UNIVERSITY OF TURIN

DOCTORAL THESIS

Use of Graphical Models in the field of Dimensionality Reduction

Development of the new algorithms for automatic feature selection and the evaluation of the drift.

Author:
Luigi Riso

Supervisor:
Prof. Marco Guarzoni

Coordinator:
Prof. Laura Sacerdote

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy in the:

Modelling And Data Science

March 25, 2021

to all women of my life.

Preface

The aim of this thesis is to investigate the main strategies to implement a dimensional reduction. In the fields of data mining or in general, the world of data analysis, the methods of dimensionality reduction have become an indispensable tool. The present digital revolution offers the researcher very rich data-sets in data analysis which can greatly improve both prediction capability and the analysis of specific variables of interest. Traditional methods like principal component analysis and classical metric multidimensional scaling are put under the pressure by this revolution. Therefore, to overcome this issue, different methods have been developed to reduce the data dimensionality in a nonlinear way.

New methods that account for this increase include, for example, the use of graphs to maps the manifold topology and the use of new metrics like the geodesic distance. In addition, new optimization schemes, based on kernel techniques and spectral decomposition, have led to spectral. In addition to this rapid growth, it is important to mention new optimization schemes based on kernel transformations and spectral decomposition. In fact, these methods have led to spectral embedding, which encompasses many of the recently developed methods. In this vein, this thesis pays attention also to distinguish the methods that perform a dimensionality reduction through a transformation of the variables, from the methods that adopt a features selection. In this category, we introduce the use of the Graphical Models as a technique to discern the most relevant variables from the irrelevant ones. In details, the Graphical Model is a technique for mapping the relationship between the variables and as representations of hierarchical Bayesian models. These methods allow managing large data set through a graph. The nodes in the graph are identified with the random variables of the data set, and joint probability distribution are used to build the edges between the nodes.

All dimensionality reduction methods can be viewed as solving an optimization problem, over a matrix of data. The application of one of these methods will depend on the opportunity to adapt at a very large dataset or the type of variables contained in the dataset and finally by the trade-off between dimensional reduction and loss of information from the variables. In order to respect these specifications, we have developed a new algorithm that takes advantage of the Graphical Models properties. This algorithm produces an automatic variable selection and belongs to Sequential Forward Search. In fact, it's search starts with an empty set and keeps on adding features.

Considering the additional dimension of time to the dataset, the dimensional reduction channels another particular problem, namely the change of relationship between the variables over time. This phenomenon is known as concept- or model- drift and describes the situation in which there exists a hidden context of data generative structure, that is an effect of the outcome variable not captured by the model features, which changes over time abruptly, incrementally, or periodically.

Finally, we applied these methods at the contest of survival analysis. The purpose of this strategy is to investigate the economic performance of new firms born during the 2009 crisis with a focus on survival and value creation. Since it leverages on a very large dataset traditional econometrics techniques require an ex-ante sound variable selection to avoid excessive loss of relevant information. The most important vantage of this approach is that we can identify the most relevant variables for each year. These selections are justified by the presence of the drift, computed with the support of the Graphical Models.

For the sake of clarity, this thesis is composed of chapters that are independent papers. Thus, some sections could be repeated.

Use of Graphical Models in the field of Dimensionality Reduction.

Development of the new algorithms for automatic feature selection and the evaluation of the drift.

by

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy

at

University of Turin
March, 2021

Contents

List of Figures

List of Tables

List of Abbreviations

PCA	**P**rincipal **C**ompoennent **A**nalysis
FA	**F**actor **A**nalysis
LDA	**L**inear **D**iscriminant **A**nalysis
LLE	**L**ocal **L**inear **E**mbedding
NLM	**N**on **L**inear **M**ethod
SOM	**S**elf **O**rganization **M**aps
kPCA	**K**ernel **PCA**
LE	**L**aplacian **E**igenmaps
DAG	**D**irected **A**cyclic **G**raph
UGGM	**U**ndirected **G**aussian **G**raphical **M**odel
OLS	**O**rdinary **L**east **S**quares
PS	**P**ath **S**teps
AM	**A**djacency **M**atrix
TM	**T**ransition **M**atrix
ty	**t**ransition **y**ears
RSS	**R**esidual **S**um **S**quare
MSE	**M**ean **S**quare **E**rror

Chapter 1

Dimensional Reduction: A Review

The paper discusses the major contributions in the field of dimensionality reduction. More specifically, it focuses on traditional methods both linear and non-linear for reducing the information overload of dataset. However, the availability of large data set, big data in some cases, suggests that a viable alternative is an ex-ante reduction of the variables rather than a transformation such as in Principal Component Analysis or mapping on a lower-dimensional space such as in Self Organizing Map. In this vein, the paper suggests that graphical models can be a feasible method for feature selection.

1.1 Introduction

The availability of the *Big Data* requires new techniques, which can handle not only a large number of observations, but also rich data sets in terms of number and relations among variables. One of the challenges for the statistician is to infer from these data set the relationship between the variables. The high-dimensional data indicates to data set having a large number of attributes/ features. The reduction of dimensionality becomes a crucial aspect to handle these datasets. For this reason, the *dimensional reduction* becomes crucial to transform the high-dimensional data into a lower dimensionality. The intrinsic dimensionality of data is the minimum number of parameters needed to account for the observed properties of the data [53]. As the main goal, dimensionality reduction allows for compression of the data through classification, visualization and, regression. And in this context the *Graphical Models* becomes important not only to mapping the relationships between the variables, but also to do feature selection [78]. Dimensional reduction is important in many domains, since it mitigates the curse of dimensionality and other undesired properties of high-dimensional spaces [76]. Tendentially before the availability of the big data, linear techniques such as Principal Component Analysis (PCA) and Factor Analysis (FA) was used to implement dimensionality reduction [116]. In the last decades, the literature has proposed a large number of nonlinear techniques for dimensionality reduction, in order to implement

a dimensional reduction also in the high dimensional space. [100]. In this view, the functionality of this paper is twofold:

- to investigate and to compare the different methods to do dimensional reduction [144];
- to introduce the Graphical Models as a method to do the dimensionality reduction.

In the following sections, first, we summarize the dimensional reduction strategies (section 1.2), then we analyze the different algorithms for each group, according to the classification proposed in the figure 1.3. In particular, in the sections 1.3,1.4,1.5 and 1.6 we do not aim for an exhaustive explanation to every method but rather to provide a a general idea about these methods.

1.2 Background: notation and basic definitions

Broadly speaking, a dataset is a rectangular matrix $\mathbf{X}$, represented in $n \times p$, where n is the number of the observation and p the number of the variables. In other words, This matrix $\mathbf{X}$ is a collection of n data data points, $\{\mathbf{x}\}_{i=1}^{n}$ in the dimensional Euclidean space $\mathbb{R}^p$. The high-dimensional space will depend by the dimension of p. Feature selection is the strategy to select only those feature form the data set $\mathbf{X}$, which are relevant or significant from the point of view of classification or clustering, the less important features are discarded by it [114] [144]. Which regard the transformation strategies of the data, the following issues must be considered:

- can be used different measure to implement the dimensional reduction (like dependency, relevance, significance, mutual information)
- missing and incomplete data (the real-world dataset mostly have some of the values that are missing or are incomplete, and some techniques require the complete dataset)
- variables of interest (discrete or continuous data, some of the dimensional reduction techniques can handle the continuous-valued data directly while some require discretization)

The purpose of dimensional reduction is to do feature selection or transformation of the variables with some well-defined properties. Indeed, these techniques present a new dataset $\mathbf{Y}$ with k variables instead of the original dataset $\mathbf{X}$ with dimensionality p, where $k < p$ and often $k \ll p$. This strategy preserves the local geometry of the data as much as possible such as distances or angles between data points [27]. In this research, we denote a high-dimensional data point by $\mathbf{x}_i$, where $\mathbf{x}_i$ is the ith row of the p-dimensional dataset $\mathbf{X}$. As shown by the author [100], the low-dimensional counterpart of $\mathbf{x}_i$ is denoted by $\mathbf{y}_i$, where $\mathbf{y}_i$ is the ith row of the k-dimensional data matrix $\mathbf{Y}$. Basically, there are two strategies to achieve matrix $\mathbf{Y}$ from matrix $\mathbf{X}$:

- feature extraction- *transforming the existing features into a lower dimensional space* [142]
- feature selection- *selecting a subset of the existing features without a transformation* [45].

There are different fields of technology or science where high-dimensional data are typically encountered, here we will mention a few.
Processing of sensor array, that represents all applications where it is used a set of several identical sensors. In this class belong numerous biomedical applications, such as electrocardiogram or electroencephalograph acquisition, where several electrodes record time signals at different laces on the chest or the scalp. Seismography and weather forecasting are also another example of these collections of data [131]. Another class of application of high-dimensional data is the *Image processing*, where a picture is considered as the output of a digital camera. However, in the last decade, image processing is often seen as a standalone domain. In fact, with the increasing development of deep learning, image processing became an important element of this research field [153].
Multivariate data analysis is another example of using a high-dimensional dataset. The main difference with the previous classes is that in multivariate data analysis, the data come from different types of sensors and focuses on the analysis of measures that can be related to each other. A clear example is a car, wherein the gearbox connecting the engine to the wheels has to take into account information from rotation sensors (wheels and engine shaft), force sensors (brake and gas pedals), position sensors (gearbox stick, steering wheel), temperature sensors (to prevent engine overheating or to detect glaze), and so forth [80] [62]. In the contest of high-dimensional for completeness, it should be mentioned *Data mining*. At first sight, multivariate data analysis, and data mining seem to be very close. Actually the former is a classical sub-domain of statistics, indeed data mining has a broader scope of applications than multivariate data analysis. Data mining can deal with more exotic data structures than arrays of numbers. For example, data mining encompasses text mining. The analysis of large sets of text documents aims, for instance, at detecting similarities between texts, like common vocabulary, the same topic, etc [14]. If these texts are Internet pages, hyperlinks can be encoded in graph structures and analyzed using tools like graph embedding. Cross-references in databases can be analyzed in the same way [95].

1.2.1 Theoretical motivations

Bellman coined the term "*curse of dimensionality* " in [21] in connection with the difficulty of optimization enumeration on product spaces. Bellman claims the fact that considering a Cartesian grid of spacing $1/10$ on the unit cube in 10 dimensions, the number of points equals 10^{10}; for a 20-dimensional cube, the number of points further increases to 10^{20}. According to [21] interpretation: if the goal consists of optimizing a function over a continuous domain of a few dozen variables by exhaustively searching a discrete search space defined by a crude discretization, one could easily be faced

with the problem of making tens of trillions of evaluations of the function. In other words, the curse of dimensionality also refers to the fact that in the absence of simplifying assumptions, the number of data samples required to estimate a function of several variables to a given accuracy (i.e., to get a reasonably low-variance estimate) on a given domain grows exponentially with the number of dimensions. This fact, responsible for the curse of dimensionality, is often called the "*empty space phenomenon*" [129]. Because the amount of available data is generally restricted to a few observations, high-dimensional spaces are inherently sparse. More concretely, the curse of dimensionality and the empty space phenomenon give unexpected properties to high-dimensional spaces [128]. The increase of the dimensional represents a limitation in the geometric representation and consequently in the managed of the data. In fact three-dimensional objects can also be sculpted or carved, but when there are more than three dimensions it becomes hard to represent them in 2 dimensions. Nevertheless, several techniques exist to solve this problem: use different colours for some dimensions or multiple linear projections.

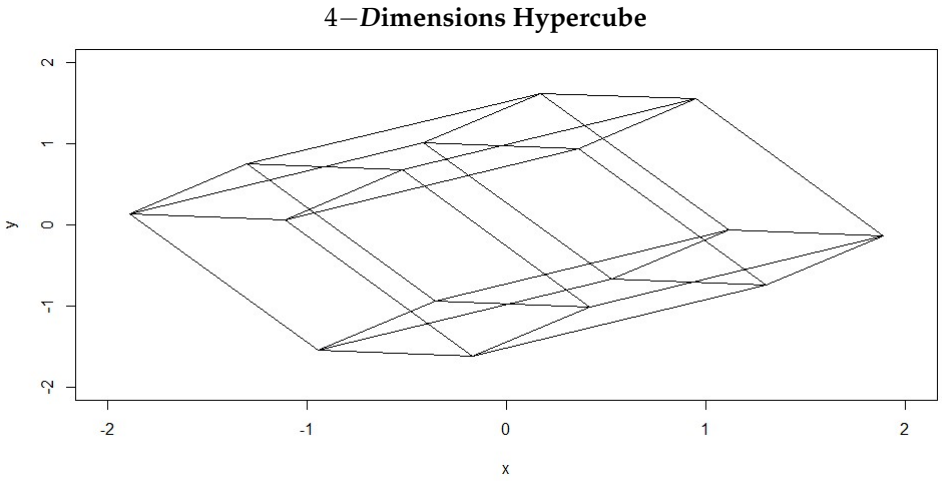


Figure 1.1: Projection of cube 4 dimensional in 2 dimensions

Figure 1.1 shows the projection of a $4D$ cube that has been projected on a plane in a linear way, wherewith D indicates the dimension of the dataset, in other words, the numbers of the variables p present in the dataset. In D-dimensional space, a sphere and the corresponding circumscripted cube (all edges equal the sphere diameter) lead to the following volume formulas:

$$V_{sphere}(r) = \frac{\pi^{D/2} r^D}{\Gamma(1 + D/2)} \tag{1.1}$$

$$V_{cube}(r) = (2r)^D \tag{1.2}$$

where r is the radius of the sphere. Surprisingly, the ratio V_{sphere}/V_{cube} tends to zero when D increases:

$$\lim_{D \to \infty} \frac{V_{sphere}(r)}{V_{cube}(r)} = 0$$

Intuitively, this means that as dimensionality increases, a cube becomes more and more spiky, like a sea urchin: the spherical body gets smaller and smaller while the number of spikes increases, the latter occupying almost all the available volume, as Figure 1.2 shows.

