$

(this writing is legitimated by Economic intuition (e.g. Lazear and Oyer, 2007, Fox and Smeets, 2011). Then, as discussed in this section, we would expect from an optimal allocation rule that $\forall\ x,\ z,\ o,\ u$, and for all $j \neq j'$

$$f_j^0(x, z) > f_{j'}^0(x, z) \implies g(j, o, u) > g(j', o, u). \tag{3.5}$$

If this is true, then we can check the estimated $\hat{f}$ *ex post* in the following way.

Remark 3.2.3. *By construction, we expect to find a negative relationship between productivity and the extent to which observed allocations deviate from the optimal allocations given by f. If $\hat{f}$ is estimating f we expect that deviations from it are negatively related to productivity in the non top-firm set.*

Let us visualize this with an example. Imagine $x,\ z,\ o$ and u to be fixed. Also assume that $j \in \{1, 2, 3\} = \mathcal{Y}$ and:

$$\tilde{f}^0(x, z) = [5, 4, 0]\,; \quad \forall j,\ \sigma_j(z) = \begin{cases} 1 & w.p.\ \frac{1}{2} \\ -1 & w.p.\ \frac{1}{2} \end{cases}.$$

In this case, it is easy to see that:

$$f_1(x, z) = \frac{5}{8}; \quad f_2(x, z) = \frac{3}{8}; \quad f_3(x, z) = 0.$$

Top-firms using f choose job 1 (no deviation observed). Other firms may choose any of the three jobs. Note that due to (3.5) we have that

$$g(1, o, u) > g(2, o, u) > g(3, o, u). \tag{3.6}$$

Evaluating the deviation from the optimal allocation $j^* = f(x, z)$ as suggested above (compare

with Remark 3.2.2 and discussion following it), $d(j^*, j) = f_{j*}(x, z) - f_j(x, z)$ and

$$d(j^*, 1) = 0, \quad d(j^*, 2) = \frac{2}{8}, \quad d(j^*, 3) = \frac{5}{8}. \tag{3.7}$$

Finally, comparing (3.6) with (3.7) we see (for non top-firms) that at increasing deviation from f we have decreasing productivity levels, as stated in Remark 3.2.3.

Were $\hat{f}$ a good estimate of f, we should observe, in the non top-firms set, the same negative relation between deviations from it and productivity. We can check this on the data (even if we do in term of positive relationship by considering degree of accordance with $\hat{f}$ and productivity).

Thus, ultimately we need an estimator of f and measures that can precisely state deviation from an optimal allocation rule. For the former we use methodologies described in Section 3.4 and for the latter we now move to the construction of such measures in Section 3.3. In addition, these measures can be directly use to account for the amount of residual productivity dispersion due to the workforce allocation strategy.

3.3 Measuring Job Allocation Quality (JAQ)

In this section we propose to measure the ability of an employer to assign its employees to tasks. We will build measures that evaluate single employees' allocations and measures to evaluate the overall ability of a firm in allocating its workforce. First, we briefly introduce the quantities that will be used for the evaluation and then in the following and in and Subsection 3.3.1 we will describe the two type of measures.

To ease the discussion, we will assume that the optimal allocation rule f is known and we disregard estimation issues. Nonetheless, if f has to be estimated, say $\hat{f}$ is the estimated value, the following discussion similarly follows by simply replacing f with its estimated counterpart, $\hat{f}$.

Let us first introduce the quantities we are interested in. We want to define measures to evaluate the quality of job assignments, j', based on observed realizations of X and Z, i.e. $x \in \mathcal{X}$ and $z \in \mathcal{Z}$. Recall that f is the optimal allocation rule and f_j are the conditional probabilities of job assignment (see (3.3) and (3.2)). The assignment j' need not to be done according to f in this case. Firms may use a different s, which may coincide with f but need not to. Consider an observed sample for (X, Z, Y) (here Y is not filtered out in the sense exposed at the end of Subsection 3.2.1). A single observation in the data is identified by the individual, x_i, and the firm where he/she is employed z_k.[2] Thus, the indexes i and k perfectly identify the observation. Since, typically a firm employs more than one person, say we observe z_k where $k = 1, \dots, K$, and K is the number of unique firms and, associated with each k, we observe I_k employees x_{i_k} where $i_k = 1, \dots, I_k$, where I_k is equal to the number of individuals employed at firm k.[3] Note that the total number of observations is given by $\sum_k I_k$, and typically $K \ll \sum_k I_k$. Then, let $(y_{i,k} \equiv j'_{i,k})$

[2] We typically have also a temporal observation, so that more precisely a single observation is perfectly identified with an individual index i, a firm index k and a time index t. To ease notation, we disregard the time dimension for the moment being. The analysis can be extend straightforwardly by repeating it for each value of the time variable (e.g. by years), adding a time index.

[3] Such a scenario could be obtained by considering a discrete Z, and a continuous X, independent one from another. Or more generally, denote Z_1 and Z_2 two random variables distributed as Z, we require $P((X, Z_1) = (X, Z_2)|X) > 0$.

be the $\{i,k\}$-th sample realization of Y, i.e. be the observed task assigned to individual i in firm k. If firm z_k allocates employees according to some $s^{(k)}$, we define the following quantities:

$$\begin{aligned} j'_{i,k} &:= s^{(k)}(x_i, z_k); & p'_{i,k} &:= f_{j'_{i,k}}(x_i, z_k); \\ j^*_{i,k} &:= f(x_i, z_k) = \underset{j\in\{1,\dots,|\mathcal{Y}|\}}{\arg\max}\ f_j(x_i, z_k); & p^*_{i,k} &:= f_{j^*_{i,k}}(x_i, z_k). \end{aligned} \tag{3.8}$$

These are: (on the first line) the observed job allocation for employee i in firm k and the conditional probability of the observed job allocation according to the optimal allocation rule f; (on the second line) the optimal job allocation for employee i in firm k and the conditional probability of the optimal job allocation according to the optimal allocation rule f.[4] Note that the p's are probabilities and thus take value in $[0,1]$. Moreover, from their definition (see (3.2)):

$$\sum_{j=1}^{|\mathcal{Y}|} f_j(x_i, z_k) = p'_{i,k} + p^*_{i,k} + \sum_{j\notin\{j'_{i,k}, j^*_{i,k}\}} f_j(x_i, z_k) = 1.$$

Referring to these quantities, we build several measures of job quality allocation that will be described in the next sections. We will define measures to evaluate the allocation of a single employee and to evaluate the overall quality in firms' workforce allocation. Before moving on, let us stress two important points.

Remark 3.3.1. *Evaluations are being done according to the optimal allocation rule: $p^*_{i,k}$ is the job showing the highest conditional probability according to the optimal allocation rule; $p'_{i,k}$ is the conditional probability of the observed job according to the optimal allocation rule, and gives a measure of the suitability of the employee to the job she/he was assigned to. Thus, the term $p^*_{i,k} - p'_{i,k}$ captures the distance between the optimal allocation and the current one, judging from the perspective of an employer who allocates optimally its employees.*

Consider the cases in which $p^*_{i,k} \approx p'_{i,k}$. Excluding cases with excessive noise, this happens when the observed job j' correspond to the job that would have been assigned according to the optimal rule, j^*. Such cases include no clear allocation for the employee, i.e. $f_j(x_i, z_k) \approx f_l(x_i, z_k)$ for all $l, j = 1, \dots, |\mathcal{Y}|$. [5]

Remark 3.3.2. *f is based on all jobs in the Economy, $\{1, \dots, |\mathcal{Y}|\}$. In practice it may happen that $j^*_{i,k} = f(x_i, z_k)$ may be a job that is not required by the firm k. That is, an ideal employer who optimally allocates employees would suggest an allocation for an employee that is not needed by the firm. We will take a task as not needed if we do not observe for the firm any employee allocated to task . This may well be the case when we compare two different populations of firms, top-firms and non top-firms (see Section 3.2.1). Intuitively, from an Economics perspective, firms for which the suggested job is never observed should be interpreted as having hired the wrong individual, who do not fit their needs, at least in the short run. In the long run, based on evidence from other top performing firms, these firms may decide to open new positions as the one suggested, if they find it beneficial.*

[4] In practice, only j' is an observed quantity. All the others are unknown since we do not get to observe f. These are estimated via $\hat{f}$.

[5] Cases in which we do not observe any clear allocation could happen when the employee is either not particularly skilled or so talented that whatever job would be suitable for him/her.

Employee-wise JAQ

Based on the above considerations, we can now define the employee-wise job assignment quality (EJAQ) measure as:

$$\text{EJAQ}_{i,k} := 1 - (p^*_{i,k} - p'_{i,k}). \tag{3.9}$$

This is an employee-wise measure in the sense that it is computed for every employee in every firm. EJAQ ranges in $[0, 1]$, and measures the distance from the optimal allocation in term of conditional probabilities.

EJAQ is maximal when $p^*_{i,k} = p'_{i,k}$, i.e. when the employer correctly allocated the employee according to f . EJAQ is minimal when there is high polarization (i.e. the individual has a very high conditional probability to be assigned to a job) and the observed job does not correspond to the optimal allocation. This may happen when the employee is manifestly a perfect fit for a job ($p^*_{i,k} \approx 1$, and by consequence $p'_{i,k} \approx 0$, if $j' \neq j^*$), but was misallocated to a different job. Thus, EJAQ is increasing in allocation quality (according to the optimal rule f).

Based on considerations in Remark 3.3.2, note that $p^*_{i,k}$ in (3.9) may well be a non open position in the firm under evaluation. Hence EJAQ captures not only the ability of the employer to allocate the worker, but also its ability to pick the right resource in the market. That is, if the employee is a clear fit for job j^*, but this position is not required by the firm, EJAQ will penalize not only for the misallocation (note j^* could never be observed in the firm, so that j' must be different from j^*), but also for having selected a worker in the market who clearly has other specializations and skills. In fact, we are not considering the optimal jobs among the positions available in the firm, but the most suitable job among all the possible positions in the market. This is in line with the idea of evaluating whether or not we observe the right (wo)man for the position (see Section 3.1).

3.3.1 Firm-wise JAQ

The EJAQ (3.9) introduced above does not automatically evaluate the overall performances of an employer in workforce allocation. For this type of statements, we need to aggregate somehow the quality of each employee-task assignment observed in the firm. In fact, there are many possible way of doing this. In this section we will propose four measure to evaluate firms' overall assignment quality and will also comment on the difference perspectives implied by each of them. We call these measures firm-wise job assignment quality measures (FJAQ). We recall that for every firm, indexed by k, we observe I_k employees (Section 3.3).

In order to evaluate overall workforce allocation, one natural approach is to simply average quality of the employee-task assignments observed from the employer. This defines a first FJAQ measure as (refer to (3.8)):

$$\text{FJAQ1}_k := \frac{\sum_{i=1}^{I_k} p'_{i,k}}{I_k}. \tag{3.10}$$

FJAQ1 has at the numerator the sum of the conditional probabilities for the observed allocated jobs, $p'_{i,k}$, divided by the total number of employees in the firm, I_k. It takes value in $[0, 1]$. The probabilities at the numerator are higher when the firm hires highly specialized[6] employees and

[6]Here we use the term "specialized" meaning a worker with characteristics that makes him/her almost a perfect

allocates them to the job for which they are specialized. Indeed, if this is the case, then we would have $p'_{i,k} = p^*_{i,k} \approx 1$ (recall that the conditional probability and the "specialization" of the employee are always evaluated according to f). Thus, in order to do well, the firm must hire perfectly specialized employees, for positions needed in the firm, and each worker must be correctly allocated to his/her most suitable position. Note that at the denominator we have the number of employees in the firm. Visualizing this as $\sum_{i=1}^{I_k} 1$, we see that this measures penalizes by comparing firm's results with that of an employer hiring perfectly specialized employees (according to f; $p^*_{i,k} = 1$, $i = 1, \dots, I_k$) and allocating them correctly.

FJAQ1 is harshly penalizing for misallocation and poor selection of employees. The penalization of hirings is even stronger than that of EJAQ, because we are comparing it with ideal employees with $p^*_{i,k} = 1$ (these might not even exist in the market).

FJAQ1 is not really taking into account a comparison with the benchmark rule. In fact, if all p'_j are approximately equal and $|\mathcal{Y}|$ is large, this measure results in a low value, even if we know that in this case any allocation should be regarded as optimal (see Remark 3.3.1). This leads us to the next measure.

To overcome this issue we can use the EJAQ (3.9) and simply take its average across employee-task allocations in the firm. This defines the FJAQ4 measure:

$$\text{FJAQ4}_k := \frac{1}{I_k} \sum_{i=1}^{I_k} \text{EJAQ}_{i,k} = \frac{1}{I_k} \sum_{i=1}^{I_k} \left(1 - (p^*_{i,k} - p'_{i,k})\right). \tag{3.11}$$

FJAQ4 is also penalizing for bad hirings, because the EJAQ's consider penalization with respect to the most suitable job for the employee among all available jobs in the Economy, and not with respect to those available in the firm. Nonetheless, this penalization is less severe than FJAQ1 (3.10) since it does not compare the actual assignment to ideal case of perfectly specialized hirings. It is likely more easy to see this by looking at the FJAQ1 as a special case of FJAQ4, obtained by replacing in $p^*_{i,k}$ in (3.11) with 1. In addition, this measure is more in line with the idea of evaluating employee-task assignment not in absolute terms, but in relative to a benchmark, i.e. the optimal feasible alternative.

One possible critique to both FJAQ measures could be as follows. In the short run, positions available at the firm and employees are fixed. Thus, we may be willing to assess whether the firms achieves the best possible allocation under f with employees and positions available. FJAQ1 and FJAQ4 can not answer these question. First, in this case, we should not use as a comparison j^* if this position is not available in the firm. Second, we can not simply evaluate employee-wise allocation in this case. The employee-wise measures are implicitly assuming that each employee should do the job she/he is the best fit for. However, this might imply that all employees should be assigned to the same task in the firm! We will address this issue in the following section.

3.3.2 Firm-wise JAQ for employees-to-tasks assignment problems

In the discussion above, we did not really consider restriction on possible job allocations. That is, whenever we compared the observed allocation j' to the optimal one, j^*, we assumed the latter to possibly be any of the jobs observed in the market (see Remark 3.3.2). Nonetheless, even if

fit for a job in the firm. This is assessed on the base of f.

the function f prescribes job $j^*_{i,k}$, for employee x_i in firm z_k, this solution should not be taken into account if firm z_k does not require task j^*.[7] To see why this is an important consideration, think at a situation in which an employee is perfectly suited for a position not needed in the firm, while for all the positions available in the firm, $f_j(x_i, z_k) = 0$. If we were not to consider this fact, we would say that the firm allocated very poorly its employee, while in a sense, the firm allocated the employee well, since, according to f, we are indifferent between any of the available positions.

Furthermore, as anticipated in Subsection 3.3.1, consider the case in which all the employees in the firm are best suited for the same job. Clearly, the firm might need multiple tasks to be done, and even if all of its employees would be allocated to the same job according to f, this is not a feasible solution. Moreover, we should take care of comparative advantages in moving worker in different allocations: it might be not optimal to assign every employee to the most suitable job, but maybe to his/her second best, to achieve an overall better allocation.