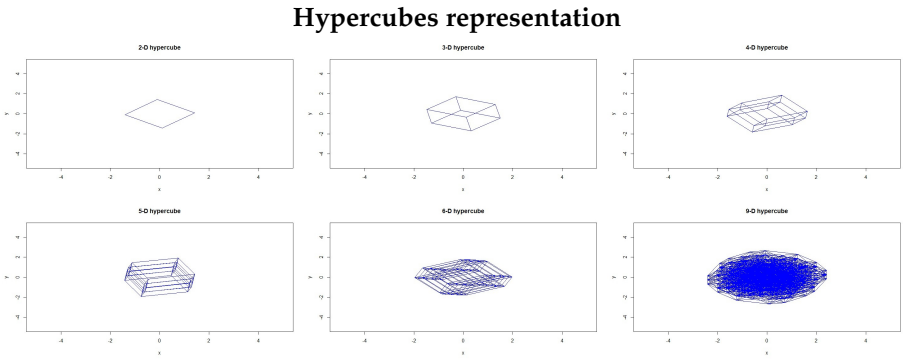


Figure 1.2: Evolution of hypercube projections in different dimensions

If we assign the value $1/2$ to r, V_{cube} will equal to 1, leading to

$$\lim_{D\to\infty} V_{sphere}(r) = 0 \tag{1.3}$$

This indicates that the volume of a sphere vanishes when dimensionality increases. By the Equation 1.1, the relative hypervolume of a thin spherical shell is:

$$\frac{V_{sphere}(r) - V_{sphere}(r(1-\epsilon))}{V_{sphere}(r)} = \frac{1^D - (1-\epsilon)^D}{1^D}, \tag{1.4}$$

where $\epsilon \ll 1$ is the thickness of the shell. When D increases, the ratio tends to 1, meaning the shell contains almost all the volume[148]. Regardless of the dimension D of the dataset with all continuous variables, the probability density function of an isotropic Gaussian distribution is written as:

$$f_{\mathbf{x}}(x) = \frac{1}{\sqrt{(2\pi\sigma^2)}^D} \exp\left(-\frac{1}{2}\frac{||\mathbf{x}-\mu_x||^2}{\sigma^2}\right) \tag{1.5}$$

with $\mathbf{x}$ is a D-dimensional vector, $\mu_{\mathbf{x}}$ represents the D-dimensional mean, while σ^2 is the isotropic variance. If we suppose that the random vector $\mathbf{x}$ has zero mean and unit variance, the equation will simplify into:

$$f_{\mathbf{x}}(x) = K(r) = \frac{1}{\sqrt{(2\pi)}^D} \exp\left(-\frac{r^2}{2}\right) \tag{1.6}$$

where $r = ||\mathbf{x}||$ can be interpreted as a radius. Indeed, because the distribution is isotropic, equiprobable contours are spherical. In the context of high-dimensional data, three possibilities exist to avoid or at least attenuate the effect of this phenomenon. We can focus on the separation between relevant and irrelevant variables. The second one

concentrates on the dependencies between (relevant) variables. Eventually, selecting the best subset of variables that better can explain or predict the variables of interest. In the first case, for example in a class of *multivariate data*, not necessarily all variables are related to underlying information the user wishes to catch. If we have a redundancy of information, we may eliminate from the dataset irrelevant variables. There are different techniques to distinguish relevant variables from irrelevant ones. The most used strategy to identify the relevance of the information, between the variable of interest than other ones presents in the data-set, is to compute the correlation matrix. For instance, the input variables that are not correlated with the outputs may then be eliminated. Even when assuming that all variables are relevant, the dimensionality of the observed data may still be larger than necessary. In that case, rather than arbitrarily removing some variables, another way to reduce the number of variables would be to find a new set of transformed variables, called *projections*. This is motivated by the fact that dependencies between variables may be very complex and that keeping one of them might not suffice to catch all information content they both convey [95]. The advantage of this second approach is that the new dataset should obviously be smaller than the start dataset but despite the reduction of dimensionality, the new one set of data preserves the interesting characteristics. These properties must ensure that the transformation does not alter the information content conveyed by the initial data set, but only represents in a different form. For example, if the given variables are assumed to be mixtures of a few unobserved ones, then a projection that inverts the mixing process is very useful. In other words, through the transformation, we can track and eliminate dependence between the observed variables. Another important task of a projection is to retrieve information from the latent variables, i.e., those that are at the origin of the observed ones but cannot be measured directly. This task, in its most generic form, is often called variable separation. In the points of weakness of the projection, there is the impossibility to infer directly on the variables presented in the dataset. Despite the projection that can explain a large part of the variance, this technique loses some information from the original variables. To avoid this problem, in the last decade some techniques from the machine learning field were developed [111]. This approach can be used to find from the original dataset the best subset of variables. In that way, we do a dimensional reduction without lost information from the variables. Furthermore, this approach can be combined with the projection. In other words, we can automatically remove the irrelevant variables and focus on the latent variables.

1.2.2 Topology, space and manifolds

From a geometrical point of view, when two or more variables depend on each other, the support of their joint distribution does not span the whole space. The dependence induces some structure in the distribution, in form of the geometrical locus that can be seen as a kind of object in the space. The hypercube illustrated in Figure 1.1 is an example of such a structure. One of the dimensional reduction aims is to give a new representation of these objects while preserving their structure [95]. The topology

studies the properties of objects that are preserved through deformation, twisting and stretching. One of the central ideas of topology is that the spatial objects like circles and sphere can be treated as objects in their own right: the knowledge of objects does not depend on how they are represented, or *embedded*, in the space. Topology is used to abstract the intrinsic connectivity of objects while ignoring their detailed form. Two objects that have the same topological properties are called *homeomorphic*. The topological objects are formally defined as topological spaces. A *topological space* is a set for which a *topology* is specified [105]. For a set $\mathcal{Y}$ a topology T is defined as a collection of subsets of $\mathcal{Y}$ that obey the following properties:

- $\emptyset \in T$ and $\mathcal{Y} \in T$;
- whenever two sets are in T, then so is their intersection;
- whenever two or more sets are in T, then so is their union.

This definition of a topology holds as well for a Cartesian space $\mathbb{R}^D$ as for the graph. From a geometrical point of view, a topological space can also be defined using neighborhoods and Haussdorf's axioms. The neighborhood of a point $x \in \mathbb{R}^D$, also called a $\epsilon-$neighborhood or infinitesimal open set, is often defined as the open $\epsilon > 0$ and centered on x. A set containing an open neighborhood is also called a neighborhood. Then, a topological space is such that:

- to each point x there corresponds at least one neighborhood $\mathcal{U}(x)$ and $\mathcal{U}(x)$ contains x;
- if $\mathcal{U}(x)$ and $\mathcal{V}(x)$ are neighborhoods of the same point x, then a neighborhood $\mathcal{W}(x) \subset \mathcal{U}(x) \cup \mathcal{V}(x)$;
- if $z \in \mathcal{U}(x)$, then a neighborhood $\mathcal{V}(z)$ of z exists such that $\mathcal{V}(z) \subset \mathcal{U}(x)$;
- for two distinct point, two disjoint neighborhoods of these points exist.

Within this framework, a topological *manifold* $\mathcal{M}$ is a topological space that is locally Euclidean, meaning that around every point of $\mathcal{M}$ is a neighborhood that is topologically the same as the open unit ball in $\mathbb{R}^D$. Commonly, the unqualified term manifold means "manifold without boundary". An *embedding* is a representation of a topological object in a certain space, usually $\mathbb{R}^D$ for some D, in such a way that its topological properties are preserved. For example, the embedding of a manifold preserves open sets. More generally, a space $\mathcal{X}$ is embedded in another space $\mathcal{Y}$ when the properties of $\mathcal{Y}$ restricted to $\mathcal{X}$ are the same as the properties of $\mathcal{X}$. A smooth manifold $\mathcal{M}$ without boundary is said to be a *submanifold* of another smooth manifold $\mathcal{N}$. If $\mathcal{M} \subset \mathcal{N}$ and the identity map of $\mathcal{M}$ into $\mathcal{N}$ is embedding. However, it is noteworthy that, while a submanifold $\mathcal{M}$ is just a subset of another manifold $\mathcal{N}$, $\mathcal{M}$ can have a dimension from a geometrical point of view, and the dimension of $\mathcal{M}$ may be lower than the dimension of $\mathcal{N}$. A *P-manifold* or *P-dimensional-manifold* $\mathcal{M}$ is defined as submanifold [132] of $\mathcal{N} \subset \mathbb{R}^D$, if the following conditions holds for all point $x \in \mathcal{M}$: there exist two open

sets $\mathcal{U}, \mathcal{V} \subset \mathcal{M}$ with $x \in \mathcal{U}$, and a diffeomorphism $\mathbf{h} : \mathcal{U} \to \mathcal{V}, y \mapsto x = \mathbf{h}(y)$ such that:

$$\mathbf{h}(\mathcal{U} \cap \mathcal{M}) = \mathcal{V} \cap (\mathbb{R}^P \times \{0\}) = \{x \in \mathcal{V} : x_{P+1} = ... = x_D = 0\}$$

This means that x can trivially be reduced to P-dimensional coordinates. If $\mathcal{N} = \mathbb{R}^D$ in the previous definition, then:

- a point $y \in \mathbb{R}^D$ is a manifold;
- a P-dimensional vector subspace (a P-dimensional hyperplane) is a P-manifold;
- the hollow D-dimensional hyperplane is (D-1)-manifold;
- any ope subset is a D-manifold.

It was showed in [149] that any P-manifold can be embedded in $\mathbb{R}^{2P+1}$, meaning that $2P + 1$ dimensions at most are necessary to embed a P-manifold. For example, an open line segment is an open 1-manifold that can already be embedded in $\mathbb{R}^1$. On the other hand, a circle is a compact 1-manifold that can be embedded in $\mathbb{R}^2$ but not in $\mathbb{R}^1$. A knotted circle, like a trefoil knot, reaches the bound of Whitney's theorem: it can be embedded only in $\mathbb{R}^D$, with $D \leq 2P + 1 = 3$. This mathematical framework has allowed the development of *Topological Data Analysis* (TDA), which is a recent field that emerged from various works in applied topology and computational geometry during the first decade of the century. TDA aims at providing well-founded mathematical, statistical and algorithmic methods to infer, analyze and exploit the complex topological and geometric structures underlying data that are often represented as point clouds in Euclidean or more general metric spaces.

1.2.3 Taxonomy of dimensional reduction

The choice of a strategy to apply dimensional reduction is not trivial. In fact, the huge development in the last decades of big data in different fields can be a limitation for most of them. Furthermore, the real-world data concerns datasets with continuous variables but also discrete or categorical variables. For this reason, there does not exist the best way to choose a priori what is the appropriate technique to implement a dimensional reduction. When working with the data, it is important to understand the goal of the research. Dimensional reduction is a powerful strategy to make more manageable huge datasets. With this strategy, we can focus on latent variables through a projection or select an appropriate subset to predict or explain a variable of interest. Figure 1.3 shows the taxonomy of some techniques for dimensionality reduction. We have grouped these methods based on linearity, non-linearity, feature selection, and graphical representation. Consequently, the choice of one of these methods will be dependent on the nature of the variables presented in the dataset and the purpose of the research. In the next sections, we present the different dimensionality reduction techniques and we try to identify the strengths aspects and the inherent weaknesses.

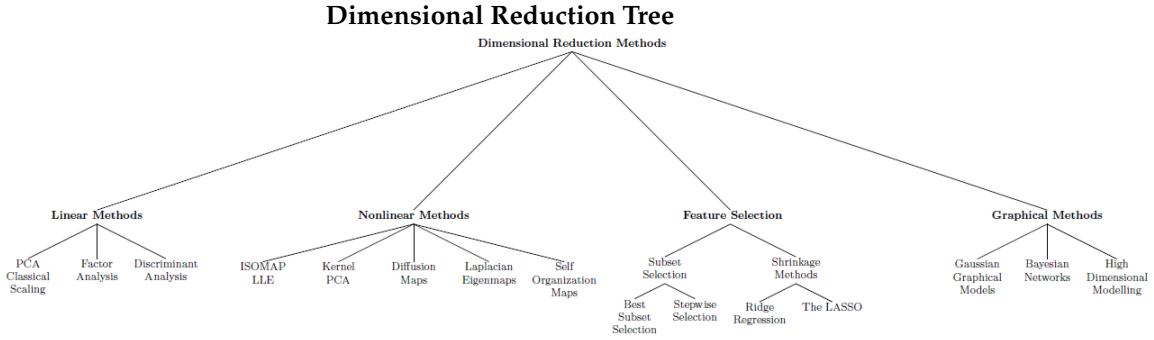


Figure 1.3: Taxonomy of dimensionality reduction methods

1.3 Linear Method

In this section, we introduce some linear methods. These methods compute a linear feature extraction or projection that expresses the new feature as a linear combination of the original variables. Perhaps the most popular techniques for dimensionality reduction belong to this group [80]. Indeed methods as Principal Component Analysis (PCA) or Factor Analysis (FA) are the oldest methods. The type of linear projection used in practice is influenced by the availability of category information about the patterns in the form of labels on the patterns. If no category information is available, the eigenvector projection is commonly used. In the classification problems, Linear Discriminant Analysis (LDA) is a linear mapping technique when category labels are available.

1.3.1 PCA and Classical Scaling

Principal Component Analysis (PCA) is the main linear method for dimensionality reduction. PCA shows a low-dimensional representation of the dataset $\mathbf{X}$ that describes the maximum variability of the data present in the dataset. The first principal component has the greatest variability. The second component has the maximum variability among all linear combinations that are orthogonal to the first. The third principal component is orthogonal respect to the first and the second, and so on for all component. The scope of PCA is to reduce the number of variables from p to k where $k < p$ or often $k \ll p$ and avoid loss of relevant information contained in the data. In mathematical terms, PCA attempts to find a linear mapping $\mathbf{M}$ that maximizes the cost function trace $(\mathbf{M}^T Cov(\mathbf{X})\mathbf{M})$, where $Cov(\mathbf{X})$ is the sample covariance matrix of the data $\mathbf{X}$ with respect to $\mathbf{M}$, under the constraint that *L2-norm* of each column $\mathbf{m}_j$ of $\mathbf{M}$ is 1, i.e. that $\|m\|^2 = 1$. If we introduce a Lagrange multiplier λ, this constraint will be enforced. Hence, an unconstrained maximization of $\mathbf{m}_j^T Cov(\mathbf{X})\mathbf{m}_j + \lambda(1 - \mathbf{m}_j^T \mathbf{m}_j)$ is performed. The stationary points of this quantity are to be found when $Cov(\mathbf{X})\mathbf{m}_j = \lambda \mathbf{m}_j$. $\mathbf{M}$ is formed by the k-principal eigenvectors of the sample covariance matrix of the zero mean data [119]. PCA solves the eigenproblem:

$$Cov(\mathbf{X})\mathbf{M} = \lambda \mathbf{M} \tag{1.7}$$

The eigenprobelm is solved for the the k principal eigenvalues λ. The low-dimensional data representations $\mathbf{y}_i$ of the datapoints $\mathbf{x}_i$ are computed by mapping them onto the linear basis $\mathbf{M}$, i.e. $\mathbf{Y} = \mathbf{XM}$. PCA is identical to the traditional method for multidimensional scaling called classical scaling [141]. The input into classical scaling is given by a pairwise Euclidean distance matrix $\mathbf{D}$ whose entries d_{ij} represent the Euclidean distance between the high-dimensional datapoints $\mathbf{x}_i$ and $\mathbf{x}_j$. Classical scaling finds the linear mapping $\mathbf{M}$ that minimizes the cost function:

$$\phi(\mathbf{Y}) = \sum_{ij}(d_{ij}^2 - \|\mathbf{y}_i - \mathbf{y}_j\|^2) \tag{1.8}$$

in which $\|\mathbf{y}_i - \mathbf{y}_j\|^2$ is the Euclidean distance between the low-dimensional datapoints $\mathbf{y}_i$ and $\mathbf{y}_j$; $\mathbf{y}_i$ is restricted to be $\mathbf{x}_i\mathbf{M}$ and $\|\mathbf{m}_j\|^2 = 1\ \forall_j$.
The minimum of this cost function is given by the eigendecomposition of the Gram matrix [141]: $\mathbf{K} = \mathbf{XX}^T$ of the high-dimensional data. The entries of the Gram matrix can be obtained by double-centering the pairwise squared Euclidean distance matrix, i.e. by computing:

$$k_{ij} = -\frac{1}{2}\left(d_{ij}^2 - \frac{1}{n}\sum_l d_{il}^2 - \frac{1}{n}\sum_l d_{jl}^2 + \frac{1}{n^2}\sum_{lm} d_{lm}^2\right) \tag{1.9}$$

The minimum cost of the function in Equation 1.8 can now be obtained by multiplying the principal eigenvectors of the double-centered squared Euclidean distance matrix with the square-root of their corresponding eigenvalues. The similarity of classical scaling to PCA is due to a relation between the eigenvectors of the covariance matrix and the Gram matrix of the high-dimensional data: it can be shown that the eigenvectors $\mathbf{u}_i$ and $\mathbf{v}_i$ of the matrices $\mathbf{XX}^T$ are related through $\sqrt{\lambda_i}\mathbf{v}_i = \mathbf{Xu}_i$. PCA may also be viewed upon as a latent variable model called probabilistic PCA. This model uses a Gaussain prior over the latent space, and a linear-Gaussian noise mode. This model leads to an EM-algorithm that may be computationally more efficient for very high-dimensional data. PCA and Classical Scaling have been successfully applied in a large number of domains that permit an useful interpretation of the results. Notwithstanding, PCA and Classical Scaling suffer from third main drawbacks. First, in PCA the size of the covariance matrix is proportional to the dimensionality of the data-points. As a result, the computation of the eigenvectors might be infeasible for very high-dimensional data. This drawbacks may be overcome by performing Classical Scaling instead of PCA, because the Classical Scaling with the number of datapoints instead of with the number of dimensions in the data. Second, the cost function in Equation 1.8 reveals that PCA and Classical scaling focus mainly on retaining large pairwise distances d_{ij}^2, instead of focusing on retaining the small pairwise distances, which is much more important. Finally, both method can not applied directly to datasets that contain discrete variables, in other words we can perform PCA method to different labels of a discrete variable.