This is an old, well known problem. Kuhn, 1955 provides a very clear introduction and illustrates the popular "Hungarian method". There are several generalizations of the problem, and it is also of interest in several fields (e.g. in Labor Economics see Crawford and Knoer, 1981). Here we deal with a basic formulation described as follows. Considering firm k, we need to allocate I_k employees to jobs. Job types are constrained to those observed in the firm. Let $m \leqslant |\mathcal{Y}|$ be the number of unique jobs. Each worker suitability for job j is evaluated by the conditional probability $f_j(x_i, z_k)$. Each employee can be allocated to only one job. The goal is to maximize the overall suitability for employee-task allocations.

Consider the following example. In the Economy, there are a total of 6 jobs: $|\mathcal{Y}| = 6$. For firm k, we observe $I_k = 6$ employees allocated to the j' tasks in Table 3.1.

Table 3.1: Observed job allocations.

employees' id:	id1	id2	id3	id4	id5	id6
observed job (j'):	5	3	2	2	2	4

Consider now the matrix of conditional probabilities, $C = (c_{i,j}) = f_j(x_i, z_k)$ (z_k is fixed), shown in Table 3.2. As argued above, we should not consider job 1 and job 6 because they are never observed in the firm, thus we assume those positions are not required and note that that the firm requires three employees in job 2: in this case $I_k = 6$ and $m = 4$. Then, we modify the

[7] We recall that to decide which tasks or jobs are needed by firm k, we simply assume that the jobs that we observe in the firm are the ones that are needed.

Table 3.2: Conditional Probability matrix. ($'$) indicates the observed allocation and (*) indicates the optimal allocation according to f; these coincides in some cases.

	job 1	job 2	job 3	job 4	job 5	job 6
id1	0	0	0	0.5*	0.5′*	0
id2	0.1	0.5*	0.4′	0	0	0
id3	0.4*	0.3′	0.1	0.1	0.1	0
id4	0.2	0.1′	0.1	0.1	0.1	0.4*
id5	0.2	0.6′*	0.1	0.1	0	0
id6	0	0.7*	0	0.3′	0	0

Table 3.3: Assignment Problem scores. Entries are $c(i,j) = f_{\text{i,j}}(\text{id } i, \text{job } j)$.

	job 2	job 2	job 2	job 3	job 4	job 5
id1	0	0	0	0	0.5	0.5
id2	0.5	0.5	0.5	0.4	0	0
id3	0.3	0.3	0.3	0.1	0.1	0.1
id4	0.1	0.1	0.1	0.1	0.1	0.1
id5	0.6	0.6	0.6	0.1	0.1	0
id6	0.7	0.7	0.7	0	0.3	0

Table 3.4: Solution to the assignment problem optimal job allocation

employees' id:	id1	id2	id3	id4	id5	id6
optimal job (j^a):	5	3	2	4	2	2

matrix C with $\tilde{C}$ as shown in Table 3.3.[8] That is, we repeat each job column as many times as we observe the corresponding job was allocated to an employee. Matrix $\tilde{C}$ can be seen as a matrix of employees-jobs scores. Then, we want to select for each row i a column j, without repetition of columns, such that the sum of the so selected $\tilde{c}_{i,j}$ is maximal. This is the assignment problem at hand and it can be formulated in term of integer linear programming (Steiglitz, 1998). Let E be the set of employees defining the rows of $\tilde{C}$, and J the set of jobs defining the columns of $\tilde{C}$. For $i \in E$ and $j \in J$ define

$$e_{i,j} = \begin{cases} 1 & \text{if employee } i \text{ is allocated to job } j, \\ 0 & \text{otherwise.} \end{cases}$$

Then, we can write the assignment problem as the following linear programming:[9]

$$\max_{e_{i,j}} \quad \sum_{i \in E, j \in J} \tilde{c}_{i,j} e_{i,j},$$

$$\begin{aligned} \text{s.t.:} \quad & \sum_{i \in E} e_{i,j} = 1, \ \forall j \in J, \\ & \sum_{j \in J} e_{i,j} = 1, \ \forall i \in E, \\ & 0 \leqslant e_{i,j} \leqslant 1, \ \forall i \in E, \ j \in J. \end{aligned}$$

Solving this problem for the example above, we obtain the allocations presented in Table 3.4. Note that this solution does not prescribe trivial allocations. Not all the employees are assigned to their optimal job j^* (refer to Table 3.2) nor to the job they are most suitable for once deleting the unavailable positions in the firm. Moreover, the solution is different from the observed allocation. Finally, by summing the conditional probabilities for the observed allocations j' (Table 3.1) and that just derived (Table 3.4) we obtain respectively: 2.2 and 2.6.

[8] A formal definition is as follows: $\tilde{C} = (\tilde{c}_{i,j}) = f_j(x_i, z_k)$ where

$$i = 1, \ldots, I_k, \quad j = \{1\}^{\sum_{i=1}^{I_k} \mathbb{1}\{j'_{i,k}=1\}}, \ldots, \{|\mathcal{Y}|\}^{\sum_{i=1}^{I_k} \mathbb{1}\{j'_{i,k}=|\mathcal{Y}|\}}.$$

[9] Note that this formulation does not exclude fractional solutions, however these solutions are not selected when the problem is solved via specific algorithms as the Hungarian method (Steiglitz, 1998).

We will refer to allocations obtained for the pair (x_i, z_k) solving firm k assignment problem as $j^a_{i,k}$, i.e. the job assigned to employee x_i in firm z_k, obtained by solving the employees-to-tasks assignment problem, by an employer using the optimal allocation rule f. The relative conditional probabilities of these jobs are denoted as

$$p^a_{i,k} := f_{j^a_{i,k}}(x_i, z_k). \tag{3.12}$$

This solution is the one that would have been chosen by an employer with allocation rule f, constrained to allocate the available employees to current open positions in the firm, in order to maximize the overall workforce allocation quality. Note that this might be considered as the optimal short run reshuffling of employees to tasks. In the long run it would be possible to either hire/fire employees or to open new positions in the firm. In this analysis, we consider the former case so that what can possibly change is just the allocation of employees.

We now proceed with the definition of the FJAQ's measures based on the solution of firms' assignment problems.

FJAQ for the firm's assignment problem

As described at the beginning of Subsection 3.3.2, in order to evaluate the overall ability of employers in workforce allocation, it may be sensible to frame the employee-to-task allocation problem as an assignment problem. This makes sense if we want to filter out the penalizations due to the hiring process and we consider a fixed set of employees and tasks.

We define two measure using the solution of firms' assignment problem. The first of these two measures is the FJAQ2, and is defined as follows:

$$\text{FJAQ2}_k := \frac{\sum_{i=1}^{I_k} p'_{i,k}}{\sum_{i=1}^{I_k} p^a_{i,k}}, \tag{3.13}$$

where p^a's are given by (3.12). This measure builds on FJAQ1 (see (3.10)). Indeed, the quantity at the numerator stays the same. The quantity at the denominator is given by the sum of conditional probabilities relative to the jobs assigned by the solution of the firm's assignment problem. FJAQ2 has a weaker penalization than FJAQ1 since it compares the observed allocation j' to the best allocation that could been achieved under f, considering workforce and available tasks constraints, j^a. Thus, a firm is not penalized for not having a specialized workforce (i.e. with polarized conditional probabilities) nor for bad hiring decisions.

The second measure we introduce is motivated by the same rationale as FJAQ2 and as a form similar to FJAQ4 (see (3.11)). It is defined as:

$$\text{FJAQ3}_k := 1 - \frac{\sum_{i=1}^{I_k} p^a_{i,k} - \sum_{i=1}^{I_k} p'_{i,k}}{I_k}. \tag{3.14}$$

This measure capture the same effects as FJAQ2 (3.13): firms are penalized for not allocating the workforce properly, while effects due to not being able to hire specialized employees are

disregarded. However, the penalization acts differently in FJAQ3, and it tends to be more generous than FJAQ2 when the workforce is overall not well suited for positions available at the firm. The latter happens when an employer is not able to hire specialized employees for the tasks required. In such cases, the optimal allocation is such that $(\sum_i j^a_{i,k}) \approx 0$. The measure is normalized with respect to the number of employees in the firm, making it comparable across firms with different size.

FJAQ3 (3.14) takes values in $[0, 1]$. It is minimal when the firm has perfectly specialized employees for every required task and does not allocate them properly. It is maximal when when the firm allocates its workforce optimally (compare with Remark 3.3.1).

Figure 3.1 (at the end of this section) gives a visual representation of FJAQ1 (3.10), FJAQ2 (3.13) and FJAQ3 (3.14) and confirms the intuitions given so far. FJAQ1 is neglecting any benchmark rule and is the one penalizing the most. FJAQ2 has a penalization for workforce allocations that is non linear with respect to the benchmark optimal allocation. FJAQ3 is more generous than FJAQ2 with not specialized workforce. Note that the graph gives only intuition for the functional forms of these measures, so that in practice the graph of FJAQ4 (3.11) will be the same as that of FJAQ3. Nonetheless, the two captures two different aspects in that FJAQ4 does not take into account solutions given by the firms' assignment problem.

It must be noted that, even if FJAQ2 (3.13) and FJAQ3 (3.13) are appealing in that they may better capture only aspects related to employers' allocation decisions, they are computationally very intensive. Consider that these require to solve an assignment problem for *every* firm in the data. For big datasets the number of unique firms may be in the order of fifty to hundred thousands firms. In addition, the complexity of each assignment problem is determined by the number of employees in the firm, which may also be very large, in the order of few thousands. Even if there are efficient algorithms to solve assignment problems, due to their numerosity this may still require unfeasible computing power to solve this problem for all the observed firm in reasonable amount of time.[10] We propose an approach to find an approximate solution to the assignment problem as described in Algorithm 3. The idea is that of splitting the firm into many smaller sub-firms. This is done by taking the pool of firm's employees, shuffling them randomly, and partitioning them in set of t employees. Then, an assignment problem is defined and solved for any sub-firm. We then join together all the solutions for the sub-firms to recover an assignment for the original firm. This procedure may be repeated multiple times and the best allocation (in term of highest sum of conditional probability) can be taken. Of course, if $t = I_k$ this correspond to solve the problem for the original firm. So that, as t and iterations increase we may expect to approximate increasingly well the solution to the original problem.

We now move to the estimation of f. In the next section, we will review the methodologies we adopt. We conclude this section with an important aspect to take into account when computing these measure with an estimation of f (refer also to Remark 3.3.1).

Remark 3.3.3. *Consider a case in which $f_j(x_i, z_k) \approx f_l(x_i, z_k)$ for all $l \neq j$. This is not a problem in general. However, when f need to be estimated, say via $\hat{f}$ these cases are rather delicate. In fact, there may be two reason why $\hat{f}_j(x_i, z_k) \approx \hat{f}_l(x_i, z_k)$ for all $l \neq j$.*

[10] For example, it is known since Munkres, 1957 that the Hungarian algorithm requires at most $(11n^3 + 12n^2 + 31n)/6$ operations. $n = I_k$ in our case.

1. *Because of his/her characteristics, the employee is indeed well suited for all type of jobs or for none in particular. Moreover, the employing firm does not have characteristics that would make any particular task more suitable for the employee.*

2. *The particular instance (x_i, z_k) is not frequently observed in the data, so that the estimator has no evidence on it and is not able to allocate it to any particular job. This is introduced by estimating f with $\hat{f}$.*

These two motivations are very different, but lead to similar results: JAQ measures for these allocations would be high, because there is basically no chance to get the allocation wrong since there is no correct allocation. If this is motivated by 1 above, this is perfectly acceptable. If it is motivated by reason 2, it is not. This problem is however lessened if the number of jobs and their diversity increases. Indeed, employees of the type implied by motivation 1 are rare, and the richer the description of the Economy (in term of jobs) the rarer should be these individuals. Thus, it is more likely that the motivation underlying equal conditional probabilities for all jobs is motivation 2.

In order to discriminate these effects, while computing JAQ's, we consider only those employees with at least a minimal amount of polarization, that is one of the conditional probabilities should exceed a threshold.[11] *That is, for some $\eta \in [0, 1]$ we consider only observations for which $p^*_{i,k} > \eta$.*

[11] To be concrete, imagine that there are 100 available tasks. A non allocable employee would exhibit $f_j \approx \frac{1}{100} = 0.01$ for all j. However, suppose we can cluster these jobs in say 5 different broad classes, of 20 jobs each. In addition, say we expect that a *factotum* employee might be expected to do well on at mos t 2 of the 5 broad classes. Than, this employee would be would be equally suitable for at most $\frac{100}{5}2 = 40$ different tasks, leading to a minimum threshold of at least $\frac{1}{40} = 0.025$. Thus, considering employees with $f_j \geqslant 0.025$ for at least one j, would make us more confident that the employee was allocated by $\hat{f}$ correctly and that the non polarized conditional probabilities are due to his/her characteristics rather than estimator flaws.

Algorithm 3: Mini Solver for Assignment Problem

Input: z_k; I_k; x_i $\forall i = 1, \dots, I_k$; $\hat{f}$; t; R.
Output: $j^a_{i,k}$, $p^a_{i,k}$ $\forall i = 1, \dots, I_k$.