1.3.2 Factory Analysis

Factor Analysis (FA) represents the variables $X_1, ..., X_p$ of the dataset as linear combination of random variables called *factors*. The propose of this method is to describe the covariance relationship among the variables in terms of these m underlying, but unobservable factors. If the original variables $X_1, ..., X_p$ are at least moderately correlated, the basic dimensionality of the data is less than p. The goal is to reduce the redundancy among variables by using a smaller number of factors ($m < p$) that explain the variables as possible. FA is a method to describe the interrelationships among a set of variables, seeking a simpler structure of the data, for this aspect is similar to PCA, but there are important difference between FA and PCA:

- PCAs are defined as linerar combinations of the variables. In FA, the original variables are expressed as linear combinations of the factors
- the goal of PCA is to explain a large part of the total variance of variables, which regards FA the intention is to explain the correlations among the variables

We assume a random sample $(x_{i1}, ..., x_{ip})$, from a population with mean vector $\{\mu_1, ..., \mu_p\}$ and covariance matrix Σ. The factor analysis model expresses each variable X_j as a linear combination of random variables $F_1, ..., F_m$ with coefficients $l_{j1}, ..., l_{jm}$ and an accompanying error term ϵ_j to account for that part of the variable that is unique:

$$X_j - \mu_j = l_{j1}F_1 + l_{j2}F_2 + ... + l_{jm}Fm + \epsilon_j \tag{1.10}$$

The $F_j's$ are random variables, called common factors, with $E(F_j) = 0, Var(F_j) = 1$ and $Cov(F_j, F_k) = 0$, l_{jk} are coefficients, called factor loading. The value of $|l_{jk}|$ indicates the importance of the factor F_k on variable X_j(we say that X_j loads highly on F_k). In other words, more is high the absolute value of l_{jk} more is explained the variable X_j by the factor F_k. While ϵ_j are random errors, called also specific factors, with $E(\epsilon_j) = 0$ but different variances $Var(\epsilon_j) = \psi_j$ (specific variance), and $Cov(\epsilon_j, \epsilon_k) = 0$, finally factors and errors are uncorrelated, $Cov(\epsilon_j, F_k) = 0$. Exist three main approaches to estimation of loadings and communalities [77] (facor extraction):

- principal component method;
- (iterated) principal factor method ;
- maximum likelihood method.

The main advantage of the Factor Analysis is that is possible to apply to discrete variables, if it is possible to order them, and continues variables but is not useful with huge dataset. Furthermore as PCA, FA can explain the latent variables but we can have the lose of information from original variables.

1.3.3 Linear Discriminant Analysis

Another strategy to obtain dimensional reduction is that to implement the classification when the data has associated class labels. Fisher's linear discriminant analysis is probably the most prominent example [117]. The purpose of LDA (Linear Discriminant Analysis) is to project the data in such a way that separation between classes is maximized. If we suppose that $f_k(x)$ is the class-conditional density of X in class $G = K$, and let π_k the prior probability of class w, where if we consider the total class we have $\sum_{K=1}^{K} \pi_w = 1$. Now we have all elements to apply the *Bayes theorem* when the class $C = k$:

$$Pr(C = k|X = x) = \frac{f_k(x)\pi_k}{\sum_{l=1}^{K} f_l(x)\pi_l} \tag{1.11}$$

If we model each class density as multivariate Gaussian:

$$f_k(x) = \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma_k^{-1}(x-\mu)} \tag{1.12}$$

and assume that classes have a common covariance matrix $\Sigma_w = \Sigma \forall_w$,we can apply linear discriminant analysis. LDA begins by partitioning the data covariance XX^T into covarinace contributed within each of the C classes (Σ_W) and covariance contributed between the classes (Σ_B) such that $XX^T = \Sigma_W + \Sigma_B$ for:

$$\Sigma_W = \sum_{i=1}^{n}(x_i - \mu_{ci})(x_i - \mu_{ci})^T \quad \Sigma_B = \sum_{i=1}^{n}(\mu_{ci} - \mu)(\mu_{ci} - \mu)^T \tag{1.13}$$

Where μ is the global data mean and μ_{ci} is the class mean associated with data point x_i. LDA seeks the projection that maximizes between-class variability $tr(M^T\Sigma_B M)$ while minimizing within-class variability $tr(M^T\Sigma_W M)$, leading to the optimization program:

$$\begin{aligned} &\text{maximize} && \frac{tr(M^T\Sigma_B M)}{tr(M^T\Sigma_W M)} \\ &\text{subject to} && M \in \mathcal{O}^{n\times p} \end{aligned} \tag{1.14}$$

This objective appears very mach like a generalized Rayleigh quotient, and is so for $r = 1$. In this special case, $M \in \mathcal{O}^{n\times 1}$ can be found as the top eigenvector of $\Sigma_W^{-1}\Sigma_B$, which can be seen by substituting $L = \Sigma_W^{1/2} M$ into Equation 1.14 above. This one-dimensional LDA projection is appropriate when there are $C = 2$ classes. A common misconception is that LDA for higher dimensional projections $p > 1$ can be solved with a greedy selection of the top p eigenvectors of $\Sigma_W^{-1}\Sigma_B$. However, this is certainly not the case, as the top p eigenvectors of $\Sigma_W^{-1}\Sigma_B$ will not in general be orthogonal. The eigenvectors solution solves the similar but not equivalent objective $tr((M^T\Sigma_W M)^{-1}(M^T\Sigma_B M))$ over $M \in \mathbb{R}^{n\times p}$ The main goal of LDA is basically separate example of classes linearly moving them to a different feature space, therefore if the dataset is linear separable, only applying LDA we can obtain a classifier, and it is possible to reduce a dimensionaly. However, if the dataset is not linear separable we can apply the algorithm and LDA

will try to organize the dataset in another space as the maximum linearly separability as possible, but it still be examples overlapping between classes because of non-linearly characteristic of data.

1.4 Nonlinear Method

In this section we review the principal Nonlinear Method (NLM) useful for dimensionality reduction. In contrast to linear method, nonlinear methods are often more powerful to find the connection between the latent variables and the observed ones. More specifically, NLM often comprise many parameters, whose identification requires large amounts of data. Many of these nonlinear dimensionality reduction methods are related to the linear method. Broadly speaking, these methods can be classified in two groups: those that provide a mapping, and those that infer to latent variables through non linear transformation of the dataset.

1.4.1 ISOMAP and LLE

Despite Classical Scaling can be used in many applications, also has certain weaknesses. In particular, it mainly aims to retain pairwise Euclidean distances, and does not take into account the distribution of the neighboring datapoints. In contrast, ISOMAP preserves pairwise geodesic distances between data point, this aspect is relevant when the the high-dimensional data lies on or near a curved manifold. Geodesic distance is the distance between two points measured over the manifold. In ISOMAP, the goedesic distances between the datapoints. $x_i (i = 1, 2, ..., n)$ are computed by constructing neighborhood graph G, in which every datapoint x_i is connected with its k nearest neighbors $x_{ij} (j = 1, 2, ..., k)$ in the dataset $\mathbf{X}$. In practice, a graph is created by either keeping only the connections between every point and its k nearest neighbors to produce a k-nearest neighbor graph (k-NNG), or simply by keeping all distances smaller than a value ϵ-neighborhood graph (ϵ-NNG). The geodesic distances between all datapoints in $\mathbf{X}$ are computed, thereby forming a pairwise geodesic distance matrix. The low-dimensional representations y_i in low-dimensional space $\mathbf{Y}$ are computed by applying classical scaling on the resulting pairwise geodesic distance matrix. Therefore, data agree with the model of ISOMAP if the pairwise geodesic distances computed between points of the P-manifold to be embedded can be mapped to pairwise Euclidean distances measured in P-dimensional Euclidean space. We assume that a developable manifold has parametric equations of the form $\mathbf{y} = \mathbf{m}(\mathbf{x})$ and that the isometry $\delta(\mathbf{y}(i), \mathbf{y}(j)) = ||\mathbf{x}(i) - \mathbf{x}(j)||_2$ holds. In $\mathbb{R}^p$ space, the geodesic distance between points $\mathbf{y}(i) = \mathbf{m}(\mathbf{x}(i))$ and $\mathbf{y}(j) = \mathbf{m}(\mathbf{x}(j))$ can be computed as follows:

$$\delta^2(\mathbf{y}(i), \mathbf{y}(j)) = \delta^2(\mathbf{m}(\mathbf{x}(i)), \mathbf{m}(\mathbf{x}(j)))$$

$$= \sum_{p=1}^{P} \delta^2(\mathbf{m}(\mathbf{x}(i)), \mathbf{m}([x_1(i), ..., x_p(j), ..., x_P(i)]^T))$$

Therefore, in a developable manifold, geodesic distances can be computed componentwise, either in the latent Euclidean space of the manifold space or in D-dimensional embedding space. Here we present the pseudo code of the ISOMAP algorithm:

- build a graph with either the K-rule or the ϵ-rule;
- weight the graph by labeling each edge with its Euclidean length;
- compute all pairwise graph distance with Dijkstra's algorithm, square them, and store them in matrix $\mathbf{D}$;
- convert the matrix of distances $\mathbf{D}$ into a Gram matrix $\mathbf{S}$ by double centering;
- once the Gram matrix is known, compute its spectral decomposition $\mathbf{S} = \mathbf{U}\Lambda\mathbf{U}^T$;
- a P-dimensional representation of $\mathbf{Y}$ is obtained by computing the product $\hat{\mathbf{X}} = \mathbf{I}_{P\times N}\Lambda^{1/2}\mathbf{U}^T$.

Unfortunately, also the the ISOMAP as the classical scaling, presents some weaknesses, in particular this algorithm is topological unstable [18]. In fact, ISOMAP can be constructed erroneous connections in the neighborhood graph in G. There are several approaches that overcome the problem of *short-circuiting*, one of them is to remove datapoints with large total flows in the shortest-path algorithm [28] or by removing nearest neighbors that violate local linearity of the neighborhood graph [124]. Furthermore there are another two weaknesses in ISOMAP, one due to *"holes"* in the manifold [96] and the second is that this algorithm can fail if the manifold is non-convex [139]. Despite these weaknesses, ISOMAP was successfully applied on task such as wood inspection, visualization of biomedical data and eventually head pose estimation.
Local Linear Embedding (LLE) is a technique that is similar to ISOMAP in that it constructs a graph representation of the data points. In contrast to ISOMAP, it attempts to preserve solely local properties of the data. As a result, LLE is less sensitive to short-circuiting than ISOMAP, because only a small number of local properties are affected if short-circuiting occurs. Furthermore, the preservation of local properties allows for successful embedding of non-convex manifolds. In LLE, the local properties of the data manifold are constructed by writing the high-dimensional data points as a linear combination of their nearest neighbors. In the low-dimensional representation of the data, LLE attempts to retain the reconstruction weights in the linear combinations as good as possible. LLE describes the local properties of the manifold around a data points $\mathbf{x}_i$ by writing the data point as linear combination $\mathbf{w}_i$ of its k nearest neighbors $\mathbf{x}_{ij}$. Hence, LLE fits a hyper-plane through the data point $\mathbf{x}_i$ and its nearest neighbors. LLE is a technique that constructs a weight matrix $\mathbf{W} \in \mathbb{R}^{n\times m}$ with elements w_{ij} so that:

$$\sum_{i=1}^{n} \left\| x_i - \sum_{j=1}^{n} w_{ij} x_j \right\|^2 \tag{1.15}$$

The Equation 1.15 is minimized under the constraint that $w_{ij} = 0$ if x_j does not belong to the neighborhood and the constraint $\sum_{j=1}^{n} w_{ij} = 1$. Finally the embedding is made in

such way that the following cost function is minimized for **Y** the low-dimensional data representation:

$$\phi(Y) = \sum_{i=1}^{n} \left\| y_i - \sum_{j=1}^{n} w_{ij} y_j \right\|^2 \quad \textbf{subject to} \quad ||\mathbf{y}^{(k)}||^2 = 1 \quad \textbf{for} \quad \forall k \tag{1.16}$$

$\phi(\mathbf{Y}) = (\mathbf{Y} - \mathbf{WY})^2 = \mathbf{Y}^T(\mathbf{I} - \mathbf{W})^T(\mathbf{I} - \mathbf{W})\mathbf{Y}$ is the function that has minimized [122]. Hence the eigenvectors of $(\mathbf{I} - \mathbf{W})^T(\mathbf{I} - \mathbf{W})$ corresponding to the smallest nonzero eigenvalues from the solution that minimizes $\phi(\mathbf{Y})$. Conceptually the method is similar to ISOMAP, but it is computationally much nicer because the weight matrix is sparse and there exist deficient solver. The popularity of LLE has led to development of linear variants of the algorithm, in fact it was applied with successful in different fields. However LLE was reported to fail in visualization [99] and it is performed worse than ISOMAP [75]. Finally, LLE tends to collapse large portions of the data very close together in the low-dimensional space, because the covariance constraint on the solution is too simple.

1.4.2 Kernel-PCA

Kernel PCA (kPCA) is an extension of linear PCA in high-dimensional space which is constructed using a kernel transformation. In contrast of PCA, kernel PCA computes the principal eigenvectors of the kernel matrix, rather than the covariance matrix. It is possible to reformulate PCA in kernel space through a kernel transformation. If the variables of dataset **X** are centered around 0, then the principal components can also be computed from the inner product matrix $K = X^T X$. Due to this way of calculating a PCA, we do not need to explicitly map all points into the high dimensional space and do the calculations there, it is enough to obtain the inner product matrix or kernel matrix $K \in \mathbb{R}^{n \times m}$ of mapped points [126]. An example of the entries in the kernel matrix is given by Gaussian kernel:

$$K = \phi(x_i)^T \phi(x_i) = \kappa(x_i, x_j) = exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \tag{1.17}$$

where σ is a length scale parameter accounting for the width of the kernel. The principal d eigenvectors $\mathbf{v}_i$ of the centered kernel matrix are computed. The eigenvectors of the covariance matrix $\mathbf{a}_i$ (in the feature space constructed by k) can now be computed, since they are related to the eigenvectors of the kernel matrix $\mathbf{v}_i$ through $\mathbf{a}_i = \frac{1}{\sqrt{\lambda_i}}\mathbf{v}_i$. In order to obtain the low-dimensional data representation, the data is projected onto the eigenvectors of the covariance matrix $\mathbf{a}_i$. The result of the projection (i.e. the low-dimensional data representation **Y**) is given by:

$$\mathbf{y}_i = \left\{ \sum_{j=1}^{n} a_1^{(j)} k(\mathbf{x}_j, \mathbf{x}_i), ..., \sum_{j=1}^{n} a_d^{(j)} k(\mathbf{x}_j, \mathbf{x}_i) \right\} \tag{1.18}$$

where $a_1^{(j)}$ indicates the jth value in the vector $\mathbf{a}_1$ and k is the kernel function that was also used in the computation of the kernel matrix. The kPCA method is very flexible and there exist many kernels for special purposes. The flexibility comes at the price that the method has to be finely tuned for the dataset because some parameter combinations are simply unsuitable for certain data. An important weakness of Kernel PCA is that the size of the kernel matrix is proportional to the square of the number of instances in the dataset. In other words the kPCA is not suitable for very large data sets, because memory scales with $\mathcal{O}(n^2)$ and computation time with $\mathcal{O}(n^3)$.

1.4.3 Diffusion Maps

Diffusion Maps are originated by the filed of dynamic systems, this method is based on defining of a Markov Random Walk on the graph of the data [108]. This measure is known as diffusion distance, the key idea behind the diffusion distance is that it is based on integrating over all paths through the graph. In the Diffusion Map first is constructed a weights graph. Through a Gaussian kernel function are computed the weights, this leads to a matrix $\mathbf{W}$ with entries $w_{ij} = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$, where σ indicates the variance of the Gaussian. Subsequently, the matrix $\mathbf{W}$ is normalized, in other words every rows add up to 1. In this way, a matrix P^1 is formed with entries: $p_{ij}^1 = \frac{w_{ij}}{\sum_k w_{ij}}$. Since Diffusion Maps originate from dynamical system theory, the matrix P^1 is a *Markov Matrix* that defines the forward transition probability matrix of a dynamical process. From the matrix P^1 we can observe the probability of transition from one to another datapoint in a single timestep. The forward probability matrix for t timesteps $P^{(t)}$ is thus given by $(P^{(1)})^t$. Using the random walk forward probabilities $p_{ij}^{(t)}$, the diffusion distance is defined by:

$$D^{(t)}(x_i, x_j) = \sqrt{\sum_k \frac{\left(p_{ik}^{(t)} - p_{jk}^{(t)}\right)^2}{\psi(x_k)^{(0)}}} \tag{1.19}$$

where $\psi(x_i)^{(0)} = \frac{m_i}{\sum_j m_j}$, and m_j is the degree of node x_i defined by $m_i = \sum_j p_{ij}$, this means that $\psi(x_i)^{(0)}$ gives more weight to parts of the graph with high density. In other words the Equation 1.19 shows that pairs of datapoints with a high forward transition probability have a small diffusion distance. Since the diffusion distance is based on integration over all paths through the graph. In the low-dimensional representation of the data $\mathbf{Y}$, diffusion maps attempt to retain the diffusion distances. Using spectral theory on the random walk it has been shown that the low-dimensional representation $\mathbf{Y}$ that retains the diffusion distances $D^{(t)}(x_i, x_j)$ as good as possible is formed by the d nontrivial principal eigenvectors of the eigenproblem:

$$P^{(t)}v = \lambda v \tag{1.20}$$

Because the graph is fully connected, the largest eigenvalue is trivial and the eigenvecotr v_1 is thus discarded. The low-dimensional representation $\mathbf{Y}$ is given by the next

d principal eigenvectors. In the low-dimensional representation, the eigenvectors are normalized by their corresponding eigenvalues. Hence, the low-dimensional data representation is given by

$$Y = \{\lambda_2 v_2, ..., \lambda_{d+1} v_{d+1}\} \tag{1.21}$$

Diffusion Map is based on the diffusion distance on integration over all paths through the graph, it is more robust to short-circuiting than the geodesic distance that is employed in ISOMAP. Diffusion Map have been applied to shape matching and gene expression analysis.

1.4.4 Laplacian Eigenmaps

Laplacian Eigenmaps is a method that find a low-dimensional data representation by preserving local properties of manifold. Indeed, the local properties with this method are based on the pairwise. Laplacian Eigenmaps computes the distances that minimized between a data point and its nearest k neighbors. Using spectral graph theory, the minimization of the cost function is defined as an eigenproblem. The Laplacian Eigenmaps first builds a neighborhood graph G in which every row x_i is connected to its k nearest neighbors. For all points x_i and x_j in graph G that are connected by an edge, the weight of the edge, as in the Diffusion Maps, is computed using the Gaussian Kernel function, leading to a sparse adjacency matrix $\mathbf{W}$. In the computation of the low-dimensional representations y_i, the cost function that is minimized is given by:

$$\phi(Y) = \sum_{ij} \|y_i - y_j\|^2 w_{ij} \tag{1.22}$$

in this function large weights w_{ij} correspond to small distances between the high-dimensional datapoints x_i and x_j. The Equation 1.22 can be solved formulating the minimization problem as an eigenproblem, indeed it can be shown that the following holds:

$$\phi(Y) = \sum_{ij} \|y_i - y_j\|^2 w_{ij} = 2Y^T LY \tag{1.23}$$

The graph Laplacian $\mathbf{L}$ is computed by $\mathbf{L} = \mathbf{M} - \mathbf{W}$ where $\mathbf{M}$ is the degree matrix of $\mathbf{W}$, it is a diagonal matrix, in which the entries are the rows sums of $\mathbf{W}$, i.e. $m_{ii} = \sum_j w_{ij}$. The low-dimensional data representation $\mathbf{Y}$ can thus be found by solving the generalized eigenvalue problem:

$$Lv = \lambda Mv \tag{1.24}$$

For the d smallest nonzero eigenvalues. The d eigenvectors v_i corresponding to the smallest nonzero eigenvalues form the low-dimensional data representation $\mathbf{Y}$. Laplacian Eigenmaps suffer from many of the same weaknesses as LLE, such as the presence of trivial solution that is prevented from being selected by a covariance constraint that can easily be cheated on. Despite these weaknesses, Laplacian Eigenmaps have been successfully applied to, e.g. face recognition [65] and the analysis of fMRI data [24].

In addition, variants of Laplaican Eigenmaps may be applied to supervised or semi-supervised learning problems [32].

1.4.5 Self Organization Maps

Self Organization Maps (SOM) is a neural network method to project data points from some input space to a position in a low-dimensional output space [20]. A SOM is organized with a two-dimensional rectangular gird (other choices, such as hexagonal grids, can also be used) of K neural prototypes $l_i \in \mathbb{R}^p$, which they correspond weight vectors w_r. SOM maps a data point x_i to a neural located at l_j in the map output space according to $x_i \mapsto l_j : l_j = \min_l \|x_i - w_j\|$. Then only the weights of the winning neuron for the present input are update. If this winning neuron has been chosen for representing an input instance for the first time its weights are set equal to the input instance. According to [85], the weights are upgraded, in which the neighborhood function is reduced to only the winning neuron:

$$w_j(t+1) = w_j(t) + \alpha(t)[x_i(t) - w_j(t)] \tag{1.25}$$

where $w_j(t)$ are the weights of the winning neurons at present iteration t, $x_i(t)$ is the present input instance and $\alpha(t)$ is the usual learning ratio that decreases over the time t through a Gaussian function [85]. The sequence $x_i(t)$ of feature vectors which constitute the original movement pattern i is transformed by the SOM into a sequence of excited neurons $l_j(t)$. Instead of considering a distance between sequences $x_i(t)$, $x_j(t)$ directly, in the p-dimensional input space, it can be operated in the only 2-dimensional output space, with all redundant, noisy extra dimensions suppressed. This leads to a distance matrix:

$$d(i,j) = \sqrt{\sum_t \left(l_i(t) - l_j(t)\right)^2}. \tag{1.26}$$

SOM is a iterative algorithm that selects the best node in the output map at each step, so that distances in the input space are preserved by their representatives in the output map. SOM provides a good approximation of the input space, furthermore is topologically order: in general, similar inputs are mapped to close neurons. SOM reflects the statistics of the input: regions of the input space where the input arrives with highest probability correspond to wider regions in SOM and are therefore represented with higher resolution.

1.5 Feature Selection

In this section, we introduce some methods developed by statistical learning. In particular, these methods try to combine the availability of the huge dataset with the statistical theory. The aim is to select a subset of variables to predict or explain the variables of interest. In other words, the feature selection produces a dimensional reduction through a selection of the features in relation to the variable of interest. This means that

if we change the variable of interest we can have different feature selection. Substantially, in this case, we see some ways in which the simple linear model can be improved, by replacing plain least-squares fitting with some alternative fitting procedures. The main advantage of these methods is that they can be applied with discrete and continuous data, without restriction. In the next sections, we define the variable of interest with Y.

1.5.1 Subset Selection

The term *subset selection* means a strategy for selecting subsets of predictors. There are different algorithms to perform this strategy and different measures to choose the best model between all possible models. The basic strategies used for selecting the attributes can be classified in that way [114]:

- *Best Subset Selection* [22]
- *Branch and Bound*[109]
- *Sequential Methods*: [63]
  - *Sequential Forward Search*
  - *Sequential Backward Search*
  - *Hybrid Approach*

All of these methods result in the creation of a set of models, each of which contains a subset of k predictors. In order to implement these methods, we consider a strategy to discern the best model $\mathcal{M}_k$ between all possible models $\mathcal{M}_0, ..., \mathcal{M}_p$. This strategy consist to evaluate the *mean square error* (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(v_i - \hat{f}(x_i))^2 = \frac{RSS}{n}$$

We fit a model to the training data using the least squares, we specifically estimate the regression coefficients such that the training *residual sum square* (RSS) is a small as possible. In particular, the training error will decrease as more variables are included in the model but the test error may not. However, there are different methods to avoid the problem for the model size. Indeed, these approaches can be used to select among a set of models with different numbers of variables. To evaluate how to chose the best model in terms of goodness of the forecast, we can consider the model selection measures C_p, *Akaike information criterion* (AIC) and *Bayesian information criterion* (BIC). For a fitted least squares model containing k predictors, the C_p estimate of the MSE is computed using the equation:

$$C_p = \frac{1}{n}(RSS + 2k\sigma^2)$$

where σ^2 is a estimation of the variance of the error $\epsilon_i = v_i - \hat{f}(x_i)$. The AIC criterion is defined for a large class of models fit by maximum likelihood. In the case of classical

model as $Y = \sum_{i=1}^{k} \beta_k \mathbf{X} + \epsilon$, with Gaussian error, maximum likelihood and least squares are the same thing. In this case AIC is given by:

$$AIC = \frac{1}{n\sigma^2}(RSS + 2k\sigma^2).$$

As we can see, the C_p and AIC are proportional to each other, for this reason one of them can be used to evaluate the goodness of the models. BIC is derived from a Bayesian point of view, but ends up looking similar to AIC as well. For the least squares model with k predictors, the BIC is up to irrelevant constants, given by:

$$BIC = \frac{1}{n}(RSS + \log(n)k\sigma^2).$$

Essential the C_p and AIC add a penalty of $2k\sigma^2$ to the training RSS in order to adjust for the fact the training error tends to underestimate the test error, while BIC adds a penalty of $k\sigma^2$. As we can see, for all of them the penalty increases as the number of the predictors in the models increases; this is intended to adjust for the corresponding decreases in training RSS. Small value of C_p, AIC and BIC indicates a model with a low test error. As consequence, these statistics tend to take on a small value for models with low test error, so when determining which of a set of models is best, we choose the model with lowest C_p or AIC or BIC value. Another strategy to choose the best model is to considering the goodness of the fit, in this case we use the *adjusted* R^2. It is another popular approach for selecting among a set of models that contain different numbers of the variables. The R^2 is defined as $1 - RSS/TSS$, where $TSS = \sum_{i=1}^{n}(v_i - \bar{v})$ is the *total sum of squares* for the response. Since RSS always decreases as more variables are added to the model, the R^2 always increase as more variables are added. For a least of squares model with k variables, the *adjusted* R^2 statistics is computed as:

$$\text{Adjusted} \quad R^2 = 1 - \frac{n-1}{n-k-1} \times \frac{RSS}{TSS}$$

The *adjusted* R^2 can be assumed value between 0 and 1. Unlike C_p, AIC and BIC for which small value indicates a model with a low test error, a large value of *adjusted* R^2 indicates a model with a small test error. Indeed, more the *adjusted* R^2 is close to 1, more this statistic tells us whether the k regressors are good to explaining the values of the interest variable [111].

Best Subset Selection

The *best subset selection* fits a separate least squares regression for each possible combination of the p predictors. That is, it fits all p models that contain exactly one predictor, all $\binom{p}{2}$=$p(p-1)/2$ models that contain exactly two predictors, and so forth. Then, the aim is to identify from among the 2^p possibilities the best model, in terms predict or explain the variable of interest. The *pseudo code* of the *best subset selection* algorithm is composed in the follow way:

- Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean foe each observation.
- For $k = 1, 2, ..., p$:
  - a) Fit all $\binom{p}{k}$ models that contain exactly k predictors
  - b) Pick the *best* among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* means the model that have the smallest RSS, or equivalent largest R^2
- Select a single best model from among $\mathcal{M}_0, ..., \mathcal{M}_p$ using cross-validated prediction error, C_p AIC, BIC, or *adjusted* R^2

The step 2 of Algorithm *Best Subset* identifies the best model (on the training data) for each subset size, in order to reduce the problem from of 2^p possible models to one of p+1 possible models.

Branch and Bound

The Branch and Bound method organizes the features in a tree, where the root pertains to choosing all the features. The children at the next level of the root consist of a combination of features by removing one feature. This strategy continues from each of these children, new nodes are formed where other features have been removed, and so on. In the leaf of the tree, we have a combination of features. Once a leaf node is reached, the criterion function's values go below bound, then that branch is not evaluated. When another leaf is reached if its criteria value is greater than bound then it is updated and that combination of features is stored as the best so far. However, the subset generated may not be optimal as some of the nodes are not expanded [114].

Sequential Methods

For computational reasons, *best subset selection* and *branch and bound* cannot be applied with very huge dataset. The larger the search space, the higher the change of finding models that look good on the training data, even though they might not have any predictive power on future data. Thus an enormous search space can lead to over-fitting and high variance of the coefficient estimates. For these reasons, the *Sequential Methods* can be considered a possible solution to these challenges, in fact, they are attractive alternatives to previous methods. In this section we present the *Sequential Methods*, where the features are sequentially added or removed at each step based on the criterion function. The criterion to evaluate the best model among the all possible models with different number of regressors k are the same present in the section 1.5.1

Sequential Forward Search is computationally more efficient than previous methods. In fact, it does not consider all possible 2^p models containing a subset of k predictors. The strategy of the dimensional reduction in *Sequential Forward Search* starts with an empty set and keeps on adding features. It adds one feature at a time until all of the predictors are in the model. The feature being added is the most significant or relevant

one. Once a feature gets added, then it cannot be removed. This procedure is reported in the following steps:

- Let $\mathcal{M}_0$, denote the null model, which contains no predictors.
- For $k = 0, ..., p-1$:
  - a) Consider all $p-k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
  - b) Choose the *best* among these $p-k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest R^2
- Select a single best model form among $\mathcal{M}_0, ..., \mathcal{M}_p$ using cross-validated prediction error,C_p, AIC, BIC, or *adjusted* R^2

Sequential Forward Search's computational advantage over best subset selection is clear. Though it can be tend to do well in practice, it is not guaranteed to find the best possible model out of all 2^p models containing subsets of the k predictors.

Sequential Backward Search provides an efficient alternative to best subset selection, like sequential forward search. It starts with all the p features inside in the model and in every iteration a feature is discarded. The feature being removed is the one which is most insignificant or irrelevant. Also once a feature is discarded, it cannot be added again. Details are given in Algorithm below:

- Let $\mathcal{M}_p$ denote the *full* model, which contains all p predictors
- For $k = p, p-1, ..., 1$:
  - a) Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$ for a total of $k-1$ models, and call it $\mathcal{M}_{k-1}$ predictors
  - b) Choose the *best* among these k models, and call it $\mathcal{M}_{k-1}$. Here *best* is defined as having smallest RSS or highest R^2
- Select a single best model form among $\mathcal{M}_0, ..., \mathcal{M}_p$ using cross-validated prediction error,C_p, AIC, BIC, or *adjusted* R^2

Like sequential forward selection, the sequential backward search examinations through only $1 + p(p+1)/2$ models, and so can be applied in settings where p is too large to apply best subset selection. Also like sequential forward search, sequential backward search is not guaranteed to yield the best model containing a subset of p features.

Hybrid Approach is the strategy that combine the *sequential algorithms*. It adds some features remove some of them in each iteration. Features being added are the most significant ones and the ones being removed are the least significant ones. This approach tries to solve the problem of the search for the best subset of predictors with p large.