$C = (c_{i,j}) \leftarrow \hat{f}_j(x_i, z_k)$ $\forall i = 1, \dots, I_k,\ j = \forall j = 1, \dots, |\mathcal{Y}|$. I.e. form the matrix of conditional probabilities for all employees in the firm
for $r \in 1, \dots, R$ **do**
  $\{\pi(1), \dots, \pi(I_k)\} \leftarrow$ random permute the set $\{1, \dots, I_k\}$
  $C' = (c'_{i,j}) \leftarrow (c(\pi(i), j))$ $\forall i = 1, \dots, I_k,\ j = \forall j = 1, \dots, |\mathcal{Y}|$. I.e. shuffle randomly the rows of C
  $bins \leftarrow \{[1, t+1), [t+1, 2t), \dots, [(I_k/t), I_k]\}$. I.e. divide I_k in bins of size t
  for $k' \in bins$ **do**
    $C_{k'} \leftarrow (c'_{i,k})$ $\forall i \in k'$. I.e. of the reshuffled matrix C' consider only the rows in bin k'
    $\tilde{C}_{k'} \leftarrow$ assignment problem relative to matrix $C_{k'}$
    $\hat{j}^a_{i,k'},\ \hat{p}^a_{i,k'} \leftarrow$ solution of $\tilde{C}_{k'}$
  $\hat{j}^r_{i,k},\ \hat{p}^r_{i,k} \leftarrow \left\{(\hat{j}^a_{i,k'})_{k'},\ (\hat{p}^a_{i,k'})_{k'}\right\}$. I.e. form the solution of the assignment problem relative to the full C by joining the solutions of the smaller problems
  $\hat{p}^r \leftarrow \sum_{(i,k)} \hat{p}^r_{i,k}$
$r^* \leftarrow \arg\max_r \hat{p}^r_r$
$j^a_{i,k},\ p^a_{i,k} \leftarrow \hat{j}^{r^*}_{i,k},\ \hat{p}^{r^*}_{i,k}$

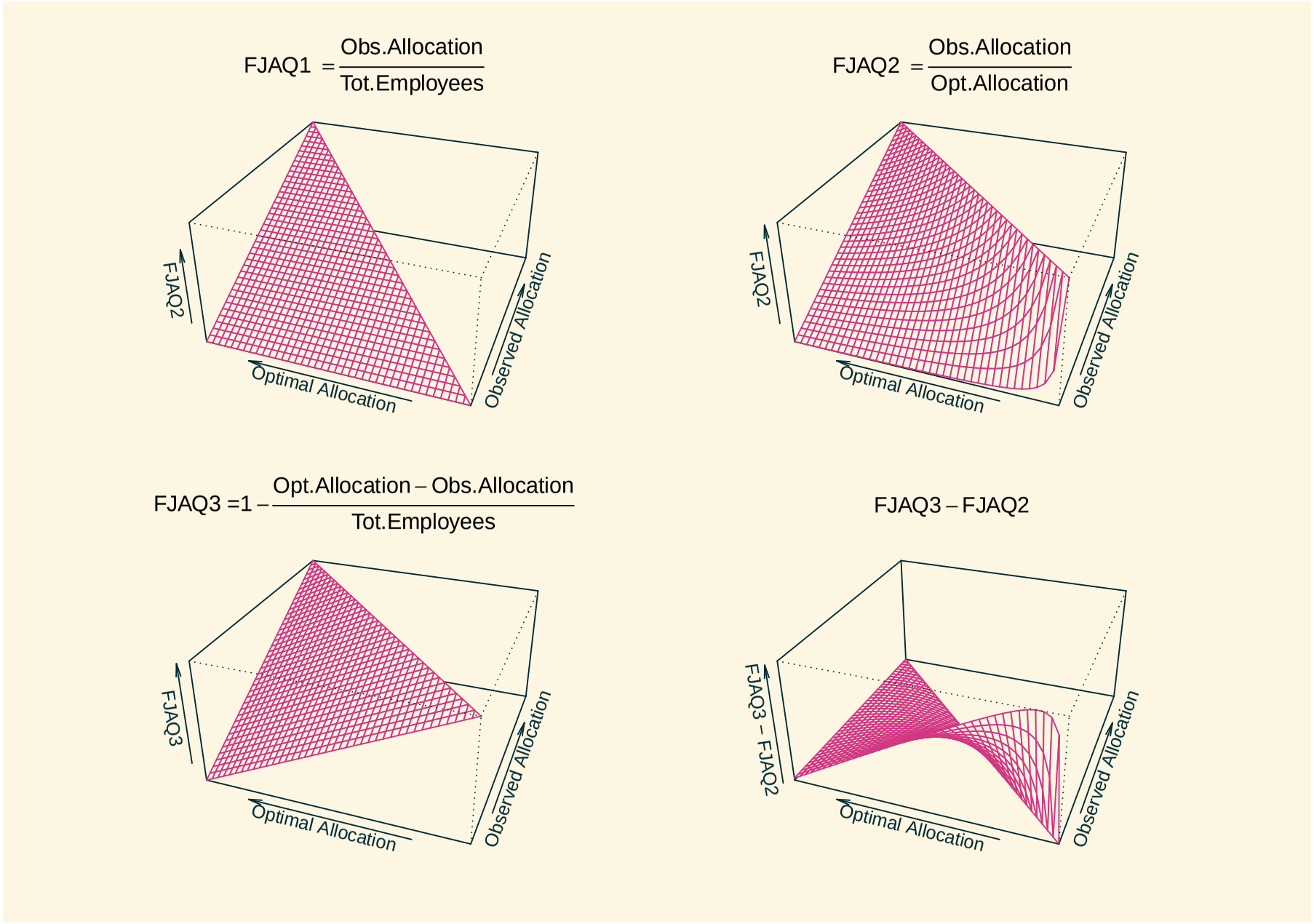


Figure 3.1: Representation of the first three FJAQ measures. Optimal Allocation is given by $\sum_{i=1}^{I_k} p^a_{i,k}$; Observed Allocation is given by $\sum_{i=1}^{I_k} p'_{i,k}$; $I_k = 1$ in these graphs. All the axes ranges from 0 to 1. Top-Left: FJAQ1 (not depending on Optimal Allocation); Top-Right: FJAQ2 (note the non linearities); Bottom-Left: FJAQ3 (linear in both argument); Bottom-Right: FJAQ3-FJAQ2 (note how FJAQ3 is higher than FJAQ2 when Optimal Allocation goes down; FJAQ2 penalizes more poor allocations).

3.4 Review of the learning algorithms

In order to estimate probabilities f_j's (see (3.2)), we will use two main approaches. The first is a classical multinomial logistic regression from Econometrics. The second approach employs machine learning methods based on trees. These are random forests and bagging. This will allow us to compare machine learning methods with standard method in applied Economic research, and assess their advantages.

In the following sections, we briefly recall the multinomial logistic model and give a more extensive treatment of the machine learning algorithms.

3.4.1 Multinomial Logit

A multinomial logistic model is a type of qualitative response model, that is, it is intended to model a categorical outcome/dependent variable. Following Amemiya, 1985, we define the model in a general formulation as follows. Let i index denote observations and let the variable Y_i take $K_i + 1$ possible values; let X_{ik} for $k = 1, \dots, K_i$ be a vector of characteristics/features and $\theta = \{\theta_0, \dots, \theta_{K_i}\}$ a conformable vector of parameters. Then, the multinomial logistic model represents the following probabilities:

$$P(y_i = j|X) = P_{ij} := \frac{\exp(x'_{ij}\theta_j)}{\sum_{k=0}^{K_i} \exp(x'_{ik}\theta_k)}, \tag{3.15}$$

where $i = 1, \dots, n$, $j = 0, \dots, K_i$ and it is assumed without loss of generality that $x_{i0} = 0$. This formulation is quite general, and can represent both conditional logit models (with class specific features; McFadden, 1973) and multinomial logit. Indeed, for the i-th observation and the j-th class we can have:

$$x'_{ij}\theta_j = a + c'_i\beta_j + z'_{ij}\gamma,$$

where c is a vector of characteristics independent of the class (e.g. individual characteristics) and we allow the parameters β_j to depend on the class; z_{ij} is a vector of characteristics that can be class specific and γ is a common parameter vector.

In what follows, we will use a multinomial logit (see Greene, 2003) where we assume $K_i = K$ for every i and some fixed K, and $x_{ij} = x_i = c_i$:

$$x'_{ij}\theta_j = a + c'_i\beta_j.$$

For this model the negative of the log-likelihood is defined as:

$$-\log L = -\sum_{i=1}^{n}\sum_{k=0}^{K} y_{ij}\log(P_{ij}),$$

where $y_{ij} = \mathbb{1}\{y_i = j\}$. We seek parameters θ in order to minimize the quantity above (this may be seen as the loss function that the algorithm tries to minimize). This can be done via gradient based methods (e.g. conjugate gradient methods; see Quarteroni, Sacco, and Saleri, 2010). With

the estimated parameters $\hat{\theta}$, we can then compute estimated probabilities:

$$\hat{P}_{ij} = \frac{\exp(x_i'\hat{\theta}_j)}{\sum_{k=0}^{K}\exp(x_i'\hat{\theta}_k)}, \tag{3.16}$$

and we can classify point x_i to class j if $j = \arg\max_{j\in\{0,\dots,K\}} \hat{P}_{ij}$. Note that in our case a single observation x_i in (3.16) is the pair (x_i, z_k) of firm and individual features.

3.4.2 Classification Trees, Random Forest and Bagged Trees

As pointed out in Hastie, Tibshirani, and Friedman, 2009, the classification trees date back at least to Morgan and Sonquist, 1963. An extensive and dedicated treatment is given in the well known Breiman's book Breiman, 2017. An advanced survey introducing trees and growing procedures at a general level is Safavian and Landgrebe, 1991. A detailed review about the usage of classification trees in different fields and related methodology is Murthy, 1998. In addition, Rokach and Maimon, 2005 is a relatively recent survey of Top-Down induced trees, which also reviews several measures to evaluate goodness of splits. We follow these references to introduce the basics of these models.

For visualizing a tree algorithm refer to Figure 3.2 (at the end of this section). A tree recursively partitions the feature space defining cells (or hypercube in multidimensional spaces). This splitting is done in order to optimize some criterion. In general, this is a difficult task.[12] For this reason, we need to use alternative strategies. Here we illustrate a Top-Down greedy approach to "grow" the tree. In Figure 3.2 a two dimensional space is presented. A tree is composed of node (circles) and edges (lines); the terminal nodes are called *leaf*. At each node, we ask along which variable and where we should split the sample in order to obtain "purer" subsets (where purity is intended as homogeneity of the classes represented in the subsets). Thus, by splitting along X_1 at value a, we are able to define a subset (at the left of the vertical line in a) containing only samples from one class. Continuing with this reasoning, we find the successive best split for X_2 at b (in this particular example an equal good split could be found at X_1) and subsequently a split for X_1 at c. At the third split, we are able to perfectly identify the two classes and there is no reason to continue the splitting procedure. Notice that only one feature at each recursion is eventually used to split the sample and that at each recursion we search the best splitting point across all features. Also, the resulting classifier is highly non-linear, and such a dataset could have not been perfectly classified by any linear model.

Formalizing the discussion above, we need to define a splitting criterion. Let us denote with h a node in the tree. Note that each tree of the node is associated with a region of the space, i.e. the region of the space collecting all the points satisfying the conditions leading to that node. Thus, the node "$X_1 > a$?", the very first node in Figure 3.2, is associated with the entire data since no condition led to this node. Its children nodes are associated to the region at the left of the vertical line at a (left node) and at the right of the line (right node). We call these regions A_h and their number of points is indicated with $|A_h|$. Now, assuming that the outcome variable

[12] For example, Laurent and Rivest, 1976 show that finding a tree minimizing the expected values of splits required to classify an unseen sample is NP-complete; Hancock et al., 1996 showed that it is NP-hard to find the minimum tree consistent with the training set (unless P=NP).

Y assumes at most K different values, we define the estimated proportion of class k in node h as:

$$\hat{p}_{hk} = \frac{1}{|A_h|} \sum_{i \in A_h} \mathbb{1}\{y_i = k\}. \tag{3.17}$$

With this notation, we can now define the splitting criterion. A very popular one is the *Gini* index, and it is given by (evaluated at node h):

$$L_h := \sum_{k \neq k'} \hat{p}_{hk}\hat{p}_{hk'} = \sum_{k=1}^{K} \hat{p}_{hk}(1 - \hat{p}_{hk}).$$

It is easy to see that if h is a pure node, i.e. containing observations from a unique class, then $L_h = 0$. On the other hand, if all the classes are equally present in node h, then L_h is maximal at the value $\frac{K-1}{K}$. Suppose now that we are at node h and we need to decide for the next best split. Say we can split along p variables and that by splitting on variable j, at cutoff s produces the nodes h_1 and h_2 associated with regions:

$$A_1(j,s) := \{x \in \mathcal{X} : x_j \leqslant s\}; \qquad A_2(j,s) := \{x \in \mathcal{X} : x_j > s\}.$$

Then we select j and s so as to minimize the following:

$$\min_{j,s} \{L_{h_1} + L_{h_2}\},$$

where j ranges from 1 to p and s ranges in the domain of x_j.

The full tree is grown by iterating this procedure at each new node. In principle, we could grow the tree as long as splits can be made, or in other words, until we reach the point where every leaf (terminal node) is associated with a single observation in the data. It should be apparent that such a complex tree would be overfitting the data and would not be desirable. We face the dilemma of the bias-variance trade-off. Tree complexity is generally measured by one of the following (Rokach and Maimon, 2005): total number of nodes, total number of leaves, tree depth, or number of features used. It is possible to limit the tree complexity while growing it (e.g. specifying a maximum number of nodes or a minimum number of points in each leaf) or to reduce its complexity ex-post (pruning). The first approach basically defines stopping criteria, leading to an underfitted tree, so that the pruning approach is usually preferred. While there are several ways to prune a tree, the basic idea is to aggregate somehow nodes from a fully grown tree to achieve a better generalization error.[13]

Building on the classification trees, we now briefly review *bagging* and *random forests*.

Bagging

Bagging stands for "**B**ootstrap **Agg**regat**ing**" and was introduced in Breiman, 1996. The basic idea is that of averaging models fitted on independent training sets. Since we usually observe

[13] In Cost-Complexity pruning, for example, by iteratively collapsing nodes from bottom to top, we form several trees from the initial fully grown tree. Loosely speaking, the nodes that are pruned at each iteration are those which carried the least improvement in the growing criterion. All the trees obtained this way are then evaluated on the validation set (or with k-fold cross validation) and the best tree is selected. For this and other pruning methods refer to Rokach and Maimon, 2005.

only one sample, the independent training sets are formed via bootstrap (introduced by Efron, 1979; an accessible detailed reference is Efron and Tibshirani, 1994). Observing a sample of size n, $\mathbb{X}_n$, sample n observations, with replacement, from it; call this sample $\mathbb{X}_n^{*(b)}$. On this sample fit a classification tree to obtain the estimated mapping $\hat{f}_n^{*(b)}$. Repeat for $b = 1, \dots, B$. Upon seeing a new sample, classify it according to:

$$\hat{f}_n^{\text{bag}}(x_{\text{new}}) = \underset{k \in \{1,\dots,K\}}{\arg\max} \frac{1}{B} \sum_{b=1}^{B} \mathbb{1}\left\{k = \hat{f}_n^{*(b)}(x_{\text{new}})\right\},$$

that is, we assign the class assigned by the majority of the bootstrap trees (majority vote). Bagging should improve the overall performance of the algorithm by reducing its variability. To see this, consider the square loss function (Subsection 1.3.2), and consider averaging estimated models $\hat{f}_b$;[14] the averaged estimator is given by $\hat{f} := \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(x)$, where its bias and variance are:

$$\text{Bias}\left[\hat{f}\right] = \mathbb{E}_{Tr}(\hat{f}(x)) - \mathbb{E}(Y|x) = \frac{1}{B}\sum_{b=1}^{B} \mathbb{E}_{Tr}(\hat{f}_b(x)) - \mathbb{E}(Y|x) = \frac{1}{B}\sum_{b=1}^{B} \text{Bias}\left[\hat{f}_b\right];$$

$$\text{Var}\left[\hat{f}\right] = \text{Var}\left[\frac{1}{B}\sum_{b=1}^{B}\hat{f}_b\right] = \frac{1}{B^2}\sum_{b=1}^{B}\text{Var}\left[\hat{f}_b\right] + \frac{2}{B^2}\sum_{i\neq j}\text{Cov}\left[\hat{f}_j, \hat{f}_i\right] \approx \frac{\sigma^2}{B}.$$

Note that averaging unbiased estimators returns an unbiased estimator and the approximation of the variance holds if the averaged estimators have a low covariance (e.g. for independent estimators) and their variance is roughly the same.

This message is general in bootstrap aggregation. It also give an intuition of why bagging works for unstable classifiers rather than stable ones, and why averaging stable poor classifiers is a bad idea and could worsen the overall performances (Breiman, 1996).[15]

It is possible to apply bagging to a variety of estimators. In what follows, we use it with classification trees. One question when using bagging with these estimators is whether to prune or not the trees. In the original paper, Breiman, 1996, pruning is performed using the original sample as a test sample. On on hand, pruning avoids averaging classifiers that overfit the data; on the other hand, too much pruning could lead to averaging classifiers that are too stable, worsening the gains from bagging. An extensive empirical study, Dietterich, 2000, showed that there is no clear pattern to whether pruning makes a substantial difference for the final performances, even though it seems that pruning reduces the number of bootstrap repetitions required.