1.5.2 Shrinkage Methods

The methods presented in the Subset Selection section, involve using least squares to fit a linear model that contains a subset of predictors. Here we present an alternative with an approach that contains all p predictors using a technique that *constrains* or *regularizes* the importance the predictors inside the model, or equivalently, that *shrinks* the coefficient estimates towards zero. The two most important techniques for *shrinkage methods* are:

- Ridge Regression
- Lasso

Ridge Regression

The Shrinkage Methods are an evolution of the *least square fitting*, where the procedure is to identify the coefficients of the linear regression β_i, $j = 0, .., k$ where k is the number of the predictors, that minimize:

$$RSS = \sum_{i=1}^{n} \left(v_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{i,j} \right)$$

The ridge regression is very similar to least squares, except that the coefficients are estimated by minimizing a slightly different quantity. The ridge regression coefficient estimates $\hat{\beta^R}$ are the values that minimize:

$$\sum_{i=1}^{n} \left(v_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{i,j} \right) + \lambda \sum_{j=1}^{P} \beta_j^2 = RSS + \lambda \sum_{j=1}^{P} \beta_j^2 \tag{1.27}$$

where $\lambda \leq 0$ is called *tuning parameter*, to be determined separately. A method to determine the value of λ, is to choose a gird of λ values, and compute the cross-validation error for each value of λ, and chose the λ associate with the smaller cross validation error. The Equation 1.27 can be studied in two component. As with least squares, ridge regression seeks coefficient estimates that fit the data well, by making the *residual sum squares* small. In the second term of the equation we the *"key"* of the dimensional reduction. In fact $\lambda \sum_j \beta_j^2$, called a *shrinkage penalty* is small when $\beta_1, ..., \beta_k$ are close to zero, and so it has the effect of *shrinking* the estimates of β_j towards zero. The tuning parameter λ serves to control the relative impact of these two terms on the regression coefficient estimates. When $\lambda = 0$, the penalty terms has no effect, and ridge regression will produce the least squares estimates. However, as $\lambda \to \infty$, the impact of the shrinkage penalty grows, and the ridge regression coefficient estimates will approach zero. The choice of λ is critical, but once passed this step this method is very useful. In fact, the ridge regression can be applied with a large dataset with continuous and discrete variables. Nevertheless, the presence of *spurious relationship* in the dataset should be taken into consideration. The main disadvantage of this method is that we can not apply the econometrics theory to interpret the coefficients.

Lasso

Ridge regression does have an obvious disadvantage. Unlike best subset and sequential methods which will generally select models that involve just a subset of the variables, ridge regression will include all p predictors in the final model. The penalty $\lambda \sum_j^p \beta^2$ will shrink all of coefficients towards zero, but it will not set any of them exactly to zero (unless $\lambda = \infty$). This may not be a problem for prediction accuracy, but it can create a challenge in model interpretation in settings in which the number of p is quite large. Ridge regression will always generate a model involving all predictors inside in the dataset. Increasing the value of λ will tend to reduce the magnitudes of the coefficients, but will not result in exclusion of any of the variables. The *Lasso* is a relative recent alternative to ridge regression that overcomes this disadvantage. The lasso coefficients, $\hat{\beta}_\lambda^L$, minimize the quantity:

$$\sum_{i=1}^{n}\left(v_i - \beta_0 - \sum_{j=1}^{k}\beta_j x_{i,j}\right) + \lambda\sum_{j=1}^{P}|\beta_j| = RSS + \lambda\sum_{j=1}^{P}|\beta_j| \tag{1.28}$$

Where the strategy to choose the best λ is the same of ridge regression. Comparing the two Shrinkage Methods, we note that they have a term in common. The main difference is that β_j^2 term in the ridge regression penalty has been replaced by $|\beta_j|$ in the lasso penalty. From a statistical point of view, the lasso uses an l_1 penalty instead of an l_2 penalty. The similarity between the lasso and ridge regression is not only in the formula of optimization but also that the lasso as the ridge regression, shrinks the coefficient estimates towards zero. However, in the case of the lasso, the l_1 penalty has the effect of forcing some of the coefficients estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large. In other words to produce a dimensional reduction through the feature selection. As a result, models generated from the lasso are generally much easier to interpret than those produced by the ridge regression [50].

1.6 Graphical Models

In this section we introduce the Graphical Model as technique to mapping the relationship between the variables and as representations of hierarchical Bayesian models. A Graphical Model defines a family of probability distribution through a graph. The nodes in the graph are identified with the random variables, and joint probability distribution are defined by taking products over functions defined on connected subset of nodes [78]. Essentially, they are an elegant framework which combines uncertainty (probability) and logical structure (independence constraints) to compactly represent complex, realworld phenomena [25]. The two most important forms of Graphical Model are represented by *directed graph* and *undirected graph*. In the first case we have a directed graph $G(V, E)$, where V are the nodes and E are the edges of the graph. Let $\{X_v : v \in V\}$ be a collection of random variables indexed by the nodes of the graph. To each node $v \in V$, let π_v denote the subset of indices of its parents. With X_{π_v} we indicate

the vector of random variables indexed by the parents of v. According to [78], given a collection of kernels $\{k(x_v|x_{\pi_v}) : v \in V\}$ that the sum (in discrete case) or integrate (in continuous case) to 1 (with respect to x_v), we define a joint probability distribution (a probability mass function or probability density as appropriate) as:

$$p(x_v) = \prod_{v \in V} k(x_v|x_{\pi_v}) \tag{1.29}$$

We can write $k(x_v|x_{\pi_v}) = p(x_v|x_{\pi_v})$ because it is possible to verify that this joint probability distribution has $\{k(x_v|x_{\pi_v})\}$ as its conditional[78]. Given an undirected graph $G(V, E)$, and as in the directed case we have that $\{X_v : v \in V\}$ be a collection of random variables indexed by the nodes of the graph and let $\mathcal{C}$ denote a collection of *cliques* of the graph (that we can define as path where the star node is the end node). Associated with each clique $C \in \mathcal{C}$, let $\psi_C(x_C)$ denote a nonnegative *potential function*. Thus we can define the joint probability $p(x_v)$ by taking the product over these potential functions and normalizing

$$p(x_v) = \frac{1}{\phi} \prod_{C \in \mathcal{C}} \psi_C(x_C) \tag{1.30}$$

where ϕ is a normalization factor obtained by integrating or summing the product with respect to x_v. Directed graphical models are familiar as representations of hierarchical Bayesian models, while the undirected graphs are often used in problems in areas as spatial statistics, statistical natural language processing and communication networks. In the next section we present in details the different strategy to build the Graphical Models.

1.6.1 Gaussian Graphical Models

Let $X = (X^{(1)}, ..., X^{(p)})$ be a p-dimensional random vector with a Multi-Gaussian distribution $\mathcal{N}_p(\mu, \Sigma)$ with unknown mean μ and nonsingular covariance matrix Σ. The key quantity in Gaussian Graphical Models is the inverse of covariance matrix $K = \Sigma^{-1}$ called as the *concentration matrix*:

$$K = \begin{pmatrix} k_{11} & k_{12} & \cdots & k_{1p} \\ k_{21} & k_{22} & \cdots & k_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ k_{p1} & k_{p2} & \cdots & k_{pp} \end{pmatrix} \tag{1.31}$$

In particular the partial correlation between x_u and x_v given all other variables, can be expressed by the value of the concentration matrix K as:

$$\rho_{uv|V\setminus\{u,v\}} = -k_{uv}\sqrt{k_{uu}k_{vv}} \tag{1.32}$$

Thus, when x_u and x_v are conditional independent given all other variables, we have from the concentration matrix that $k_{uv} = 0$. This is the covariance selection problem [38]

or the model selection-problem in the Gaussian concentration graph model [33]. From a construction point of view, a Gaussian concentration graph model for the Gaussian random vector $\mathbf{X}$ is represented by an undirected graph $G = (V, E)$ where V represents the set of the vertices, direct correspondence with the p variables, while the edges $E = (e_{u,v})_{1 \leqslant u < v \leqslant p}$ describe the conditional independence relationships among the variables: $X_1, ..., X_p$. If the edge between X_u and X_v is absent means that X_u and X_v are conditional independent given all other variables present in the dataset [152]. This graph is called the *dependence graph* [67] and it is often convenient to represent the model by the cliques $\mathcal{C} = \{C_1, ..., C_Q\}$ of the dependence graph. Recall that a probability distribution factorizes according to an undirected graph $G = (V, E)$ is given by the Equation 1.30. As we have seen above $K = \Sigma^{-}1$ and now we introduce the following equivalence $h = K\mu$. The multivariate Gaussian density is given by the Equation:

$$\begin{aligned} f(x) &= (2\pi)^{-d/2} det(K)^{\frac{1}{2}} exp\left\{-\frac{1}{2}(x-\mu)^T K(x-\mu)\right\} \\ &= (2\pi)^{-d/2} det(K)^{\frac{1}{2}} exp\left(-\frac{1}{2}\mu^T K\mu + h^T x - \frac{1}{2}x^T Kx\right) \end{aligned} \tag{1.33}$$

Letting $a = -\frac{d}{2}\log(2\pi) + \frac{1}{2}\log\det(K) - \frac{1}{2}\mu^T K\mu$ we can write the Equation 1.33 as:

$$f(x) = exp\left(a + h^T x - \frac{1}{2}x^T Kx\right) = exp\left(a + \sum_u h_u x_u - \frac{1}{2}\sum_u\sum_v k_{uv} x_u x_v\right) \tag{1.34}$$

According to the [32], if the vertices A and B are separated by a set C in the dependence graph we have $k_{uv} = 0$ for $u \in A$ and $v \in B$. By appropriately collecting terms we can write $f(x) = g(x_A, x_C)h(x_B, x_C)$. The factorization criterion applied to the equation 1.34 yields that for Gaussian graphical models the *global Markov property* holds: $A \perp B \mid C$. The maximum likelihood estimation of (μ, Σ) for a sample $x_1, ..., x_n$ of n observations from a multivariate Gaussian distribution, is $(\bar{x}, \bar{S})$ where: $\bar{S} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T$ denote the empirical covariance matrix. The log-likelihood function based on the sample is:

$$logL(K, \mu) = \frac{n}{2}\log\det(K) - \frac{n}{2}tr(KS) - \frac{n}{2}(\bar{x} - \mu)^T K(\bar{x} - \mu) \tag{1.35}$$

If we fixed K, $\hat{\mu} = \bar{x}$ the equation 1.35 is clearly maximized. The profile likelihood for K thus becomes

$$\log L(K, \hat{\mu}) = \frac{n}{2}\log\det(K) - \frac{n}{2}tr(KS) \tag{1.36}$$

Since $tr(KS) = \sum_u \sum_v s_{uv} k_{uv}$ it follow that only elements s_{uv} for which the corresponding elements k_{uv} of K are non-zero will contribute to the likelihood. There are three main methods to perform a Gaussian Graphical Models:

- Stepwise methods
- Convex optimization
- Thresholding

In stepwise methods performs model selection based on a variety of criteria. AIC or BIC criterion which minimizes the negative of a penalized likelihood is given by $-2\log L + Kdim(\mathcal{M})$. Where $dim(\mathcal{M})$ is the number of independent parameters in model $\mathcal{M}$ and $k = 2$ for AIC model otherwise in BIC model $k = \log n$. Figure 1.4 shows the Gaussian Graphical Models for *carcass* dataset, found by BIC stepwise method. The dataset is available on R with the library(gRim) and it is composed by 344 observation and 7 variables.

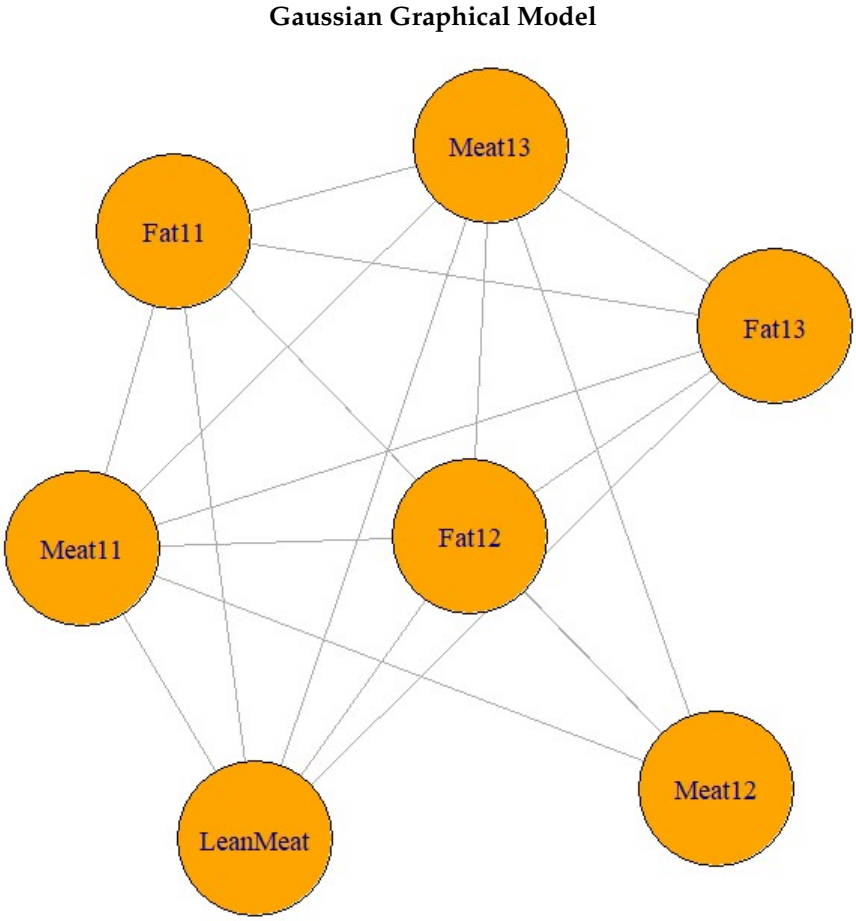


Figure 1.4: Model for carcass data found by stepwise selection using the BIC criterion

One way to avoid a stepwise search is to use the glasso algorithm. This technique finds a Undirected Gaussian Graphical Model (UGGM) that maximizes a *log*-likelihood for K which is penalized by the L_1-norm $|K|$. From equation 1.36 adding L_1-penalize the *log*-likelihood is equivalent to:

$$L_{pen}(K, \hat{\mu}) = \log\det(K) - tr(KS) - \rho|K| \tag{1.37}$$

where ρ is a non-negative penalty parameter. The L_1-norm $|K|$ is the sum of the absolute values of the elements of concentration matrix K. This sum is largely a proxy for the number of non-zero elements of K and as this penalized *log*-likelihood is convex in K, it can be optimized by convex programming methods. The smaller the value of ρ, the denser the graph that results. No penalization occurs for values of ρ close to zero. The simple and apparently naive method for selecting UGGM is to set a specific threshold for the partial correlations, so edges are removed for all partial correlations less than a given value. It is showed that all of these methods converged with the same result [32]. One of the advantage of UGGM is that can be applied when we have dataset with a large number variables and we want to do dimensional reduction through feature selection, furthermore we can visualization the relationship between the variables directly from the graph, and we can control which are many import with the penalty parameter. However It is important to remember that UGGM can be applied only with continuous data and the choice of the threshold can be conditioned the variables select in the model. In the past the main application of the UGGM regards in particular biological science and medical field [24], today the UGGM is an approach used in varied contexts[5].