Random Forests

Random Forests were introduced in Breiman, 2001 and build on the same concept of bagging. Actually, in the paper, the author uses the term Random Forest to refer to a collection of classification procedures based on trees, where each tree uses some independent (identically distributed) "information" with respect to the other trees. However, what is commonly known with the name of Random Forests is a particular example from this class: it is the same as

[14] The following representation is taken from lecture notes of prof. Joachim M. Buhmann's Advanced Machine Learning course, given at ETH Zurich in Autumn 2018 (unpublished material).

[15] A formal treatment of the properties of bagged estimators are given in Bühlmann and Yu, 2002.

Algorithm 4: Random Forests

Input: $\mathbb{X}_n$, training set; B, integer; L, integer;
Output: $\hat{f}$, classifier;

for $b \in 1, \ldots, B$ **do**
  $\mathbb{X}_n^{*(b)} \leftarrow$ random sample with replacement of size n from $\mathbb{X}_n$
  $\hat{f}_n^{*(b)} \leftarrow$ a tree grown on $\mathbb{X}_n^{*(b)}$ where: **while** *not fully grown, for each node* h, **do**
    $\mathbf{l} \leftarrow$ sample without replacement L features among the features set of features of $\mathbb{X}_n$
    $h_1\ h_2 \leftarrow$ children nodes obtained splitting h, considering only the best split among the l features

$\hat{f}(x) \leftarrow \arg\max_{k \in \{1,\ldots,K\}} \frac{1}{B} \sum_{b=1}^{B} \mathbb{1}\left\{\hat{f}_n^{*(b)}(x) = k\right\}$ (majority vote).

bagging, except that each of the tree is grown only on a randomly selected sample of the features. The procedure is illustrated in algorithm 4.

In random forests, the trees are neither pruned nor limited in their growth. It is possible to show that this reduces the correlation of the trees in the forest and that, similarly to bagging, this is desirable when averaging across trees. Also, the overfitting problem arising from fully grown trees has a negligible impact with random forests (Hastie, Tibshirani, and Friedman, 2009).

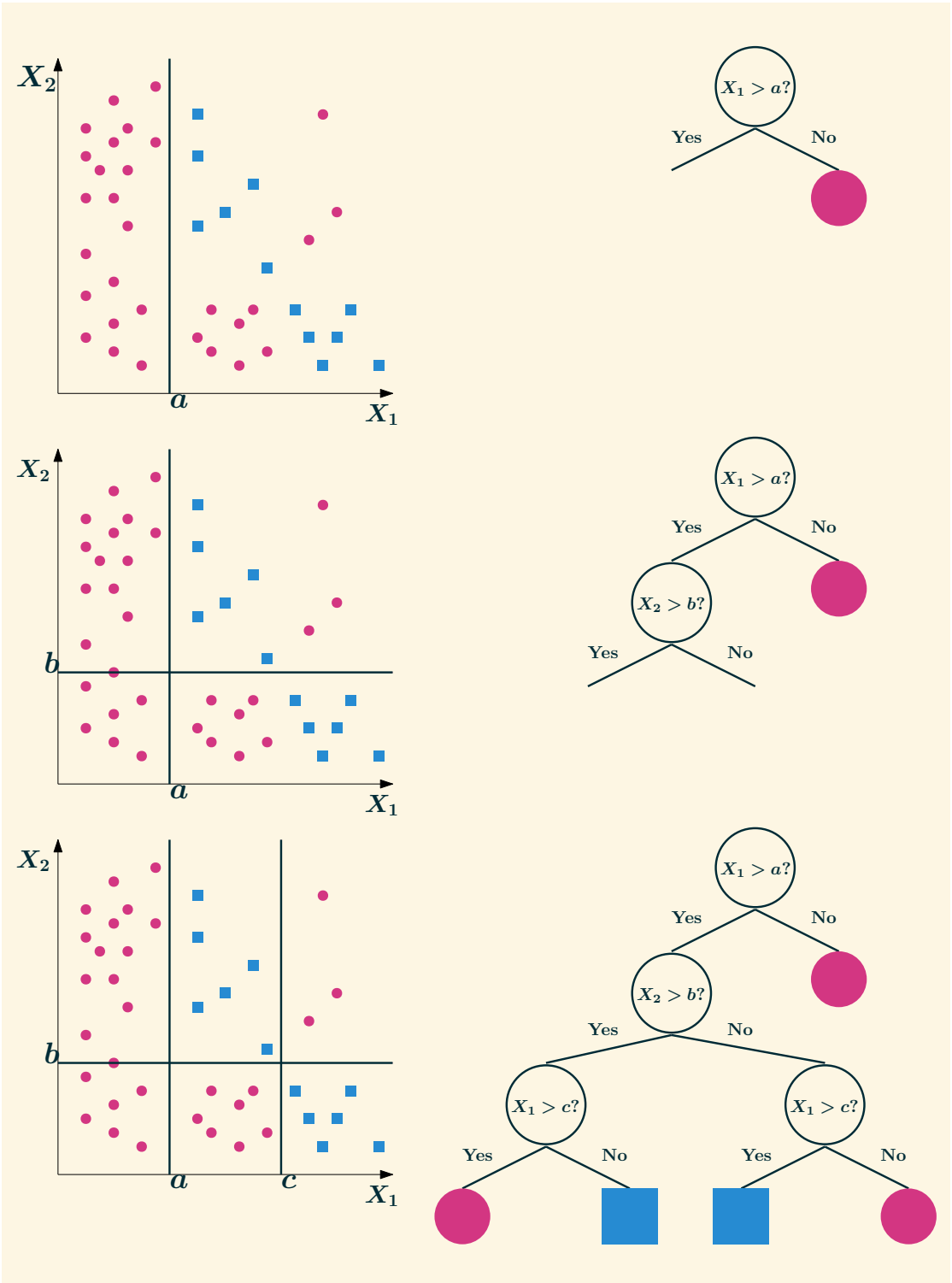


Figure 3.2: Classification Tree. At each node a binary split of the feature is selected in order to induce higher class homogeneity in the created cells.

3.5 Data and Top-firms set construction

In this section we provide data description and some technical notes on the methods we used in the estimation process.

Remark 3.5.1. *The data used contains sensitive information and it was never possible to access it directly. The data contains private information on Swedish citizens a industries. The analysis need to be performed via Statistics Sweden proprietary servers. This constitutes a strong limitation to the methodologies that can be adopted in the analysis, since it is not possible to have direct control on the software available from the servers. This motivated some choices that had to be made to simplify the analysis.*

In addition, the access to the server is granted upon approval from Statistic Sweden. This procedure is very long and can take several months to complete. For this reason, the analysis was performed indirectly, through the aid of a third person, Dr. Joacim Tåg, Program Director at the Research Institute of Industrial Economics (IFN), who mediated the communications with the servers and to whom goes the author's gratitude.

The data used is a database from Statistics Sweden: LISA database (seeTåg, Åstebro, and Thompson, 2016 and Olsson and Tåg, 2017). The dataset is a matched employer-employee panel, collecting individual-level information for all people, older than 15, registered as living in Sweden. Individuals' information is matched with employers' information. In addition, this is also matched with information of occupations, describing for some of the individuals the type of task in which they are employed.

An individual ceases to exist in the data either by dying or by moving to another country. Individual information is available from year 1990 up to 2011. Individual information on occupation is available since year 2001. For this reason, we limit ourselves to the time window 2001-2011.

Individual level characteristics include: personal and social characteristics (e.g. age, immigrant status, sex ...), working history (e.g. days of unemployment, number of firms worked for, ...), education history (e.g. detailed information on type of education, ...). About 70% of individuals do have firm-worker links and information on occupation. For firms there are several characteristics: operating industry, structural components (e.g. family owned, size, ...), productivity measures (e.g. value added per employee ...).

The information on occupation is very detailed and is given by the Swedish Standard Classification of Occupations (SSYK), which is very similar to the International Standard Classification of Occupations (ISCO-88). This data is gathered via two surveys (one is the official wage statistics survey and the other one is a supplementary survey). These surveys are organized as rolling panel and every eligible firm is surveyed at least once every 5 years, but for firms with more than 500 employees, which are always included. The surveys exclude self-employed workers and owners who receives solely dividend as payment from their companies.

Overall, the data comprises roughly 70000 observations on firms and 30 million observations on employees. Tables 3.5 and 3.6 summarize available information at employees and firms levels (these are the features referred as x and z in Section 3.3). Table 3.7 summarizes information on the occupation variable (SSYK; this is the variable we use as the outcome variable, y, in

Subsection 3.3.2).

3.5.1 Finding top-firms in the data

As discussed in Section 3.1, being able to filter out firms using the optimal allocation rule f is essential to estimate it. Were this not possible our estimates would not capture a unique underlying allocation rule used to assign workforce. This advocates for a careful selection of top-firms.

Based on the relationships in (3.4) and (3.5), a possible approach is to select firms with higher level of productivity. In fact, we know that using the allocation f maximizes it. However, this is a naive and likely poor solution as it is. It is well known that (e.g. Syverson, 2011) productivity levels varies strongly across non homogeneous firms (e.g. different industries, different size etc.). Moreover, there are different drivers for this productivity dispersion. If these are not taken into account and if they have much larger impact on productivity than workforce allocation, we are going to to capture these in our split and not the allocation rule f. Thus, a better approach is to first isolate the residual productivity that can be reasonably attributed to workforce allocation and only then select firms showing higher levels of productivity.

Controlling for other factors (the O variables in (3.4)) allows to split firms in (more or less) homogeneous bins and makes the selection of top-firms within bins more reliable. Nonetheless, there is a trade-off. On the one hand, adding more controls allow to better filter out residual productivity depending on allocation rule. On the other hand, adding too much of them will kill all the residual productivity dispersion, also that arising from the allocation rule. For example, in the extreme case of one firm-one bin, all the firms would be top-firms, and all the allocation rules would be optimal allocation rules.

For this reason, controlling for observables should be supported by economic theory. For example, it known that different industries may behave very differently in term of productivity one from another (Bernard and Jones, 1995 or Syverson, 2011), so it may reasonable to consider industries when splitting firms. Controlling for firm's age, on the contrary, might be a more arguable choice: there seems to be contrasting evidence on the relationship between the age and productivity, at least for firms older than 10 years (see Kok, Brouwer, and Fris, 2005). Adding this control could result in adding just more noise to the selection of top-firms.

We now describe the procedure we use to split the data in *top-firms* and *non top-firms*. We also refer to the first set the *learning set*, because on this set we are going to estimate $\hat{f}$. The second set is also referred to as *control set*, and is the set of firms where we are going to evaluate $\hat{f}$ to compute the JAQ measures (see Remark 3.2.3). These will be used to assess the extent by which these firms deviate from the optimal rule f.

The learning set is constructed according to Algorithm 5, which is briefly described as follows. According to some criteria, we split the full data in bins and from these we extract a part of top-firms. Further refinement criteria are applied and the learning and control set are defined. The criteria used to split the data in different bins are defined by control variables.

We now give an example of how the data are split according to Algorithm 5. Consider the

Algorithm 5: Create top-firms set

Input: $\mathbb{X}$ full firms data; $B_1, \ldots, B_M$ binning criteria; t thresholding criterion; s splitting variable; R further refinement criterion.

Output: *learning set*, *control set*.

for $m \in 1, \ldots, M$ **do**

$\mathbb{X}_m \leftarrow \{x \in \mathbb{X} : x \text{ satisfies } B_m\}$

$\text{top}_m \leftarrow \{x \in \mathbb{X}_m : s(x) \geqslant t(\mathbb{X}_m)\}$. I.e. define top the elements of the subset for which the variable s is above a threshold, computed on the subset itself.

$\text{top} \leftarrow \{(top_m)_m\}$. I.e. join the top elements.

learning set $\leftarrow \{x \in \text{top} : x \text{ satisfies } R\}$

control set $\leftarrow \mathbb{X} \backslash$ *learning set*

following splitting criteria:

1. Size intervals: $[50, 100]$; $[101, 250]$; $[251, 1000]$; $[1001, 6000]$.

2. Year: 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010.

3. Industry: 58 categories of industries (see Table 3.6).

A bin B_m will be defined as one of the possible combinations of the above criteria (e.g. observations in year 2005, firms with size in $[50, 100]$ and within industry 6). We split on value added per employee, and this defines variable s. For each bin, we compute the top productivity decile; this defines the thresholding criterion t. Then, we collect for each bin, firms with s greater or equal than the (within bin) top productivity decile. Finally, we refine the set by keeping as top-firms only those that for at least 6 years were classified as top-firms in this way. This defines the refinement criterion R. Note that top-firms set is made by firm, and not by firm in particular years. Thus, if a firm is in this set, it is in this set for the whole data time window, even if for some of the year considered its productivity level was not above the threshold. The idea is that allocation rule is not changing with the years, while productivity may undergo some shocks.

Table 3.5: Summaries on Employees Characteristics. Total Observations: 28710464

(a) Observations on Employees per Year

year	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010
obs.	2816199	2857449	2818214	2834160	2828429	2900556	2964534	2953657	2836605	2900661

(b) Age

min.	16
25%	33
50%	43
75%	53
max.	99
mean.	42.6
std. dev.	12.47

(c) Labor Market Experience

min.	0
25%	8
50%	19
75%	29
max.	83
mean.	19.42
std. dev.	13.07

(d) Tenure

min.	0
25%	1
50%	4
75%	10
max.	20
mean.	5.80
std. dev.	5.55

(e) Days in unemployment

min.	0
25%	0
50%	0
75%	228
max.	5181
mean.	192.07
std. dev.	365.28

(f) Education level (Obs.)

Basic	3466087
High school	13679968
Vocational	3811652
University	7668471
N/A	84286

(g) Female/Male composition

Female	15592450
Male	13118014

(h) Immigrant composition

Immigrant	4054687
Not immigrant	24655777

(i) Lives in birth country

Yes	17321074
No	11389390

(j) Graduated during recession

Yes	5311539
No	23398925

(k) Narrow Education level

N. different Categories	347
Obs. Smallest Category	4
Median Obs. in Categories	22670
Obs. Biggest Category	3663270
Mean. Obs. in Categories	82739.09
Std. dev. Obs. in Categories	245780.28

(l) Municipality (Obs.)

N. different Categories	291
Obs. Smallest Category	5868
Median Obs. in Categories	43725
Obs. Biggest Category	2569869
Mean. Obs. in Categories	98661.39
Std. dev. Obs. in Categories	198399.91

(m) N. industries worked in (since 1990)

0	29312
1	20627635
2	6807573
3	1086231
4	142798
5	15612
6	1228
7	69
8	6

(n) N. firms worked at (since 1990)

1	7938172	11	13133
2	7507355	12	4735
3	5655882	13	1703
4	3648224	14	584
5	2056739	15	178
6	1048291	16	63
7	495679	17	22
8	215264	18	5
9	89409	19	1
10	35025		

Table 3.6: Summaries on Firms Characteristics. Total Observations: 71432

(a) Observations on Firms per Year

year	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010
obs.	6884	6936	6788	6891	6928	7152	7476	7600	7249	7528

(b) Productivity: value added per employee (MSEK 2017)

min.	−17.28
25%	0.46
50%	0.60
75%	0.81
max.	245.10
mean.	0.76
std. dev.	1.70

(c) Firm Size (in Employees)

min.	50
25%	65
50%	99
75%	210
max.	52151
mean.	401.93
std. dev.	1592.15

(d) Sales (MSEK 2017)

min.	0
25%	77.84
50%	166.42
75%	397.12
max.	120641.4
mean.	664.92
std. dev.	2954.57

(e) Total assets (MSEK 2017)

min.	0
25%	12.19
50%	61.74
75%	203.03
max.	362723.5
mean.	789.71
std. dev.	6959.61

(f) Firm age from 1990

min.	0
25%	9
50%	13
75%	16
max.	20
mean.	12.01
std. dev.	5.29

(g) Firm industry

N. different Categories	58
Obs. Smallest Category	7
Median Obs. in Categories	688
Obs. Biggest Category	7322
Mean. Obs. in Categories	1231.59
Std. dev. Obs. in Categories	1511.39

(h) Family Firm

Yes	10870
No	60562

(i) State or municipality owned

Yes	9866
No	61566

(j) Listed Firm

Yes	870
No	70562

(k) Broad industry (Obs.)

Not harmonized or other	466	Agriculture, hunting, forestry, and fishing	429
Mining	141	Manufacturing	18846
Utilities	889	Construction	3753
Wholesale and retail	11109	Hotels and restaurants	2070
Transport, storage, and communications	4761	Financial intermediation	1886
Real estate, renting, and business	12486	Public administration and defence	1496
Education	3285	Health and social work	5577
Other service activities	4238		

Table 3.7: Occupation Variable (SSYK). Right columns of panels (b), (c), (d) report values in number of employees observed in the data.

(a) Occupation Categories. Major groups, number of subdivisions and skill level.

Code	Major groups	N. Sub-major groups	N. minor groups	N. Unit groups	Skill level (ISCO)
1	Legislators, senior officials and managers	3	6	29	N/A
2	Professionals	4	21	67	4th
3	Technicians and associate professionals	4	19	72	3rd
4	Clerks	2	8	17	2nd
5	Service workers and shop sales workers	2	7	27	2nd
6	Skilled agricultural and fishery workers	1	5	11	2nd
7	Craft and related trades workers	4	16	58	2nd
8	Plant and machine operators and assemblers	3	20	59	2nd
9	Elementary occupations	3	10	14	1st
0	Armed forces	1	1	1	N/A
Totals	9	27	113	355	

(b) Composition Major groups (Obs.).

Not available	1317155
Managers	1279702
Professionals	5760078
Technicians and associate professionals	5580636
Clerks	2474938
Service, shop, and market sales workers	6105558
Skilled agricultural and fishery worker	110989
Craft and trades related workers	1787794
Plant and machine operators and assemblers	2597792
Elementary occupations	1695822
Total	28710464

(c) Minor groups composition

N. different Categories	113
Obs. Smallest Category	114
Median Obs. in Categories	115047
Obs. Biggest Category	4549131
Mean. Obs. in Categories	243451.19
Std. dev. Obs. in Categories	469362.06
N/A	1200480
Total	28710464

(d) Unit groups composition.

N. different Categories	355
Obs. Smallest Category	7
Median Obs. in Categories	19863
Obs. Biggest Category	1526392
Mean. Obs. in Categories	67435.17
Std. dev. Obs. in Categories	152118.08
N/A	4770978
Total	28710464

3.6 Empirical Analysis

As seen in Section 3.4 and Section 3.5, our analysis consists of two main tasks. The first is the creation of the learning set and the second in the estimation of an allocation rule $\hat{f}$. In this section, we present the empirical results for both of them.

3.6.1 Top-firms and non top-firms sets

Motivated by the discussion in Subsection 3.2.1, the degree by which we are able to estimate the optimal allocation rule f depends on the ability to isolate the top-firms from the non top-firms.

Data presented in Section 3.5 is split in bins according to the following criteria in input to Algorithm 5.

1. Size quantiles:$(-\infty, Q_{25\%})$; $[Q_{25\%}, Q_{50\%})$; $[Q_{50\%}, Q_{75\%})$; $[Q_{75\%}, \infty)$.
2. Total Assets quantiles:$(-\infty, Q_{25\%})$; $[Q_{25\%}, Q_{50\%})$; $[Q_{50\%}, Q_{75\%})$; $[Q_{75\%}, \infty)$.
3. Year: 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010.
4. Industry: 58 categories of industries (see 3.6).
5. s is taken to be the value added per employee.
6. t is taken to be the top decile, $Q_{90\%}$ of s in the bin, B_m.
7. R is taken to be: "Satisfy t for at least 6 different years".

The quantiles above are computed on the average firms' size and total assets over the years. Results of this split are visualized in Figures 3.3 and 3.4.

Figure 3.3 shows the average counts of firms in each bin, across years and industries. These bins are created splitting firms by number of employees (size) and total assets, within industries and years. We used these quantities to control for labour and capital input, known to determine productivity (Syverson, 2011). Note that the bin identified by $[Q_{75\%}, \infty)$ for both size and asset is inflated with respect to the others: this is because of the upper limit ∞, used to take into account everything above the 75% quantile. Even if none of the bin is empty, the low average counts suggests that adding further controls would be detrimental to the analysis, to the trade-off exposed in Subsection 3.5.1. For each bin, the top 10% firms for value added per employee will be assigned a "top" tag; firms exhibiting this tag for at least 6 years will be considered top-firms (for all the years).

Figure 3.4 provides summaries for the two sets constructed (for variables, refer to Table 3.5 and Table 3.5). This is also a sanity check in the sense that we do not want the two sets to be very different in nature. In fact, if they were, it would not make sense at all to allocate workforce using f in the non top-firms set. After all, for the world of non top-firms resources would be totally different with respect to top-firms. Instead, we want to capture deviations from the optimal allocations where it is reasonable that this is optimal for both worlds. Thus, at least common support for the features involved is a requirement. Nonetheless, differences are still expected by sample selection. We require them not to be too severe. In addition, higher degree

Figure 3.3: Histogram of the average number of firms in each bin across years and industries. Of each bin, the top decile will be candidate to be in the top-firms set (or learning set). Size and Assets quantiles on the x-axis and y-axis; counts on the z-axis; mean of the bin counts is shown in the title. Note that the last bin along the diagonal is inflated with respect to the others; this is due to the ∞ bound, used to capture everything above the 75% quantile.

of similarity implies that the extension of the inferred assignment rule to the non top-firm set is more reliable. Overall, the two sets seem to share a good amount of similarity. The only notable difference is on productivity ("VA/Employee") and is due to the construction of the sets. The resulting sets are composed of a total of ($\approx$ 400) thousands observations in top-firms set against ($\approx$ 10) million in the non top-firm set. This is a ratio of approximately 5% (observations in number of employees). This is due to the stringent criteria for a firm to qualify for the top-firm set. Unless otherwise specified, algorithms shown later ran with this version of the learning set data.

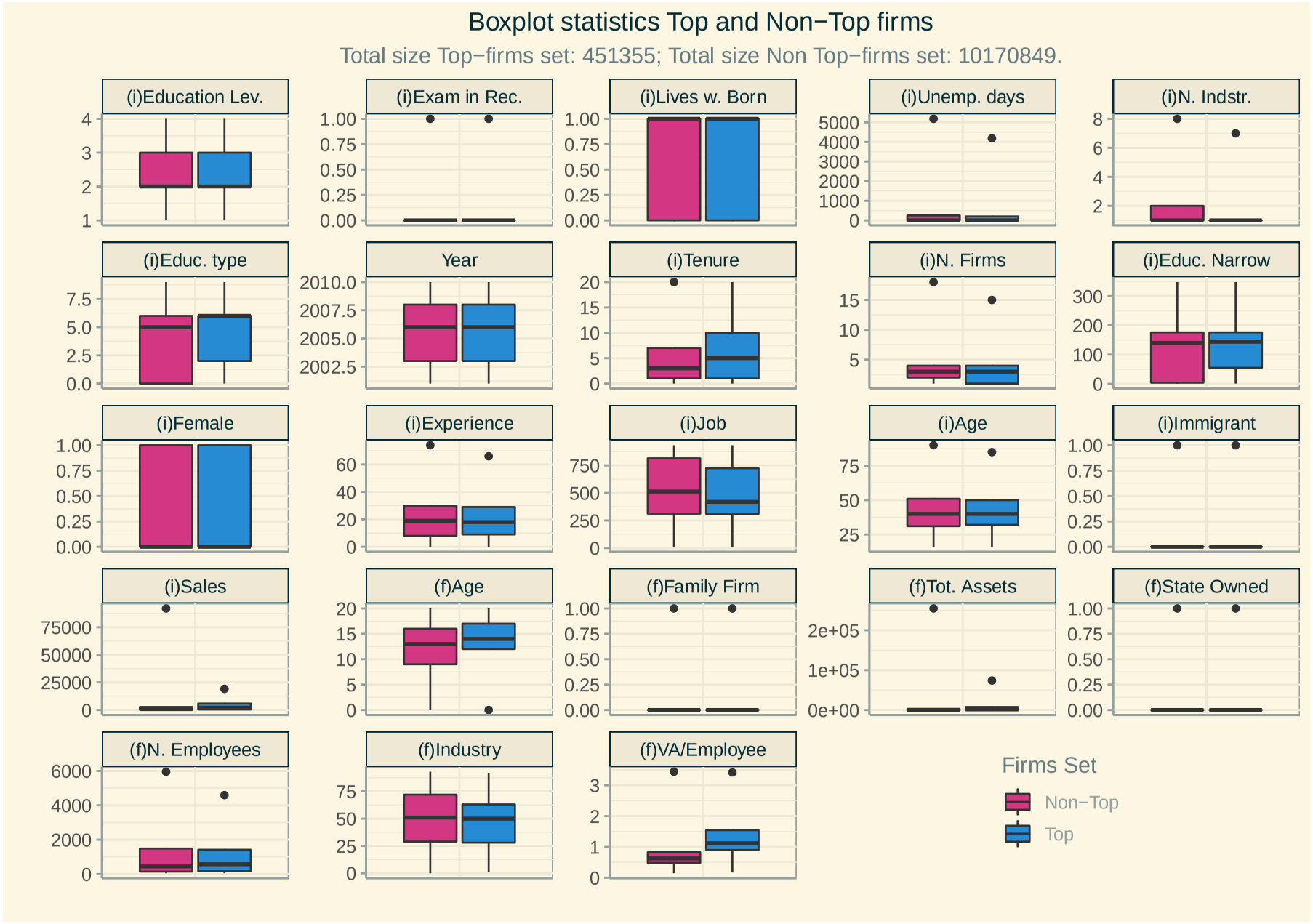


Figure 3.4: Boxplot visualization for employees' (i) and firms' (f) variables (x and z) in top-firms and non top-firms sets. Overall, the firms shows common supports and good amount of similarity. Even if different sets are expected, good similarity helps trusting the extension of the inferred assignment rule to non top-firms. Note: boxplots are shown for categorical variable as well; these provide indirect information only on big different concentration of categories between the two sets.

3.6.2 Notes on algorithms implementation

The algorithms used in the empirical analysis are multinomial logit, random forest and bagging with tree classifiers (see Section 3.4)

Outcome and feature variables used are collected in Table 3.8. Unless otherwise specified, all the algorithms used these as input features. The outcome variable used changes according to the algorithm due to computational issues, and will be indicated when discussing results. Input features are to be intended as the pairs (x, z) and outcome variable (classes) is to be intended as the jobs y (refer to Section 3.2).

We used two programming languages in this analysis: R (R Core Team, 2019) and Python (for a reference, see `www.python.org`). Because we run the algorithms on a third party server, it is not at all possible to modify the libraries providing them. In addition we were forced to drop some methods (e.g. neural network) because of missing libraries. A positive aspect is that we had to use off-the-shelf algorithms implementations for the core estimation and we can report on the performances of ready-to-use packages.

Multinomial Logit

The multinomial logit in Subsection 3.4.1 was implemented via the scikit-learn library in Python (Pedregosa et al., 2011). The algorithm used is the Logistic Regression algorithm with no penalization. For this model, we added as a regressor interacting firms' Industry variable with the individuals' Narrow Education level (see Table 3.8). This algorithm (differently from the others) has as objective the conditional probabilities, so that $\hat{f}_j$ are directly estimated. estimated.

Random Forests

Random forests (see Subsection 3.4.2) are particularly interesting in our case for multiple reasons: they have (relatively) few tuning parameters, and this is a plus in our case since we can not easily iterate on the server; they generally powerful estimators; they are less prone to overfit than other methods. We implemented the random forest estimator via the *randomForest* R package (Liaw and Wiener, 2002), building on the original Breiman's Fortran code.

Because this implementation does not handle categorical variables with more than 56 categories, we used the encoding via the target statistics proposed in (Micci-Barreca, 2001).[16] This was used for the Narrow Education level and also for the interaction variable between Narrow Education level and firms' Industry.[17]

For this algorithm, we use the Gini information criterion and set the number of features sampled at each node at the square root of total features multiplied by 1.5 (roughly 20

[16] This method encodes the categorical variable of interest with as many variables as the levels of the output variable. Say j is a level of the outcome variable, y; x is the categorical variable of interest and $t = 1, \dots, T$ are the levels of x. Then, for each j, create $x^{(j)}$ by replacing level t (for every t) of variable x with: $\sum_{i=1}^{n} \frac{\mathbb{1}\{y_i=j\}\,\mathbb{1}\{x_i=t\}}{\mathbb{1}\{x_i=t\}}$. In the paper, the author proposed a more structured a rigorous version. However, we use this one because of its simplicity and the high number of available observations.

[17] These interaction would not be useful in a tree, since it is automatically captured by the estimator structure. However, in a random forest, due to the random sampling of the feature, this is not always the case. We decided to include these in the estimator because of high conjectured relevance: experimentations confirm the conjecture and the performance of the estimator tend to be higher when these variables are included.

feature per split; this was chosen in a training-validation approach as described in Subsection 1.3.2). The size of terminal nodes was set to a minimum of 3 to 5 observations (going below this number caused memory overflows). We used total number of trees in the forests ranging from 1000 to 5000 trees.

For this implementation, class probabilities $\hat{f}_j$ are obtained from the proportion of trees voting for a class (see Algorithm 4). This is how they are implemented in Liaw and Wiener, 2002, even if they are know not to be good estimators of class probabilities (Hastie, Tibshirani, and Friedman, 2009; this is also the reason why we considered bagging).

Because of the excessive memory footprint of this package, we were also forced to consider random forest implementation from *ranger* (Wright and Ziegler, 2017) and the *CORElearn* (Robnik-Sikonja and Savicky, 2018). The former is efficient on estimation but not on prediction, thus it can be problematic on extremely large prediction tasks. The latter, on the contrary, is efficient in estimations and has a much smaller memory footprint.

Bagged trees

Bagged trees (Subsection 3.4.2) shares with the random forest the advantage of few tunable parameters. The estimator is implemented via the *CORElearn* R package (Robnik-Sikonja and Savicky, 2018).