1.6.2 Bayesian Networks

A Bayesian Networks belong of graphical models approach, that allow hierarchical representation of the probabilistic dependence between a given set of random variables $\mathbf{X} = \{X_1, ..., X_p\}$ through a *directed acyclic graph* (DAG) $G = (\mathbf{V}, A)$, where each node $v_i \in \mathbf{V}$ corresponds to a random variable $\{X_1, ..., X_p\}$. A *directed acyclic graph* is a graph with directed edges in which there are no cycles. Formally, a directed graph is a pair $(V, A \subseteq V \times V)$ consisting of a set nodes V and a binary relation A on it that specifies a directed edge from a node n to another one m whenever $(n, m) \in A$. The acyclicity condition of a dag (V, A) is ensure by requiring that transitive closure $A^+$of the relation A is irreflexive;*i.e.* $(n, n) \notin A^+$ for all $n \in V$ [46]. Figure 1.4 shows an example of a Bayesian Networks. This Network is a representation of the chest clinic proposed by [93] and we can note the relationships between the variables expressed in a hierarchical way. In the recent years Bayesian Networks have been used in many fields, in particular On-line Analytical Processing (OLAP) performance enhancement [102], medical service [3], gene expression analysis [52] and cancer prognosis and epidemiology [68]. The high dimensionality of datasets available in many field have led to the development of several learning algorithms focused on reducing computational complexity (in other words to do a feature selection) while still learning the correct network [130]. The DAG defines a factorization of the joint probability distribution of the joint probability distribution of $\mathbf{V} = \{X_1, ..., X_p\}$ often called global probability distribution, into a set of local probability distributions, one for each variable. The form of the factorization is given by the Markov property of Bayesian networks [86], which states tat evry random variable X_i directly depends only on its parents Π_{X_i}. The two most important aspects of a Bayesian network, are the conditional independence between the variables and the

graphical separation that corresponding of the nodes in the graph. Therefore model selection algorithms first try to learn the graphical structure of the Bayesian network and then estimate the parameters of the local distribution functions conditional on the learned structure. This two-step approach has the advantage that it considers one local distribution function at a time, and it does not require to model the global distribution function explicitly. Another advantage is that learning algorithms are able to scale to fit high-dimensional models without incurring in the so-called curse of dimensionality. Bayesian network structure learning algorithms can be grouped in two categories:

- *constraint-based algorithms*: these algorithms learn the network structure by analyzing the probabilistic relations entailed by the Markov property of Bayesian networks with conditional independence tests and then constructing a graph which satisfies the corresponding d-separation statements. The resulting models are often interpreted as causal models even when learned from observational data [115] [115]
- *score-based algorithms*: these algorithms assign a score to each candidate Bayesian network and try to maximize it with some heuristic search algorithm. Greedy search algorithms are a common choice, but almost any kind of search procedure can be used.

Constraint-based algorithms are all based on the Inductive Causation (IC) algorithm by [56], which provides a theoretical framework for learning the structure casual models. It can summarized in three steps:

- 1. first the *skeleton* of the network (the undirected graph underlying the network structure) is learned. Since an exhaustive search is computationally unfeasible for all but the most simple datasets, all learning algorithms use some kind of optimization such as restricting the search to the *Markov blanket* of each node v (defined as the minimal set that d-separates$v$ from the remaining variables, it is derived as the set of neighbours to v in moral graph of $\mathcal{G}$).
- 2. set all direction of the arcs that are part of a *v-structure* (a triplet of nodes incident on converging connection $X_j \rightarrow X_i \leftarrow X_k$)
- 3. set the directions of the other arcs as needed to satisfy the acyclicity constraint.

In contrast, *score-based algorithms* are simply applications of various general purpose heuristic search algorithms, such as *hill-climbing* [143], *tabu search* [54], and *simulated annealing* [34]. The networks with the same probability distribution are assigned the same score. To perform the conditional independence between the variables, there are several conditional independence test from information theory and classical statistics. Conditional independence test for discrete data are functions of the conditional probability tables implied by the graphical structure of the network through the observed frequencies $\{n_{ijk}, i = 1, ..., R, j = 1, ..., C, k = 1, ..., L\}$ for generic random variables X_a and X_b and all the configurations of the conditioning variables $\mathbf{X}$:

- *mutual information*: an information-theoretic distance measure [89] defined as

$$MI(X_a, X_b|\mathbf{X}) = \sum_{i=1}^{R}\sum_{j=1}^{C}\sum_{k=1}^{L}\frac{n_{ijk}}{n}\log\frac{n_{ijk}n_{++k}}{n_{i+k}n_{+jk}} \tag{1.38}$$

  It is proportional to the log-likelihood ration test G^2 (the differ by a $2n$ factor, where n is the sample size) and it related to the deviance of the tested models.

- *Person's* X^2: that is the classical Person's X^2 test contingency tables,

$$X^2(X_a, X_b|\mathbf{X}) = \sum_{i=1}^{R}\sum_{j=1}^{C}\sum_{k=1}^{L}\frac{(n_{ijk}-m_{ijk})^2}{m_{ijk}} \quad m_{ijk} = \frac{n_{i+k}n_{+jk}}{n_{++k}} \tag{1.39}$$

- *Akaike Information Criterion*: an experimental AIC-based independence test computed comparing the mutual information and the expected information gain. It rejects the null hypothesis if

$$MI(X_a, X_b|\mathbf{X}) \geq \frac{(R-1)(C-1)L}{n} \tag{1.40}$$

  which corresponds to an increase in the AIC score of the network.

In the continuous case conditional independence tests are functions of the partial correlation coefficients $\rho_{X_aX_b|\mathbf{X}}$ of X_a and X_b given $\mathbf{X}$:

- *linear correlation*: the linear correlation coefficient $\rho_{X_aX_b|\mathbf{X}}$
- *Fisher's Z*: a transformation of the linear correlation, it is defined as:

$$Z(X_a, X_b|\mathbf{X}) = \frac{1}{2}\sqrt{n-|\mathbf{X}|-3}\log\frac{1+\rho_{X_aX_b|\mathbf{X}}}{1-\rho_{X_aX_b|\mathbf{X}}} \tag{1.41}$$

- *mutual information*: an information-theoretic, distance measure [89] defined as

$$MI_g(X_a, X_b|\mathbf{X}) = -\frac{1}{2}\log(1-\rho^2_{X_aX_b|\mathbf{X}}) \tag{1.42}$$

  It has the same relationship with log-likelihood ratio as the corresponding test defined in the discrete case.

Several score function are used to compare different way to represent a dataset with a Bayesian network, here present the Bayesian information criterion, a penalized likelihood score defined as:

$$BIC = \sum_{i=1}^{n}\log P_{X_i}(X_i|\Pi_{X_i}) - \frac{d}{2}\log n, \tag{1.43}$$

for the discrete case, and

$$BIC = \sum_{i=1}^{n} \log f_{X_i}(X_i|\Pi_{X_i}) - \frac{d}{2}\log n \tag{1.44}$$

for continuous case. Where d is the number of the parameters of the global distribution. It is numerically equivalent to the information-theoretic *minimum description length* [120] These score functions are said to be *score equivalent*, since they assign the same score to networks belonging to the same equivalence class. They are also *decomposable* into the components associated with learning the structure of the network (the only parts of the score that need to be computed are those that differ between the networks being compared).

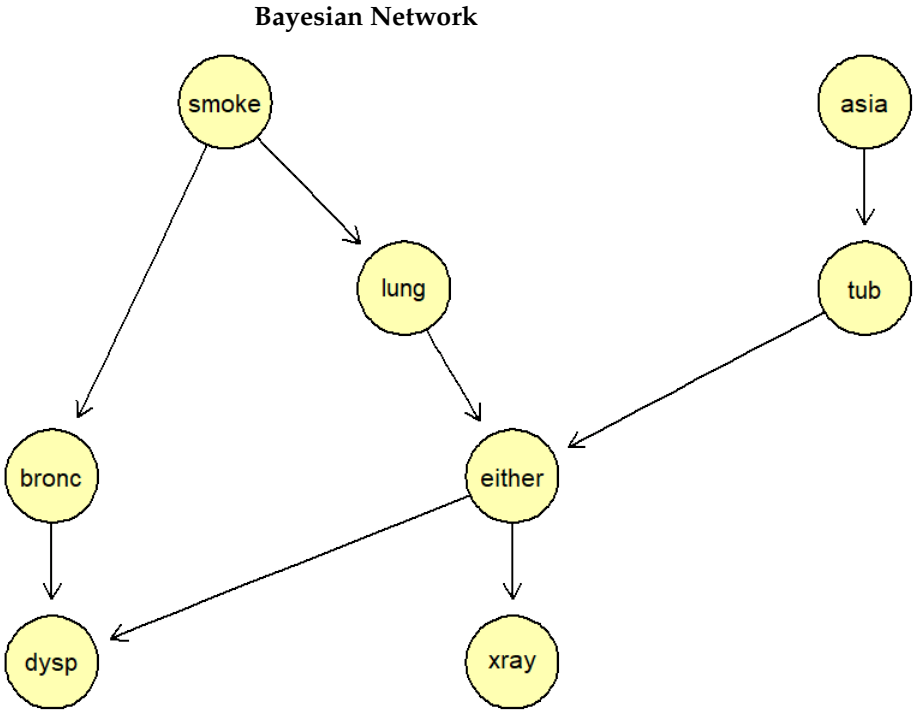


Figure 1.5: The directed acyclic graph corresponding to the clinic example from [93]

The Bayesian networks are very useful to understand the relationship between the variables and provide an intuitive and comprehensive framework to model high dimensional data. However, it is often the case that the available data have a more complex mean structure plus additional components of variance, which must then be accounted for in the estimation of a Bayesian network. The main limitation of the Bayesian network is that does not works well when our dataset is composed by mixed data (continuous and discrete random variables)

1.6.3 High Dimensional Modeling

In this section we present the high-dimensional graphical models, where the expression *"high-dimensional"* means of models with hundreds to tens of thousands of variables. Often this data has become of central importance in molecular biology, but with the availability of the Big Data can found this kind of data from other fields. The techniques seen until now for build a Graphical Models, work well in low dimensional applications (order of hundreds of variables), but them may be quite infeasible for high dimensional ones. Furthermore, the Gaussian Graphical Models, as suggests the name, work only with continuous variables and the Bayesian Networks can be infeasible with mixed data. In contrast, the models that will be present in this sections work with mixed data and we can do feature selection with *high-dimensional* dataset. As we seen in previous sections, *High Dimensional Modeling* represents a dataset through a graph that is a couple $G = (V, E)$, where V is the a finite set of nodes (given by the random variables present in our dataset) and $E \subset V \times V$ is a subset of ordered couples of V. Two vertices are connected by an edge (line) when the corresponding variables are conditionally dependent given the other variables represented in the graph. Suppose that the dataset is composed by d discrete and q continuous random variables, **(W,Z)**, where $\mathbf{W} = (W_1, ..., W_d)$ and $\mathbf{Z} = (Z_1, .., Z_q)$, thus i-observation can be written as a couple $(\mathbf{w}, \mathbf{z})$. Here, $\mathbf{w}$ is a d-tuple containing the values of the discrete variables, and $\mathbf{z}$ is a real vector of length q. We will denote with $P(\mathbf{x})$ a joint probability distribution for the random variables (**W,Z**). Let the corresponding set of nodes be V, where we can write the subset of d discrete nodes as Δ and Γ as the subset of q continuous node. Now the main task is to find an approximation of the joint probability distribution $P(\mathbf{x})$ to build a conditional graph from the data. A product approximation of $P(\mathbf{x})$ is defined to be a product of several of its component distribution of lower order ($P_a(\mathbf{x})$). We consider the class of second-order distribution approximation, i.e.:

$$P_a(\mathbf{x}) = \prod_{i=1}^{p} P(x_i, x_{j(i)}) \quad 0 \leqslant j(i) \leqslant p \tag{1.45}$$

where $(j_1, ..., j_p)$ is an unknown permutation of integers $(1, 2, ..., p)$, where $p = d + q$ [1]. [29] proved that for discrete random variables a goodness approximation between $P(\mathbf{w})$ and $P_a(\mathbf{w})$ is given by the minimization of the closeness measure computing the *mutual information* I :

$$I(P, P_a) = \sum_{w} P(\mathbf{w}) \log \frac{P(\mathbf{w})}{P_a(\mathbf{w})} \tag{1.46}$$

where $\sum_w P(\mathbf{w})$ is the sum over all levels of the discrete variables, this is equivalent to maximizing the total branch weight $\sum_{i=1}^{p} I(w_i, w_{j(i)})$, where

$$I(w_i, w_{j(i)}) = \sum_{w_i, w_{j(i)}} P(w_i, w_{j(i)}) \log \left(\frac{P(w_i, w_{j(i)})}{P(w_i) P(w_{j(i)})} \right) \tag{1.47}$$

Calculating the total branch weight for each of the p^{p-2} trees would be computationally too expensive even for moderate p. So we adopt a radical way of dealing with high-dimensional sparse data and restrict attention to forest and tree graph. A forest is is a special case of undirected graph, in particular is a graph with no cycles and it may be composed of several connected components called trees, i.e. a tree is a connected acyclic graph [106]. To build the graph is used the Kruskal's algorithm [88], in order to solve the problem of finding dependence tree or forest of maximum weight. This algorithm starts from a square weighted matrix p by p, where a weight for a pair of variables (W_i, W_j) is computed by the *mutual information* $I(W_i, W_j)$. From this point of view, the problem to found the best tree dependence from the data is reduced to calculating $p(p-1)/2$ weights, because the *mutual information* is symmetric, i.e. $I(W_i, W_j) = I(W_j, W_i)$ and there are not self loop. Consider, now a real application where probability distributions are not given explicitly. Let $\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_N$ be N independent samples of a finite discrete variable $\mathbf{x}$. Then, the mutual information can be estimated as follow:

$$\hat{I}(w_i, w_j) = \sum_{u,v} f_{u,v}(i,j) \log \frac{f_{u,v}(i,j)}{f_u(i) f_v(j)} \tag{1.48}$$

where $f_{u,v}(i,j) = \frac{n_{uv}(i,j)}{\sum_{u,v} n_{u,v}(i,j)}$, and $n_{u,v}(i,j)$ is the number of samples such that their ith and jth components assume the values of u and v, respectively. It can be shown that with this estimator we also maximize the likelihood for a dependence tree. This procedure it was extended to data with both discrete and continuous random variables [42]. The assumption is that random variables $\mathbf{X}$ are conditionally Gaussian distributed, i.e. the distribution of $\mathbf{Z}$ given $W = w$ is multivariate normal $N(\mu_i, \Sigma_i)$, so that both the conditional mean and covariance may depend on i. We discern two possible case, the homogenous and heterogeneous case. In the first case Σ depends on i and in the second case Σ does not depend on i. To apply the algorithm we need to derive the *mutual information* between a discrete variable W_u and a continuous variable Z_v, we introduce the sample cell counts, means, and variances as $\{n_i, \bar{z}_v, s_i^{(v)}\}_{i=1,...,|W_u|}$. In homogenous case an estimator of the *mutual information* is given by:

$$\hat{I}(w_u, z_v) = \frac{N}{2} \log\left(\frac{s_0}{s}\right), \tag{1.49}$$

where $s_0 = \sum_{k=1}^{N} (z_v^{(k)} - \bar{z})^2 / N$ and $s = \sum_{i=1}^{W_u} n_i s_i / N$. In this case we have $k_{w_u z_v} = |w_u| - 1$ degree of freedom. Otherwise, in heterogeneous case an estimator of the *mutual information* is given by:

$$\hat{I}(w_u, z_v) = \frac{N}{2} \log(s_0) - \frac{1}{2} \sum_{i=1,...,|W_s|} n_i \log(s_i) \tag{1.50}$$

in this case we have $k_{w_u z_v} = 2(|W_u| - 1)$ degree of freedom. Note that the algorithm will always stop when it has added the maximum number of edges, i.e. $p - 1$ for an undirected tree or forest. A disadvantage with selecting a tree based on maximum likelihood is that it will always include the maximum number of edges, irrespective of

whether the data support this or not. It is desirable to take account of number parameters in some fashion. In machine learning approach, it is prevalent the idea to penalize the likelihood using the minimum description length principle [121], whereas in the statistical literature the use of information criteria is well-established, particularly AIC (*Akaike information criterion* [6]) and BIC (*Bayesian information criterion* [127]). For this reason is used the Kruskal's algorithm in order to penalize *mutual information* quantities $\hat{I}^{AIC}_{u,v} = \hat{I}_{u,v} - 2k_{x_u,y_v}$ or $\hat{I}^{BIC}_{u,v} = \hat{I}_{u,v} - \log(n)k_{x_u,y_v}$, where k_{w_u,z_v} are the degrees of freedom, to avoid inclusion of links not supported by the data. The class of tree graphical models can be too restrictive for real data problem. However, we can start from the best spanning tree and determine the best strongly decomposable graphical model that is a key property which regards the feature selection. The definition of strongly decomposable graphical model is a graphical model whose graph neither contains cycles of length more than tree nor forbidden path [92]. A path exists between nodes A and B if one can reach A from B in a finite number of steps.