We use the Gini information criterion to split nodes. The terminal nodes' size is kept at a minimum of 5 (going below this number increases estimation time excessively). We use 300 to 500 trees. The class probabilities are estimated via classes fraction in trees' terminal nodes (exact formulation given in Section 3.8, (3.18)) and averaged across the trees (as advocated in Hastie, Tibshirani, and Friedman, 2009). Moreover, we do not prune trees and use a Laplace smoothing for class probability estimates. More detail on these are to be found in Section 3.8.

Table 3.8: Variables used in the estimation.

(a) Dependent Variable / Outcome Varaible

Variable	Type
Occupation: Major groups	Categorical (9 cat.)
Occupation: Sub-major groups	Categorical (27 cat.)
Occupation: Minor groups	Categorical (100 cat.[18])

(b) Feature variables as input to the estimation task.

individual (i) firm (f)	Variable	Type
(i)	Firms worked at	quantitative
(i)	Tenure	quantitative
(i)	Sex	dummy
(i)	Education Type	categorical (10 cat.[19])
(i)	Graduate during Recession	dummy
(i)	Labour Market Exp.	quantitative
(i)	Age	quantitative
(i)	Industries worked at	quantitative
(i)	Education Level	categorical (4 cat.)
(i)	Lives where born	dummy
(i)	Unemployment days	quantitative
(i)	Immigrant	dummy
(i)	Narrow Education	categorical (347 cat.)
(f)	Family firm	dummy
(f)	Industry	categorical (50 cat.[20])
(f)	State owned	dummy
(f)	Total assets	quantitative
(f)	Sales	quantitative
(f)	Age	quantitative
(f)	Number of employees	quantitative

[18]The following occupations were aggregated due to the few number of observations: $110 = \{111, 112\}$; $520 = \{521, 522\}$; $619 = \{611, 612, 613, 615\}$; $740 = \{741, 742, 743, 744\}$. The following occupations were dropped due to the few number of observations and diversity with respect to other categories: 246; 345; 348; 733; 911.

[19]This variable is not in the summary Table 3.5; it is a reduced version of the Narrow education type variable.

[20]Due to the few observations and to lower the number of categories in order to be handled by standard software, the following were aggregated:$10 = \{10, 13, 14, 16, 17, 18, 19\}$; $93 = \{93, 99\}$.

3.6.3 Results on the estimation of employees-to-tasks allocation

In this section we analyse the results on the employees-to-task classification problem, i.e. estimation of f.

Unless otherwise specified, the categorization of the dependent variable is taken to be the "Minor groups compositions" (see Table 3.7), for a total to 100 jobs (classes). This is the finest level of the occupational information we were able to use. Being able to exploit such a fine level of categorization is a great plus of some of the procedure considered. Analyses were conducted also on the other two categorization levels. Estimates are less noisy in this case, due to the higher balancedness of classes. With the finer categorization, some of the jobs are extremely under-represented in the data (see Table 3.7).

To evaluate performances on the classification task we used the overall accuracy score and the class-wise F1-score. We also aggregate the latter by averaging it across the 100 classes (both simple and weighted averages; weights are given by class proportions). These measure the classification ability of the classifier (i.e. $\hat{f}$). To evaluate the goodness of fit of estimated conditional class probabilities (i.e. $\hat{f}_j$) we use the Brier score. Due to the limitations of the analysis (see Remark 3.5.1), it was not possible to implement more sophisticated measures. Details on the evaluation methods are to be found in Section 3.8.

Results are reported in Figure 3.5, Figure 3.6a, and Figure 3.6b at the end of this section. These report the evaluation metrics just described (when available) at increasing cutoffs for the highest class probability. In fact, to cope with the problem of unclassifiable observations, based on Remark 3.3.3, at each threshold we consider only individuals with at least one class' predicted probability ($\hat{f}_j$) higher than the threshold. Class-wise F1-scores are reported for a selected threshold.

Results are presented for top-firms set split in a training set and a test set in 90%-10%, motivated by the big size of the data (Subsection 1.3.2).

Multinomial Logit

From our experiments it is clear that multinomial logit, at least as implemented in its classical version (see Subsection 3.4.1), can not be used to estimate an optimal allocation rule with the data analysed. The high number of observations and the huge number of categories of the outcome variable caused the algorithm to not converge (in term of likelihood improvements) in feasible amount of time.

We had to downscale the problem considering only for firms with less than 250 employees in the manufacturing industry (data presented in above; see Figure 3.4). This amount to roughly 50 and 600 thousand observations for top-firms and non top-firms respectively. Moreover, we also had to consider the other two coarser categorizations for the outcome variable (see Table 3.7).

Referring to Figure 3.5, we can see that results are not particularly good. The aggregated scores are low unless imposing a high threshold. This however results in a massive drop of data. At the selected threshold of 0.1, where almost 70% of the data are kept, the algorithm is not able to identify class 6 at all. For the 27 class categorization, only aggregated results are shown. These confirm general poor results. However, we note that the algorithm did

not achieve convergence.

Random Forests

Figure 3.6a shows results for an implementation of random forests with the *randomForest* package. The forest has 5000 trees. This algorithm was trained considering firms with less than 250 employees (from data in Figure 3.4) (300 thousands observations in top-firm set; 1.7 million in non top-firms).

The forest was able to successfully handle the 100 classes categorization. Nonetheless, we see that it clearly overfits the data. This can be seen by the accentuated distance of the two line for the aggregated scores and from the very different pattern of F1-score. In this particular implementation it seems that the number of features selected at each split was too high, inducing excessive correlation among trees in the forest.

Even if overfitting, these results are instructive in the sense that they show that it is possible to achieve an high fit of the data. At a cutoff of 0.3, 70% of the observations are kept and we achieve 80% aggregated scores for the test set.

This particular implementation, even with a lower number of trees (500), has a large memory footprint, so that it was not possible to test it again on Statistic Sweden servers on the full dataset. For this reason other implementations are being considered (see random forests in Subsection 3.6.2).

Bagged Trees

Results for this algorithm are shown in Figure 3.6b. This algorithm was trained of full data shown in Figure 3.4. This was the only algorithm able to handle such a dataset in our experimentations.

We can see that the algorithm does not overfit. This is shown by very similar pattern in all the considered metrics for both train and test results.

A 0.1 cutoff keeps roughly 80% of the data (in Figure 3.6b this is represented by shaded areas in the metric score panels) and achieves 70 to 75% accuracy and F1 aggregated scores, and a brier score of ≈ 0.5. As shown by the class-wise F1 score, not all the classes are retrieved by the algorithm. This is expected due to the high unbalance in classes proportions, especially for the 100 classes categorization.

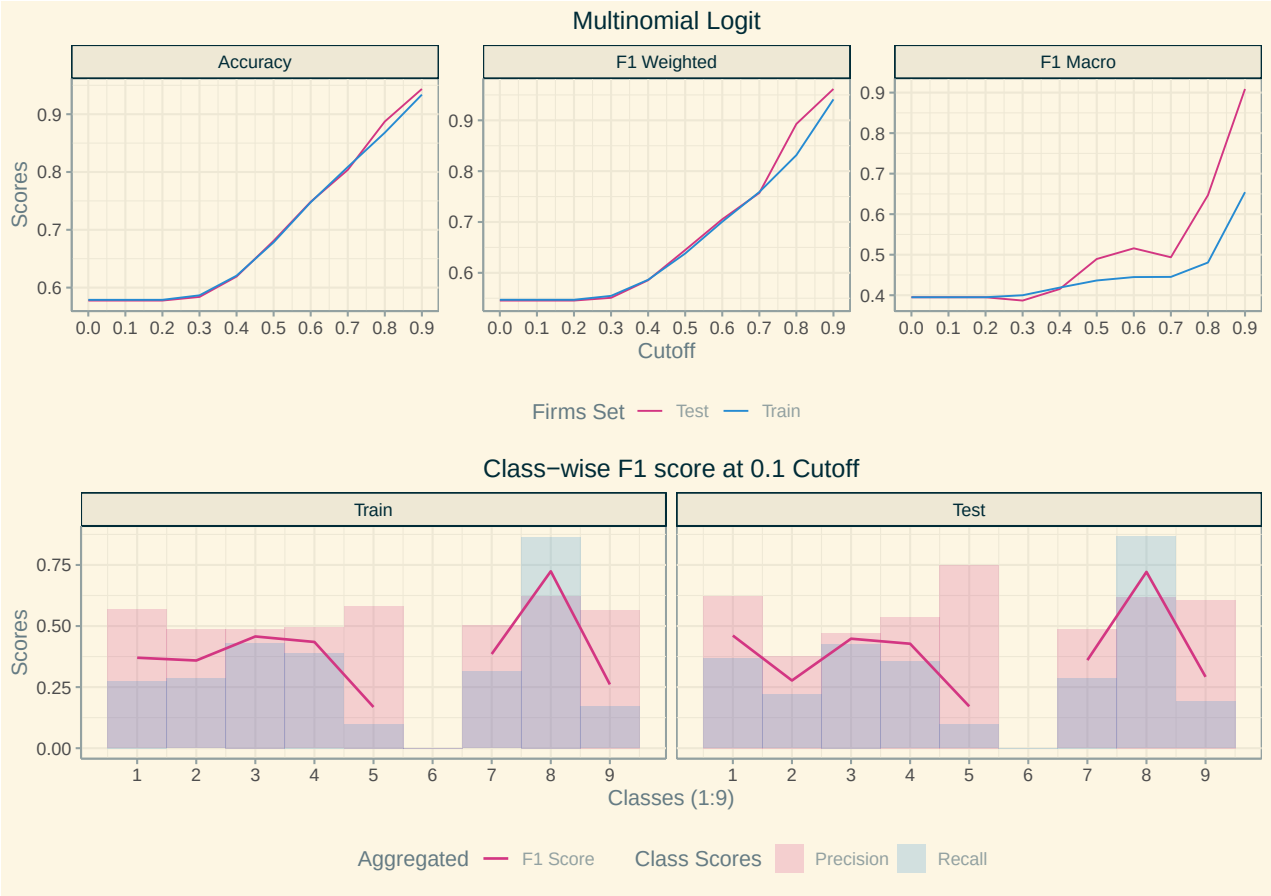
Even if these results are not as good as the those seen for the random forest (Figure 3.6a) they are much more reliable and are also stable across several iterations we ran.

Overall, we find that the most stable and reliable estimate is given by bagged trees. Since we need to use the estimated rule on a different set than the training one (see considerations immediately preceding Remark 3.2.3), we value even more than usual stability of the estimates and lack of overfitting behaviour. For this reason the bagged trees are the advocated solution to estimate the optimal allocation rule f.

In the next section, we show results on the FJAQ4 obtained with this estimator.

Figure 3.5: Multinomial logit results. Total sample size ≈ 300000; ≈ 50000 top-firms size.

(a) Results with Major groups categorization.



(b) Results with Sub-Major groups categorization (27 classes). Class-wise scores not estimated.

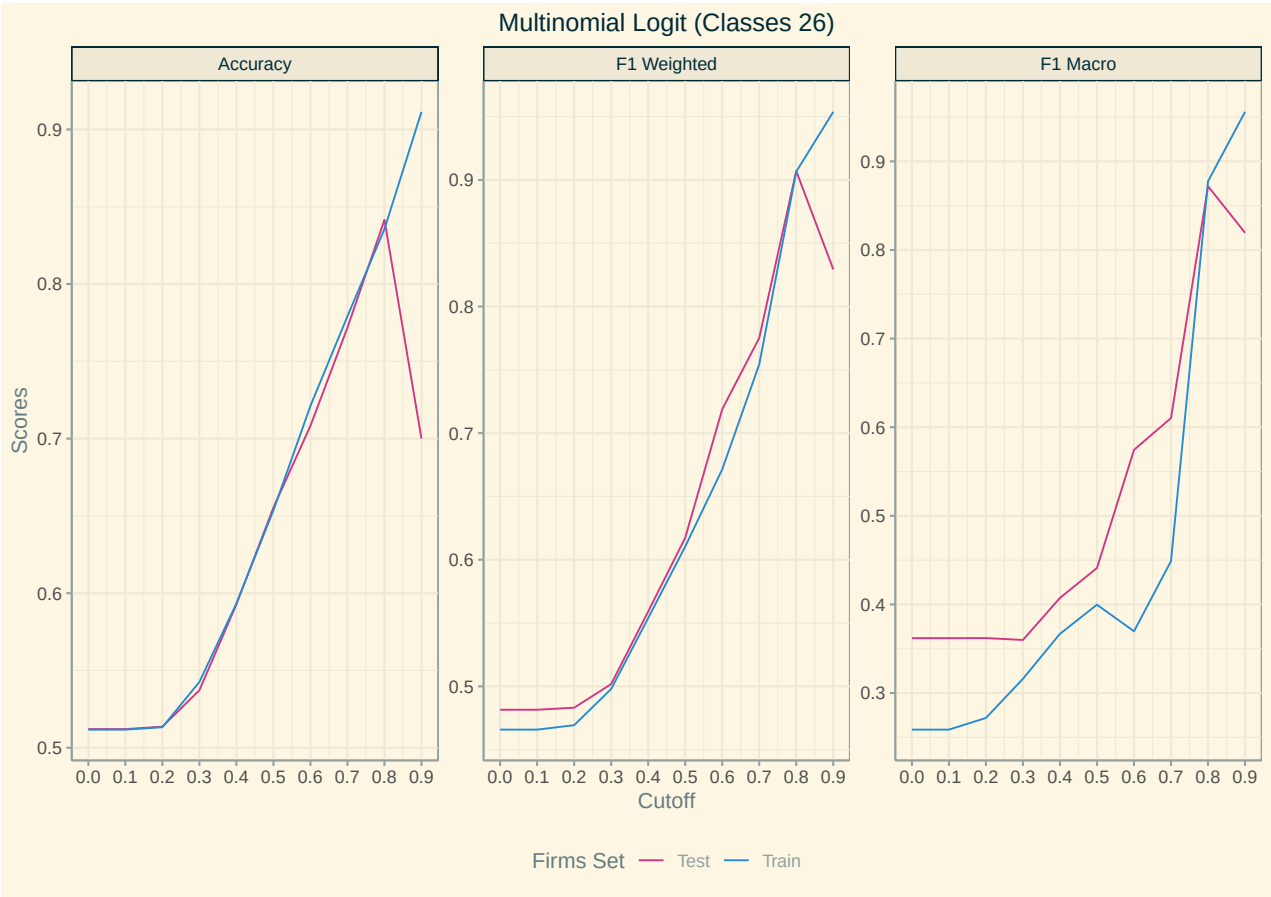
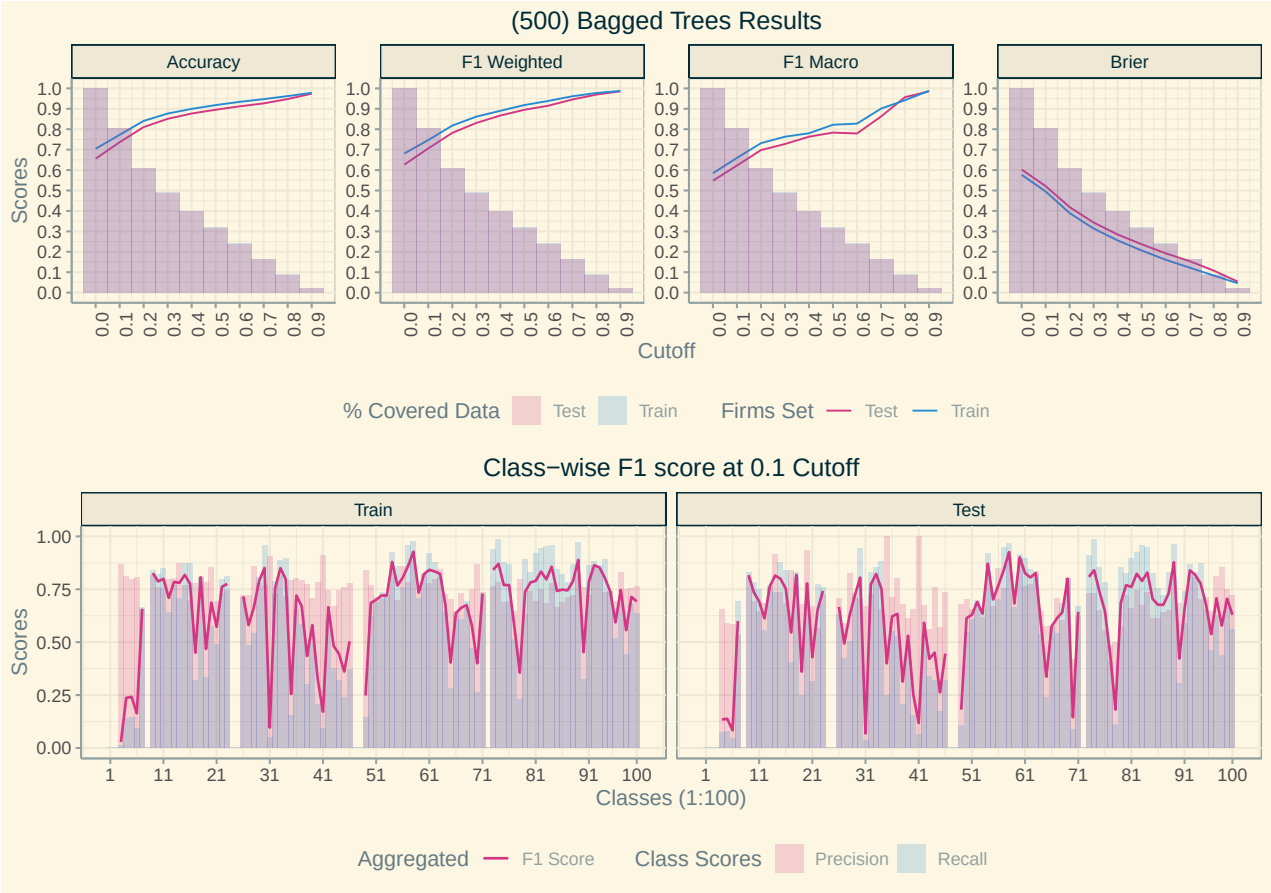


Figure 3.6: Random Forests and Bagged Trees results

(a) Random Forest

(b) Bagged Trees



3.6.4 FJAQ figures and productivity

In this section, we report the analysis on FJAQ4 (3.11) and productivity, where the former was computed with estimates from the bagged trees estimator as shown in Subsection 3.6.3.

As commented in Subsection 3.3.2 FJAQ2 and FJAQ3 measure ((3.13) (3.14)) are computationally intensive. Even if Algorithm 3 is used (this is needed to make the problem feasible), these measures still requires more than one week to compute on a reduced version of the data, as the one used for random forests (see Subsection 3.6.2). However, preliminary results showed that, qualitatively, FJAQ4 is a good proxy of the others. Indeed, it seems that neglecting the assignment problem does not lead to a dramatically different interpretation for deviating behaviours with respect to the optimal (estimated) allocation rule. We attribute this to relative few occurrences of predicted assignments that coincide with non available positions or that are overly-concentrated on the same tasks.

Results are thus shown only for the FJAQ4 measure only.[21]

Figure 3.7, shows a plot of FJAQ4 against productivity. For non top firms there seems to be a strong positive relationship, which deteriorates in the top-firms set. This is as expected from Remark 3.2.3. Firms that deviates less from the optimal (estimated) allocation rule are (higher FJAQ4) achieves higher productivity.

This is also confirmed in a preliminary regression analysis. Here, we regress the FJAQ4 measure against productivity (deciles), controlling for other factors. Even after adding controls, the positive correlation in control set (non top-firms) seems to hold. In the top-firm this is less clear and the estimates, relative to productivity deciles, show more variability. An intuition for this fact is as follows. The split in top-firms and non top firms is made by conditioning on some factors. Then, the estimated function $\hat{f}$ is estimated on the firms that (supposedly) use the optimal allocation rule. For this reason, once we control for the same factors with which we operated the split, there should be no left variability in the top-firm set to be correlated with the FJAQ. On the contrary, in the non top-firm set, the residual productivity can still be affected by the different allocation rules by which firms in this set allocate their workforce.

These results are not meant to be causality statements, but are encouraging in investigating further these measures.

[21] We are not showing FJAQ1, FJAQ2 and FJAQ3 results since the available ones were produced in early stages of the experimental phase (in our implementation FJAQ1 is produced along with FJAQ2 and FJAQ3). At the time, the split in top-firm and non top-firm was implemented considering only industry as control variable for the split. For this reason, estimates are much less reliable. However, the similar pattern between FJAQ4, FJAQ2 and FJAQ3 was already observed. Hence, in subsequent trials, the computations based on the assignment problems were suspended until a stable solution for the estimation of f would have been achieved.

Table 3.9: Regression tables: FJAQ4 against productivity deciles (top decile omitted), years, total assets, firm size and industry. Robust standard errors adjusted by firms' clusters. (∗) means significant at 10% level.

(a) Non Top-Firms set. Bagged Trees.

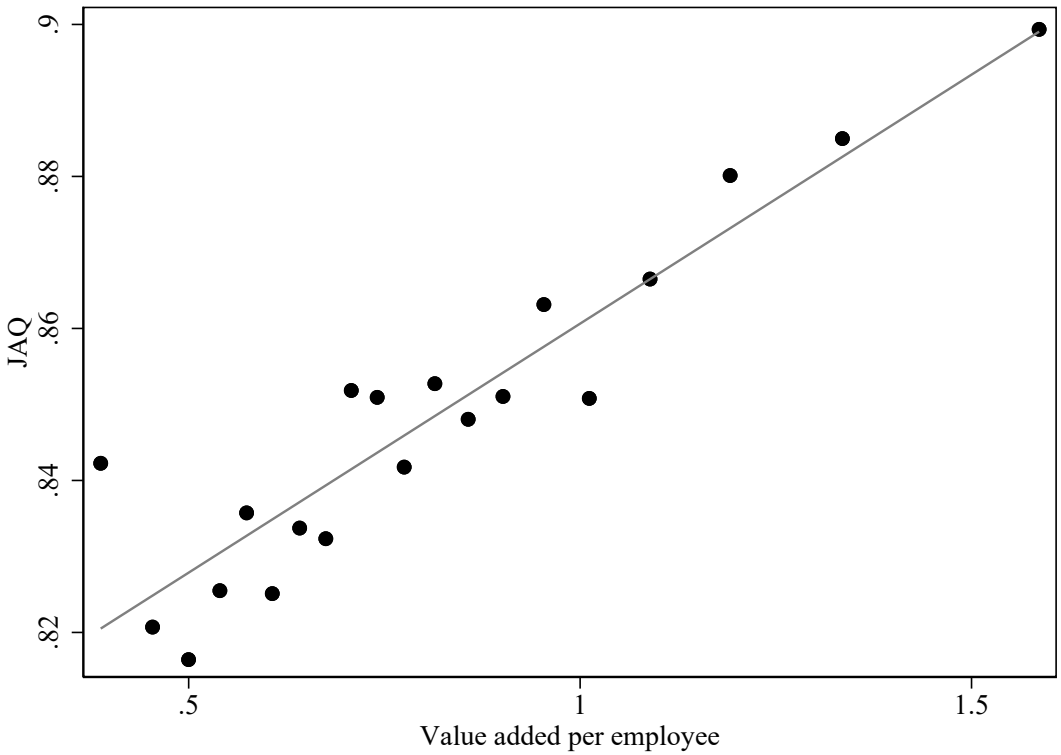
dep.: FJAQ4	Coef.	Std. Error	T-stat	$p-value$	95% conf.	interval
Productivity Decile						
1	−0.021	0.011	−1.940	0.052*	−0.041	0.000
2	−0.023	0.008	−2.790	0.005*	−0.038	−0.007
3	−0.023	0.008	−2.990	0.003*	−0.038	−0.008
4	−0.013	0.007	−1.830	0.068*	−0.028	0.001
5	−0.013	0.007	−1.810	0.070*	−0.026	0.001
6	−0.012	0.007	−1.680	0.092*	−0.026	0.002
7	−0.015	0.007	−2.160	0.031*	−0.029	−0.001
8	−0.004	0.007	−0.650	0.513	−0.017	0.009
9	0.003	0.006	0.390	0.694	−0.010	0.015
(f) Size	−0.000	5.97E-06	−5.31	0.000*	−0.000	−0.000
(f) TA	1.13E-06	7.28E-07	1.56	0.119	−2.93E-07	2.56E-06
year			(omitted)			
(f) Industry			(omitted)			
constant			(omitted)			
N. Obs (firms): 4812			$R^2 = 0.49$			

(b) Top-Firms set. Bagged Trees.

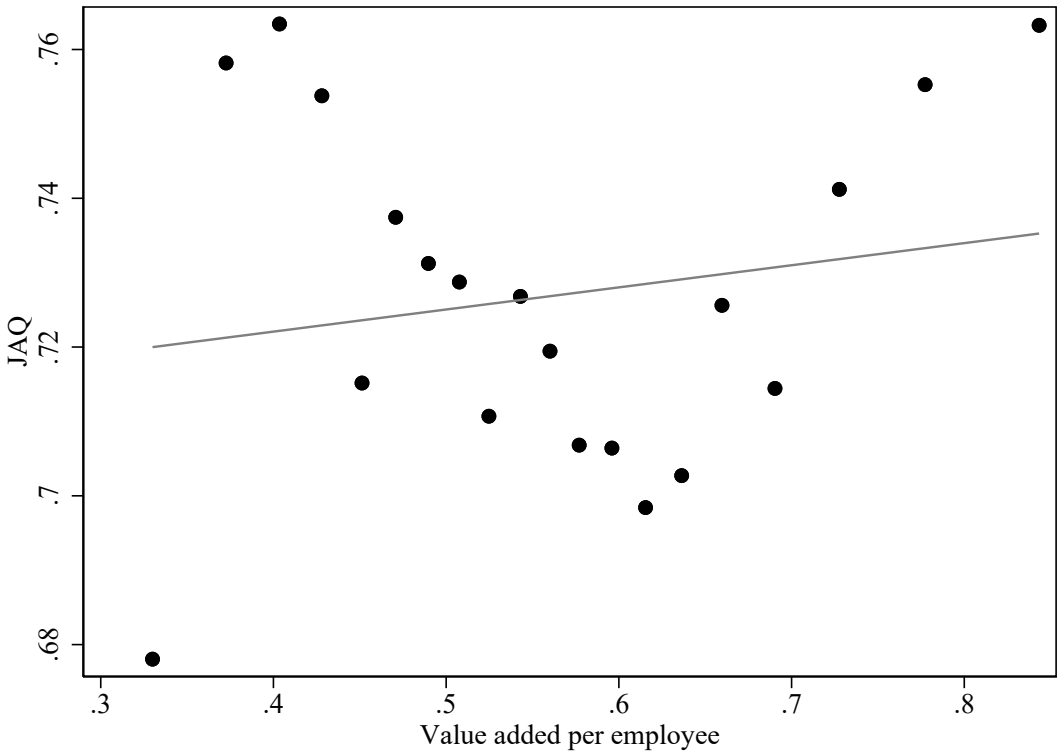
dep.: FJAQ4	Coef.	Std. Error	T-stat	$p-value$	95% conf.	interval
Productivity Decile						
1	−0.102	0.020	−5.190	0.000*	−0.141	−0.064
2	−0.019	0.014	−1.330	0.185	−0.047	0.009
3	−0.029	0.014	−2.130	0.033*	−0.056	−0.002
4	−0.007	0.013	−0.570	0.567	−0.033	0.018
5	−0.021	0.012	−1.690	0.091*	−0.046	0.003
6	−0.014	0.012	−1.110	0.266	−0.038	0.011
7	−0.027	0.013	−2.150	0.031*	−0.052	−0.002
8	−0.015	0.012	−1.220	0.223	−0.038	0.009
9	−0.004	0.011	−0.360	0.721	−0.026	0.018
(f) Size	−6.9E-5	1.56E-5	−4.410	0.000*	−9.97E-5	3.84E-5
(f) TA	2.24E-5	9.84E-06	2.270	0.023*	3.07E-06	4.17E-5
year			(omitted)			
(f) Industry			(omitted)			
constant			(omitted)			
N. Obs (firms): 4130			$R^2 = 0.48$			

Figure 3.7: FJAQ4 against productivity for both top-firms and non top-firms sets. The positive correlation is evident in the control set, while it is rather unclear in the learning set.

(a) FJAQ4 vs Productivity; Top-firms set.



(b) FJAQ4 vs Productivity; Top-firms set.



3.7 Conclusion

In this chapter we proposed an application of supervised learning techniques to Labor Economics. The problem of interest was that of estimating optimal allocation rule for employees-to-tasks assignment. We proposed to approach the problem as a classification task. This perspective has the advantage of being model-free. The allocation rule is estimated on a subset of observed data (top-firms), that can be reasonably thought as adopting an optimal allocation rule.

Moreover, we introduced different measures to evaluate assignation quality against a benchmark rule. These captures different aspect of workforce allocation. Some of them, are computationally intensive.

We applied out methodology to the Statistics Sweden LISA database, a rich dataset matching information on employers and employees. Overall, the empirical results suggest the validity of our approach. We find that it is possible to construct measures of deviation from an optimal allocation rule and that firms who deviate the most are even those that are less productive.

The proposed approach provides a different perspective to look at productivity drivers. This methodology can also be used to predict the most probable allocation of a new employee to jobs according to the top performing firms in the economy.

Concluding, there are several possible improvements and other unexplored uses for our methodology:

1. The overall quality of the whole procedure heavily relies on the selection of top-firms. Further refinements of the selection procedure may improve the results and reliability of estimates.

2. Due to the limitations exposed above, in principle, there is still room to improve algorithmic performances with a more careful tuning process. Also, other algorithms and estimation schema could be added for further comparisons.

3. The theoretical framework needs further development. In addition, given the promising behaviour of the constructed measure, it would be interesting to investigate further causal relationship between productivity and constructed measures.