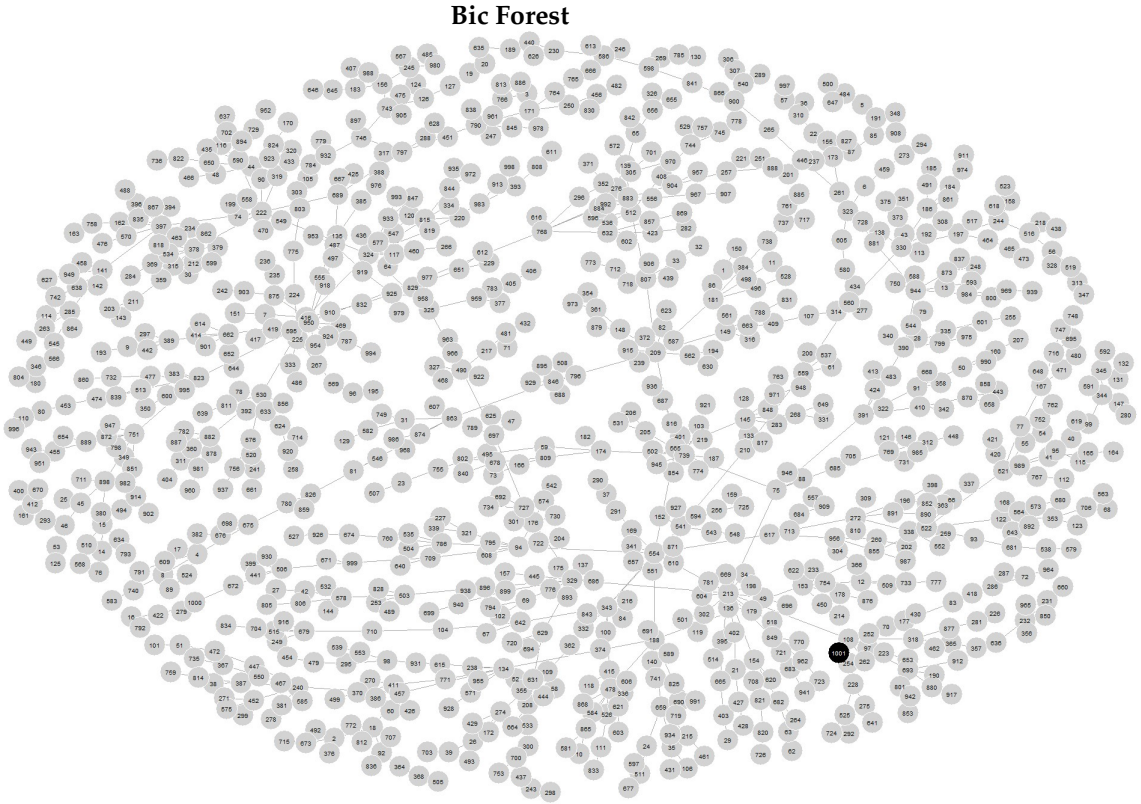


Figure 1.6: Representation of the extended Chow-Liu Algorithm build over the *breast cancer* dataset

The definition of forbidden path is a path between two not adjacent discrete nodes which passes through continuous nodes [106]. The distributional assumption is that

the random variables are conditional Gaussian distributed. A key property of decomposable graph is that densities of such models can be factorized as:

$$f(\mathbf{v}) = \frac{\prod_{C \in \mathcal{C}} f(\mathbf{v}_c)}{\prod_{S \in \mathcal{S}} f(\mathbf{v}_S)^{v(S)}} \tag{1.51}$$

where C is the class of cliques in a perfect sequence, and v($\mathbf{S}$) is the number of times that $\mathbf{S}$ occurs as separator in this perfect sequence. This means that If we are only interested in a subset of such relations, we can define a *subgraph* of $\mathbf{G}$ as $\mathbf{G}_A = (\mathbf{A}, E_A)$ where $\mathbf{A} \subset \mathbf{V}$ and $\mathbf{E}_A \subset \mathbf{E}$. In other words we can consider the variables suggests by the graph to do a feature selection. If $\mathcal{M}_0 \subset \mathcal{M}_1$ are decomposable models differing by one edge $e = (v_i, v_j)$ only, then e is contained in one clique of $\mathcal{M}_1$ only, and the likelihood ratio test for $\mathcal{M}_0$ versus $\mathcal{M}_1$ can be performed as a test of $v_i \perp v_j | C_{\setminus \{v_i, v_j\}}$. These computations only involve the variables in C. It follows that for the likelihood-based scores such as AIC or BIC, score differences can be calculated locally which is far more efficient than fitting both $\mathcal{M}_0$ and $\mathcal{M}_1$. This leads to considerable efficiency gains. To summarize, strongly decomposable model is an important class of model that can be used to analyze mixed data. This class restricts the class of possible interaction models which would be to huge to be explored. Moreover, we have the important results that for strongly decomposable graphical models closed-form estimator exists [92]. In particular, this aspect can be extend in the case when the cliques $\mathcal{C}$ may be quite large and it is often useful to consider potential functions that are themselves factorized in ways that need not be equated with conditional independencies. As suggest [78], we consider a set of 38 factors $\{f_i(X_{\mathcal{C}_i}) : i \in \mathcal{I}\}$ for some index set $\mathcal{I}$, where $\mathcal{C}_i$ is the subset of nodes associated with the ith factor. Note in particular that the same subset can be repeated multiple times (i.e., we allow $\mathcal{C}_i = \mathcal{C}_j$ for $i \neq j$). We define a joint probability by taking the product across these factors:

$$p(x_v) = \frac{1}{\phi} \prod_{i \in \mathcal{I}} f_i(x_{C_i}) \tag{1.52}$$

Factor graphs provide a more fin-grained representation of the factors that make up a joint probability and are useful in defining inference algorithms that exploit this structure. To sum up, all steps of the algorithm to searches an optimal tree or forest is the following:

- 0 Starting from an empty edge set
- 1 Calculate the BIC for all possible edges
- 2 Select the edge that improves the most the model's BIC
- 3 If there is no such edge, stop
- 4 Test if the there is another edge that creates a new path
- 5 If it does, select the best edge and return to Step 3

- 6 Add the edge to E, remove it from the list of possible edges, and return to Step 2.
- Note: a *Forbidden Path* and *Cycles* are avoid

Figure 6 shows the result of the Extended Chow–Liu Algorithm. We applied this algorithm to breast cancer dataset composed of 250 observations and 1001 variables [103]. This dataset is available on R with library(gRbase).

1.7 Conclusion

This paper presented a review of techniques for dimensionality reduction. In this case, we have distinguish between the methods that perform a transformation of the variables to find latent variables and method that perform feature selection. Linear Methods are most useful for dimensional reduction, but them suffer large numbers of variables. On the other hand, nonlinear techniques for dimensionality reduction do not suffer from the presence of trivial optimal solutions, may be based on non-convex objective functions, and do not rely on neighbourhood graphs to model the local structure of the data manifold. The Graphical Models are very powerful techniques to understand the relationship between the variables. This aspect can be used to identify redundancy of information or to select the most relevant variables. Eventually, we underline that the choice of one of these methods will depend by the numbers of variables present in the dataset, by the variables that compose the dataset (if they are continues or not) and by the main goal of the research.

Chapter 2

Use of High Dimensional Modeling for Variables Selection

The Best Path Algorithm

This paper presents a new algorithm for variable selection. In particularly, using the Graphical Models properties it is possible to develop a method that can be used in the contest of large dataset, in order to implement an automatic variable selection. The advantage of this algorithm is that can be combined with different forecasting models. In this research we have used the OLS (Ordinary Least Squares) method and we have compared the result with the LASSO method.

2.1 Introduction

In the last decade the challenge of feature selection becomes a problems in different fields of the research [98]. The dimensionality reduction is a strategy to solve this challenge. Although overtime scholars developed different methods, recently these methods have been challenged by *Big Data Problem*, in which the increasing availability of data is calling for new techniques able to handle not only a large amount of observations, but also rich data sets in terms of number and relations among variables [151]. In general, a dimensionality reduction problem can be viewed as an optimization problem, over a matrix of data: $\mathbf{X}$ [125]. In details, $\mathbf{X}$ is represented in $n \times p$ consisting of a collection of p data vectors $\mathbf{x}_i$ and n observations. The purpose of dimensional reduction is to achieve new dataset $\mathbf{Y}$ with k variables instead of the original dataset $\mathbf{X}$ with dimensionality p, where $k < p$ and often $k \ll p$. In order to obtain this reduction, we can consider two strategies: feature selection and feature extraction [79]. Algorithms based on feature selection, select only those feature from the data set $\mathbf{X}$, which are relevant or significant from the point view of classification or clustering and the

least important features are discarded [114]. In the last decades, this approach was expanded by the *Bayesian Model Selection* [47] [36]. It produces a variable selection based on posterior model probabilities and the corresponding posterior model odds, which are functions of Bayes factors [48]. While feature extraction algorithms utilize all the information contained in measurement space to obtain a new transformed space, thereby mapping high dimensional data to lower dimensional one [80]. [35] explains that the choice of the strategy depends on the problem at hands, and specifically must be examined these three points:

- the dimension of the data-set
- the type of variables contained in the data-set (continues or discrete),
- a *trade-off* between dimensional reduction and loss of information from the variables.

The novelty of this paper is to present an algorithm that it is suitable for each of this point. Indeed, this method is compatible with large datasets, and works with continuous and discrete data [1]. Furthermore, I will show that this algorithm in an exercise of prediction minimizes both the loss of information and its redundancy. More specifically, the algorithm is an application of Probabilistic Graphical Models [78]. They are a form of structural learning in High Dimensional Modeling, where the term *'high-dimensional'* means models with hundreds to tens of thousands of variables [67]. Graphical Models are an elegant framework which combines uncertainty (probability) and logical structure (independence constraints) to compactly represent complex, realworld phenomena [25] and they are widely used, from biological applications [51] to computer science [40] . A Graphical Model is a family of probability distributions defined in terms of a directed or undirected graph where [78]:

- the nodes in the graph are identified with random variables
- the connections between the nodes are defined by the joint probability distributions

The attractiveness of Graphical Models for the problem at hands lies in the fact that they can be interpreted solely in terms of patterns of conditional independence [67]. This feature is functional for two reasons: the structure of the graph allows us to consider the global relationships between the variables and to optimize the feature selection. The aim of this work is to leverage the property of graphical models to select the best subset of variables (covariates) to explain or predict a variable of interest. In doing so, the algorithm exploits mutual information to rank the variables according to their relevance and minimizing the redundancy in the model [87]. In section 2.2, I recall the basic properties of Graphical Models and in Section 2.3, I present the algorithms. Section 2.4 shows an illustrative example of real data and conclusions follow.

2.2 Background Graphical Models

Graphical Models model the conditional independence relationships between random variables in a dataset. Graphically, this relationship is depicted as a network of variables in a graph. We define a graph as a mathematical object $G = (V, E)$, where V is a finite set of nodes with a one-to-one correspondence with random variables in the dataset, and $E \subset V \times V$, is a subset of ordered couples of V, define as edges or links, representing interactions between the nodes. The aim is to estimate the rank of relevance of key variables, consequently following [29] algorithms to find the maximum likelihood tree model. Trees and forest are special case of undirected graph, in details they are an acyclic undirected graph. In other words, an acyclic undirected graph is a tree and the forest is a collection of acyclic undirected graphs (trees). If a link between two nodes is absent, it can be interpreted in terms of conditional independence. Indeed, these two nodes (variables) are conditional independence given the rest of the dataset. Pairwise, local and global Markov properties are the connections between graph theory and statistical modeling [94]. Suppose that the dataset is composed by n observation of p random variables $\mathbf{X}_p$. Where p can be divided in d discrete and q continuous random variables. Given a correspondence one-to-one between the variables and the nodes, we can write the sets of nodes as Δ and Γ, where $V = \{\Delta \cup \Gamma\}$. Let the corresponding random variables be $(\mathbf{Z}, \mathbf{Y})$ where $\mathbf{Z} = (Z_1, ..., Z_d)$ and $\mathbf{Y} = (Y_1, ..., Y_q)$ and a i-observation be $(\mathbf{z}_i, \mathbf{y}_i)$. [29] showed that joint probability distribution of $P(\mathbf{Z})$ can be approximate efficiently with a second order product $P_a(\mathbf{Z})$,i.e:

$$P_a(\mathbf{Z}) = \prod_{i=1}^{p} P(z_i, z_{j(i)}), \quad 0 \leqslant j(i) \leqslant p \tag{2.1}$$

where $(j_1, ..., j_d)$ is an unknown permutation of integers $(1, 2, ..., d)$. Indeed, [29] proved that for discrete random $\mathbf{Z}$, the problem of finding the goodness of approximation between $P(\mathbf{Z})$ and $P_a(\mathbf{Z})$, is the minimization of the closeness measure:

$$I(P, P_a) = \sum_{z} P(\mathbf{z}) \log \frac{P(\mathbf{z})}{P_a(\mathbf{z})} \tag{2.2}$$

where $\sum_z P(\mathbf{z})$ is the sum over all levels of discrete variables. The Equation 2.2, is equivalent to maximizing the total branch (link) weight $\sum_{i=1}^{p} I(z_i, z_{j(i)})$, where:

$$I(z_i, z_{j(i)}) = \sum_{z_i, z_{j(i)}} P(z_i, z_{j(i)}) \log \left(\frac{P(z_i, z_{j(i)})}{P(z_i) P(z_{j(i)})} \right) \tag{2.3}$$

The task is to build a tree or forest of maximum weight. Thanks to [88] algorithm, we can find the trees of minimum of total length. To choose a tree of maximum total branch weight, we first index the $d(d-1)/2$ according to decreasing weight. This algorithm starts from a square weighted matrix $d \times d$, where a weight for a couple of variables (Z_i, Z_j) is given by the mutual information $I(z_i, z_j)$. In the real world the probability

distributions are no given explicitly, for this reason we have to estimate the mutual information. Let $\mathbf{z}^1, \mathbf{z}^2, ..., \mathbf{z}^N$ be independent samples of finite discrete variables $\mathbf{z}$. Then the mutual information is given by:

$$\hat{I}(z_i, z_j) = \sum_{u,v} f_{u,v}(i,j) \log \frac{f_{u,v}(i,j)}{f_u(i) f_v(j)}, \tag{2.4}$$

where $f_{u,v}(i,j) = \frac{n_{uv}(i,j)}{\sum_{uv} n_{uv}(i,j)}$ and $n_{uv}(i,j)$ is the number of samples such that their ith and jth components assume the values of u and v, respectively. It was showed that with this estimator we also maximize the likelihood for a dependence tree [29]. This procedure works only with the discrete random variables, but it can be extended to dataset with both discrete and continuous random variables [42]. To present this extension, we have to consider the distributional assumption of our random variables $\mathbf{X}$ i.e. the distribution of $\mathbf{Y}$ given $\mathbf{Z} = \mathbf{z}$ is a multivariate normal $\mathcal{N}(\mu_i, \Sigma_i)$ so that both the conditional mean and covariance may depend on ith component. The authors [42] distinguish between homogenous and heterogeneous case, if Σ depend by the levels of $\mathbf{Z}$ we are in the homogenous case, otherwise we are in the heterogeneous case. More details this conditional Gaussian distribution can be found in [94]. Before applying the Kruskal's algorithm, we need to find an estimator of the mutual information $I(z_u, y_v)$ between each couple of variables in the mixed case. For a couple of variables (Z_u, Y_v) we can write the sample cell count, mean, and finally the variance, respectively, $\{n_i, \bar{y}_v, s_i^{(v)}\}_{i=1,...,|Z_u|}$. An estimator of mutual information, in the homogenous case is give by:

$$\hat{I}(z_u, y_v) = \frac{N}{2} \log\left(\frac{s_0}{s}\right), \tag{2.5}$$

where $s_0 = \sum_{k=1}^{N} (y_v^{(k)} - \hat{y}_v)/N, \quad s = \sum_{i=1}^{|Z_u|} n_i s_i / N$, and $k_{z_u,y_v} = |Z_u| - 1$ are the degree of freedom associated to the mutual information in the homogenous case. While, in the heterogeneous case an estimator of the mutual information is equal to

$$\hat{I}(z_u, y_v) = \frac{N}{2} \log(s_0) - \frac{1}{2} \sum_{i=1,...,|Z_s|} n_i \log(s_i) \tag{2.6}$$

with $k_{z_u,y_v} = 2(|Z_u| - 1)$ degrees of freedom. According to [42] it is useful to use one of these measures,to avoid inclusion of links not supported by the data:

- $\hat{I}^{AIC} = \hat{I}(x_i, x_j) - 2k_{x_i,x_j}$
- $\hat{I}^{BIC} = \hat{I}(x_i, x_j) - \log(n) k_{x_i,x_j}$

Where k_{x_i,x_j} are the degree of freedom. This aspect is suggested by the algorithm to find the best spanning tree, because it stop when it has added the maximum number of edges. Figure 2.1 shows a representation of a generic dataset $\mathbf{X}$ composed by the variables $\{X_1, X_2, ..., X_{10}\}$. In detail we can approximate the joint probability $P(\mathbf{X})$ of either dataset with $P_a(\mathbf{X})$.