3.8 Appendix Chapter 3

Calibration

The problem of *calibration* is that of refining the estimated class probabilities of a classifier. That is, given a set of estimated and observed probabilities, $\{(\hat{p}_1, p_1), \ldots, (\hat{p}_n, p_n)\}$, where, in our notation,

$$\hat{p}_i = \left[\hat{f}_1(x_i, x_z), \ldots, \hat{f}_{|\mathcal{Y}|}(x_i, x_z)\right], \quad p_i = \left[\, \mathbb{1}\{y_i = 1\}, \ldots, \mathbb{1}\{y_i = |\mathcal{Y}|\} \,\right]$$

we want to find a mapping $\hat{p} \mapsto \phi(\hat{p})$, so as to minimize some loss function. There are several choices for the latter, and a popular one in classification methods is the so called Brier Score (i.e. mean square error):[22]

$$\text{BS} := \frac{1}{n} \sum_{i=}^{n} \frac{\|\phi(\hat{p}_i) - p_i\|_2^2}{n}.$$

Intuitively, better calibrated probabilities better estimates of class conditional probabilities, and one can be more confident on the resulting class ranking. To see why this problem is relevant in classification, imagine estimation of class probabilities by vote averaging in random forests If the true class probability is 1, the only way of getting such an estimated value would be that all trees in the forest estimated the correct class, which highly unlikely. Calibration tries to cope exactly with this problem (see Hastie, Tibshirani, and Friedman, 2009).

Most of the times, poor class probability estimates are due to the fact they are not the target of the classifier (rather, we are interested in classifying points), but come as a byproduct of the estimation (Subsection 1.3.2).

Regarding the methods discussed above, the multinomial logistic regression exactly estimates class probabilities (by maximum likelihood) and the classification is done according to these probabilities. Thus, the multinomial logistic regression is is a well calibrated model. Niculescu-Mizil and Caruana, 2005 and Boström, 2008 find by empirical studies that random forests perform better at estimating class probabilities if calibrated. However, due to the additional computational overhead[23] (not feasible in our case), we do not calibrate in this case. Note that un-calibrated random forests should also perform better at classification task.

Finally, we decided to calibrate bagged trees. These are known to be well calibrated models by default (Niculescu-Mizil and Caruana, 2005). While there is general consensus that bagging improves probability estimations by decision trees, there is contrasting evidence on whether to calibrate or not these models. Provost and Domingos, 2003 argue that both bagging and Laplace calibration remarkably improve probability estimation (for binary problems). They also recommend not pruning. Similar evidence was found in Ferri, Flach, and Hernández-Orallo, 2003. Based on their findings, we decided to use bagged unpruned trees with Laplace smoothing due to their superior performances. Then, let T be the number of bagged trees, $|\mathcal{Y}|$ be the number of classes, $nh_t(x)$ and $nh_t(x)^{(j)}$ be the number of points and the number of points of

[22] This measure was introduced for multi-class classification in Brier, 1950. It must be noted that the term calibration is more general than how we use it here, and refers to different problems (Bella et al., 2010).

[23] The type of calibration considered for random forests in the cited studies implies estimating calibration parameters on a validation set.

class j, in the terminal node in tree t where x fall. The calibrated bagged trees estimates the following:

$$\hat{P}(y^{(\text{new})} = j|x^{(\text{new})}) = \frac{1}{T}\sum_{t=1}^{T}\frac{nh_t^{(j)}(x^{(\text{new})}) + 1}{nh_t(x^{(\text{new})}) + |\mathcal{Y}|}. \tag{3.18}$$

We decided for this calibration method because, while being effective (even if outperformed by other methods, e.g. Ferri, Flach, and Hernández-Orallo, 2003), it is simple and easy to deploy. Also, there is essentially no additional overhead in estimation time.

Evaluation Methods

In order to evaluate classifiers, we decided to use relatively simple metrics [24] as the accuracy score and the F1-score, to evaluate classification performances. The accuracy score is simply defined as the ratio of the correctly classified points as:

$$\text{Acc.} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}\left\{\hat{f}(x_i, z_i) = y_i\right\}, \tag{3.19}$$

(where the set of points where we compute Acc. may vary). This measure simply takes into account the proportion of correctly classified points and its generalization to multi-class problems is straightforward. However, there are problems with this metrics. Consider for example a binary classification problem with 95% points from one class. A constant classifier predicting always that class would achieve 95% accuracy, even if the classifier itself is bad or does not uses data information at all. Thus, accuracy is problematic with highly unbalanced data.

For this reason, we also provide the precision, recall and F1-score metrics. To extend these measure to the multi-class problem, we define each of them class-wise, considering for each class a one-vs-others approach:

$$\text{precision}_i := \frac{\sum_{i=1}^{n}\mathbb{1}\left\{\hat{f}(x_i, z_i) = i\right\}\mathbb{1}\{y_i = i\}}{\sum_{i=1}^{n}\mathbb{1}\left\{\hat{f}(x_i, z_i) = i\right\}}; \tag{3.20}$$

$$\text{recall}_i := \frac{\sum_{i=1}^{n}\mathbb{1}\left\{\hat{f}(x_i, z_i) = i\right\}\mathbb{1}\{y_i = i\}}{\sum_{i=1}^{n}\mathbb{1}\{y_i = i\}}; \tag{3.21}$$

$$\text{F1}_i := \frac{2 \cdot \text{precision}_i \cdot \text{recall}_i}{\text{precision}_i + \text{recall}_i} \tag{3.22}$$

(so that measures that refer to class i consider the classification as a binary problem: class i against classes $\{1, \ldots, i-1, i+1, |\mathcal{Y}|\}$). Precision measures the ability of the classifier to classify only the true instances as being true. At the numerator we have points correctly classified to class i (true positives); at the denominator we have all point classified by the algorithm as belonging to class i (whether this is true or not; these is the sum of the true positives and the false positives). Recall measures the ability of the classifier to retrieve the instances of class i correctly. At the

[24] Tuning and evaluating classifiers was not an easy task. First, we deal with a multi-class classification problems, and all the usual binary evaluation metrics need to be adapted or modified (and are in general more complicated). Secondly, the huge time needed to deploy and fit any of the algorithms forced us to discard any methodology based on Cross-Validation or resampling. Thirdly, the memory limitations prevented us to extract the fitted classifiers and to produce elaborated evaluation methods.

numerator we have points correctly classified to class i (true positives); At the denominator we have all the points belonging to class i (true positive plus false negatives). The F1-score is the harmonic mean of the two. As an example, a constant classifier (classifying to i) would have high recall but poor precision; on the other hand, a classifier with extremely high precision may have poor recall. There is usually a trade-off. These measure are reviewed, together with others, in Sokolova and Lapalme, 2009. Also, the authors point out that F score measures might be a more sensible choice for high unbalanced data and might be more appropriate in a one-vs-others approach.

We also provide two aggregated measure for the F1-score. The first one is a sample average across classes, *F1-macro*; the second one is a weighted average of the scores across classes, weighted by classes prior probabilities (i.e. the amount of points belonging to the classes), *F1-weighted*.

These measures are all bounded in $[0, 1]$, where higher values means better performances.

Finally, to evaluate class probability estimation, we use the Brier score defined as:

$$\text{BS} := \sum_{(i,k)} \sum_{j=1}^{|\mathcal{Y}|} \frac{\left(\hat{f}_j(x_i, z_k) - \mathbb{1}\{y_i = j\}\right)^2}{\sum_{(i,k)} 1}, \tag{3.23}$$

where $\sum_{(i,k)}$ indicates the sum over all points in the considered set and $\sum_{(i,k)}$ is the total number of points. This measure ranges in $[0, +\infty)$, where values closer to 0 implies better estimates.[25]

Given these evaluation metrics, tuning was performed in the following way. At each run on the server, we randomly split the learning set in a training set and a test set, of sizes 90% and 10% respectively. We used the results on the test set to aid the choice of parameters for the next run. We note that this was possible because the classifiers used (primarily random forests, multinomial logit and bagged trees) do not require excessive parameter tuning. The test set was thus used to avoid overfitting problems. Note also that, since the parameters are tuned on a test set from a previous iteration, the test set of the subsequent run constitutes a valid set to evaluate classifiers' final performances. The relative few percentage sampled for test set are motivated by generally massive data size. This also lessen the possible gains from cross-validation procedures as well as the need for calibration in case of bagged trees (Ferri, Flach, and Hernández-Orallo, 2003).

[25] In the case of a classifier that predicts $|\mathcal{Y}|$ classes equally at random, the expected value of this score is $\frac{|\mathcal{Y}|-1}{|\mathcal{Y}|} = 0.99$, for $|\mathcal{Y}| = 100$.

References

Abowd, J. M. et al. (2005). "The relation among human capital, productivity, and market value: Building up from micro evidence". In: *Measuring capital in the new economy*. University of Chicago Press, pp. 153–204.

Adhvaryu, A., Kala, N., and Nyshadham, A. (2019). *Management and shocks to worker productivity*. Tech. rep. National Bureau of Economic Research.

Amemiya, T. (1985). *Advanced econometrics*. Harvard university press.

Bella, A. et al. (2010). "Calibration of machine learning models". In: *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*. IGI Global, pp. 128–146.

Bernard, A. B. and Jones, C. I. (1995). "Comparing apples to oranges: productivity convergence and measurement across industries and countries". In: *The American Economic Review*, pp. 1215–1238.

Bloom, N. et al. (2019). "What drives differences in management practices?" In: *American Economic Review* 109.5, pp. 1648–83.

Boström, H. (2008). "Calibrating random forests". In: *2008 Seventh International Conference on Machine Learning and Applications*. IEEE, pp. 121–126.

Breiman, L. (1996). "Bagging predictors". In: *Machine learning* 24.2, pp. 123–140.

— (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

— (2017). *Classification and regression trees*. Routledge.

Brier, G. W. (1950). "Verification of forecasts expressed in terms of probability". In: *Monthly weather review* 78.1, pp. 1–3.

Bühlmann, P. and Yu, B. (2002). "Analyzing bagging". In: *The Annals of Statistics* 30.4, pp. 927–961.

Costa, J. E. R. I. (1988). "Managerial task assignment and promotions". In: *Econometrica: Journal of the Econometric Society*, pp. 449–466.

Crawford, V. P. and Knoer, E. M. (1981). "Job matching with heterogeneous firms and workers". In: *Econometrica: Journal of the Econometric Society*, pp. 437–450.

Dietterich, T. G. (2000). "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization". In: *Machine learning* 40.2, pp. 139–157.

Eeckhout, J. (2018). "Sorting in the labor market". In: *Annual Review of Economics* 10, pp. 1–29.

Efron, B. (1979). "Bootstrap Methods: Another Look at the Jackknife". In: *The Annals of Statistics* 7.1, pp. 1–26. DOI: 10.1214/aos/1176344552. URL: https://doi.org/10.1214/aos/1176344552.

Efron, B. and Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC press.

Eiselt, H. A. and Marianov, V. (2008). "Employee positioning and workload allocation". In: *Computers & operations research* 35.2, pp. 513–524.

Ferri, C., Flach, P., and Hernández-Orallo, J. (2003). "Decision trees for ranking: effect of new smoothing methods, new splitting criteria and simple pruning methods". In: *Technical report, DSIC 2003*.

Fox, J. T. and Smeets, V. (2011). "Does input quality drive measured differences in firm productivity?" In: *International Economic Review* 52.4, pp. 961–989.

Greene, W. H. (2003). *Econometric analysis*. Pearson Education India.

Hancock, T. et al. (1996). "Lower bounds on learning decision lists and trees". In: *Information and Computation* 126.2, pp. 114–122.

Hastie, T., Tibshirani, R. J., and Friedman, J. (2009). *The Elements of Statistical Learning*. 2nd ed. Springer New York. DOI: 10.1007/978-0-387-84858-7. URL: https://doi.org/10.1007/978-0-387-84858-7.

Kok, J. de, Brouwer, P., and Fris, P. (2005). *Can firm age account for productivity differences?* Scales Research Reports N200421. EIM Business and Policy Research.

Kuhn, H. W. (1955). "The Hungarian method for the assignment problem". In: *Naval research logistics quarterly* 2.1-2, pp. 83–97.

Laurent, H. and Rivest, R. L. (1976). "Constructing optimal binary decision trees is NP-complete". In: *Information processing letters* 5.1, pp. 15–17.

Lazear, E. P. and Oyer, P. (2007). *Personnel economics*. Tech. rep. National Bureau of Economic Research.

Lazear, E. P., Shaw, K. L., and Stanton, C. T. (2015). "The value of bosses". In: *Journal of Labor Economics* 33.4, pp. 823–861.

Liaw, A. and Wiener, M. (2002). "Classification and Regression by randomForest". In: *R News* 2.3, pp. 18–22. URL: https://CRAN.R-project.org/doc/Rnews/.

McFadden, D. (1973). "Conditional logit analysis of qualitative choice behavior". In:

Micci-Barreca, D. (2001). "A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems". In: *ACM SIGKDD Explorations Newsletter* 3.1, pp. 27–32.

Morgan, J. N. and Sonquist, J. A. (1963). "Problems in the analysis of survey data, and a proposal". In: *Journal of the American statistical association* 58.302, pp. 415–434.

Munkres, J. (1957). "Algorithms for the assignment and transportation problems". In: *Journal of the society for industrial and applied mathematics* 5.1, pp. 32–38.

Murthy, S. K. (1998). "Automatic construction of decision trees from data: A multi-disciplinary survey". In: *Data mining and knowledge discovery* 2.4, pp. 345–389.

Niculescu-Mizil, A. and Caruana, R. (2005). "Predicting good probabilities with supervised learning". In: *Proceedings of the 22nd international conference on Machine learning*. ACM, pp. 625–632.

Olsson, M. and Tåg, J. (2017). "Private equity, layoffs, and job polarization". In: *Journal of Labor Economics* 35.3, pp. 697–754.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Provost, F. and Domingos, P. (2003). "Tree induction for probability-based ranking". In: *Machine learning* 52.3, pp. 199–215.

Quarteroni, A., Sacco, R., and Saleri, F. (2010). *Numerical mathematics*. Vol. 37. Springer Science & Business Media.

R Core Team (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. URL: https://www.R-project.org/.

Robnik-Sikonja, M. and Savicky, P. (2018). *CORElearn: Classification, Regression and Feature Evaluation*. R package version 1.53.1. URL: https://CRAN.R-project.org/package=CORElearn.

Rokach, L. and Maimon, O. (2005). "Top-down induction of decision trees classifiers-a survey". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 35.4, pp. 476–487.

Rosen, S. (1982). "Authority, control, and the distribution of earnings". In: *The Bell Journal of Economics*, pp. 311–323.

Safavian, S. R. and Landgrebe, D. (1991). "A survey of decision tree classifier methodology". In: *IEEE transactions on systems, man, and cybernetics* 21.3, pp. 660–674.

Sokolova, M. and Lapalme, G. (2009). "A systematic analysis of performance measures for classification tasks". In: *Information processing & management* 45.4, pp. 427–437.

Steiglitz, K. (1998). *Combinatorial optimization: algorithms and complexity*. Dover Publications.

Syverson, C. (2011). "What determines productivity?" In: *Journal of Economic literature* 49.2, pp. 326–65.

Tåg, J., Åstebro, T., and Thompson, P. (2016). "Hierarchies and entrepreneurship". In: *European Economic Review* 89, pp. 129–147.

Wright, M. N. and Ziegler, A. (2017). "ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R". In: *Journal of Statistical Software* 77.1, pp. 1–17. DOI: 10.18637/jss.v077.i01.