The selection of the variables goes through an important property of the Graphical

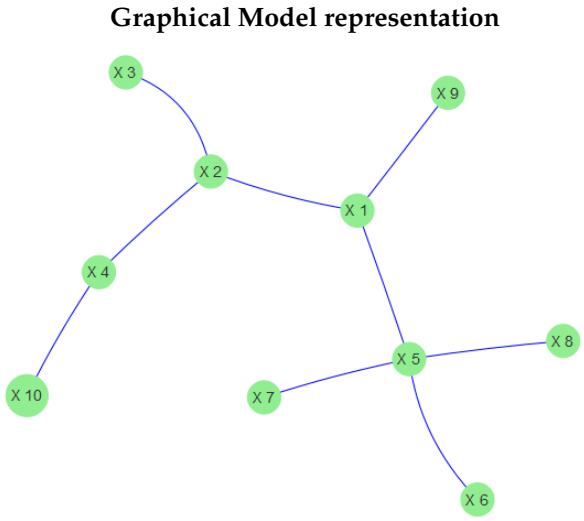


Figure 2.1: Approximation of dataset **X** through a tree

Models: the *strong decomposable* [94]. A mixed model is *strongly decomposable* triangulated and no forbidden paths occurs. In general, a tree is triangulated in every node [92]. While, a forbidden path is a path between two non-adjacent discrete node passing through a continuous node [94]. If we consider two mixed model $\mathcal{M}_0$ and $\mathcal{M}_1$ where $\mathcal{M}_0 \subset \mathcal{M}_1$ and them differ only by one edge $e = (v_i, v_j)$. The Likelihood ration test for $\mathcal{M}_0$ and $\mathcal{M}_1$ can be computed as a test of $v_i \perp v_j | C_{\setminus\{v_i,v_j\}}$, where C is a clique of $\mathcal{M}_1$ that contain e only [1]. Specifically, these computations only involve the variables in C.

2.3 The best path algorithm

The availability of many variables requires specific strategies to avoid the loss of information to build a predictive model and at the same time to work in a lower-dimensional space [80]. This trade-off must have a theoretical coherence in high-dimensional space. For these reasons, the aim of this algorithm is to solve the common challenge encountered when working with high dimensional dataset. In details, this algorithm provides a strategy to select k variables, from a dataset with p variables, where $k < p$. In order to achieve this task, we have implemented a strategy that optimizes the selection. Indeed this algorithm maximizes the relevance of the model with the variables that better explain the variable of interest, and minimizes the redundancy by removing the variables that are statistically irrelevant. For this reason, we can categorize this algorithm as a method for the feature selection. The starting point will be the extension of the Chow-Liu algorithm [42], discussed in the previous section, computed over the entire dataset. In order to chose the optimal number of variables (k), the algorithm considers all possible *path steps* w_i, with $i = 1, ..., N$ starting from node of interest. The path steps w_N are subsets of variables at a specific distance of the variables of interest. The introduction of the path steps, is the cornerstone of the algorithm. In fact, this

approach organizes a hierarchy of the information of the dataset. As we have seen in the section 2.2, the spanning tree can be read as conditional tree. In other words, if two variables are not connected this means that, they are conditional independent given the rest of the variables present in the dataset. For example in the Figure 2.1, the relationship between the variable X_4 and the variable X_1 can be write in the following way: $X_1 \perp X_4 | \mathbf{X} \setminus \{X_1, X_4\}$. In other words, if we want to explain the variable X_1, to produce a coherent variable selection we can consider X_4 only in the case where we have contemplated also X_2. This strategy allows us to have, a limited number of possible candidates for feature selection. In this specific way, we optimize the selection in terms of dependence, since the algorithm is based on the graph structure.

Moreover, if the Graphical Model is composed by an only one component (tree), the algorithm will contemplate all $p-1$ variables inside the dataset, because exist a finite number of path steps between the $p-1$ variables and a node of the tree. On the contrary, if the Graphical Model is composed by more components (forest) in the all possible path steps w_i there will be not all variables $(p-1)$, but only h variables present in the component which contains the node of interest, with $h < p-1$. Figure 2.2 shows an example of feature selection for each path steps w_i for the node Y. As we can see from the figure 2.2, the node X_{13} being an isolated node, is not contemplated in any selection. This aspect is a direct consequence of the structure of the graph that optimizes the feature selection, in contrast with classical methods of statistical learning that contemplate all possible variables in the selection[50]. It is important to underline that the nodes contemplate in a generic path step $w = i$ are all nodes at distance $d \leq i$. Once we got the best subset of the possible variables, we can consider only the variables that are relevant to explain the variable of interest.

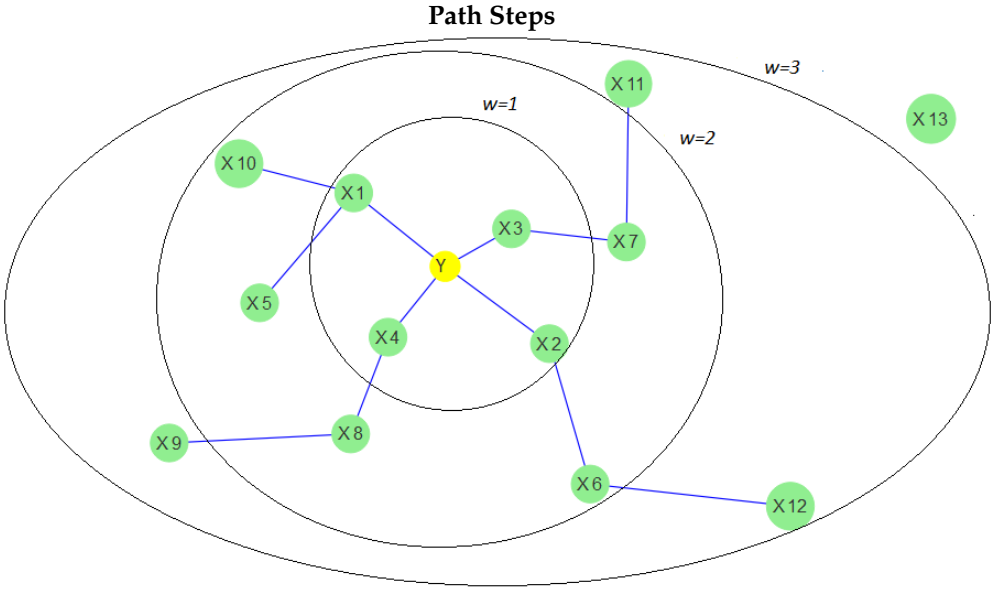


Figure 2.2: Example of tree with Y variable of interest and path steps $w = 1, 2, 3$

I translate this strategy in the following algorithm, where I reported the *pseudo* code:

- **Step 0**: Run the algorithm to find the best spanning tree or forest, and call this model $\mathcal{M}_0$
- **Step 1**: Select the variable of interest and identify all path steps w_i, starting form the variable of interest
- **Step 2**: For $i = 1, ..., N$:
  - (a) Fit the model with k variables present in path step w_i
  - (b) Implement cross-validation, and compute the MSE [1]
- **Step 3**: Pick the best among N models (smallest MSE), and call it $\mathcal{M}_w$
- **Step 4**: Fit the model $\mathcal{M}_w$ and select only the significant variables. Call this model $\mathcal{M}_f$

It is simple to note that $\mathcal{M}_0 \subset \mathcal{M}_w \subseteq \mathcal{M}_f$. This algorithm is an special case of the *mRMRe* approach [87], originally proposed by Battiti [19] that measures the importance of variables based on a relevance penalized by redundancy measure. In this case, the algorithm at **Step 0** produces a rank of the importance for all variables $p-1$, then the from **Step 1** to **Step 3**, it maximizes the relevance by selecting the best subset of variables that explain the variable of interest, and at **Step 4** it and minimizes the redundancy discarding the variables that increase the noise. For instance, if the variable of interest is Y we can write an equation that explains the algorithm' surgery:

$$f(Y; \mathcal{M}_f) = \underbrace{MI(Y; \mathcal{M}_w)}_{Relevance} - \overbrace{MI(Y; \mathbf{S})}^{Redundancy} \tag{2.7}$$

Where **S** denotes the set of variables not significant to explain the variables of interest Y. In other words the information shared between **S** and Y are redundant for the model:

$$y_i = f(w_i) + \epsilon_i \tag{2.8}$$

The *best path algorithm*, belongs to *Sequential Forward Search*. In fact, the search of the algorithm starts with an empty set and keeps on adding features [114]. The advantage of this method is that it starts with an unsupervised learning structure. In fact, we can study the dependence between the variables of the dataset thanks to the extension of the Chow-Liu algorithm [42]. Once selected the node of interest, the analysis becomes supervised learning. In other words, we implement a feature selection according to the structure of the Graphical Model. Figure 2.2 explain better this concept. Indeed, it shows that for the node Y there are well defined the possible subset of the variables

[1] $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2 = \frac{RSS}{n}$, where $\hat{f}(x_i)$ is the estimation of the y_i

candidate for the feature selection. But on the other hand, if we change the node of interest, the algorithm will define other subsets of variables for the feature selection.

There are different papers that investigate of the mutual information criterion to evaluate a set of candidate features and to select an informative subset [61]. Indeed, the mutual information can be consider to evaluate the information content of each individual feature with regard to the output class. Therefore, the mutual information is the amount by which the knowledge provided by the feature vector decreases the uncertainty about the class. The limitations of these algorithms are that some of them work only with variable of interest binary [19] and suffer with large dataset. While the *the best path algorithm* can be used with discrete and continuous outcome, moreover the algorithm is built upon the graphical model, thus it works very well with large data sets.

2.4 Illustrative example

In this section, we present an application of the *best path algorithm*. One of the most important advantage of this algorithm is that it uses econometric techniques to estimate the best model [135]. In other words, it is possible to interpret the coefficient of the model unlike to methods of statistical learning. Furthermore, it could be suitable at other techniques of prediction (SVM [41] or Neural Networks [104]). In this example, we used the ordinary least squares (OLS). In that way, it is possible to combine the interpretability of the coefficients and the prediction of the outcome variable. This algorithm can be apply in different fields, where there is a need of feature selection. For this example, I propose a management dataset which regards the sales of the cars. The *cars sales* dataset includes information about different cars, in particular we have 157 observation and 14 variables that includes *sales* and *price* for each cars. This dataset is being taken from the *Analytixalbs* and is available on *Kaggle* [2]. In this example, the intention is to find the best model to explain *Sales in thousands*. Figure 2.3 shows the *best-spanning tree* for the dataset *car sales*, the yellow vertices indicate the factor variables while the green vertices the continuous variables. As we can see from the Graph, in the car's market the variables *Sales in thousands* and *Price in thousands* are connected given the same features of the car. This is the **Step** 0 of the algorithm, where we can see the relationship between the variables present in the dataset. In some way, the figure 2.3 represents a heuristic way to have an *"information hierarchy"*. This is a key point, because permits to optimization the computation to find the best subset of variables for our model. In the **Step** 1, we select the variable of interest: *Sales in thousands*, then from the Graph we define the different path steps. In the table 2.1, we have the organization of the variables for the distance for the variable *Sales in thousands*. We remember that in path step $w = k$, we consider all variables at distance $d \leq k$.

[2] https://www.kaggle.com/gagandeep16/car-sales

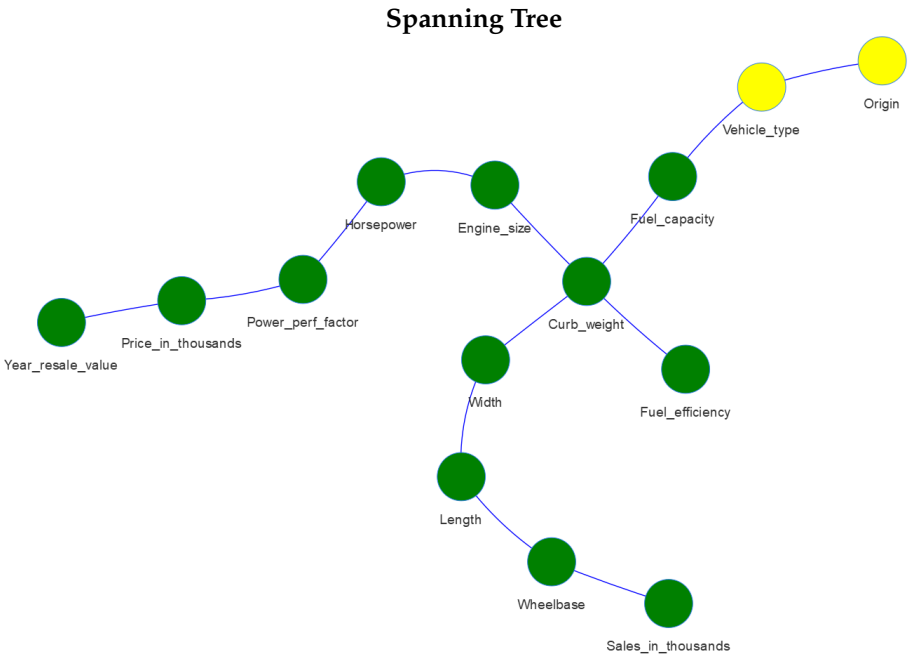


Figure 2.3: The best spanning tree for dataset Car Sales

Organization of the variables

Variables	Distance
Sales in thousands	**0**
Wheelbase	**1**
Length	**2**
Width	**3**
Curb weight	**4**
Engine size	**5**
Fuel capacity	**5**
Fuel efficiency	**5**
Vehicle type	**6**
Horsepower	**6**
Power per factor	**7**
Origin	**7**
Price in thousands	**8**
Year resale value	**9**

Table 2.1: Path steps

Output results

Path	MSE	R^2	MAE
Path 1	**60.9**	**0.203**	**43.9**
Path 2	**61.2**	**0.223**	**43.9**
Path 3	**60.9**	**0.215**	**44.1**
Path 4	**59.0**	**0.238**	**42.7**
Path 5	**60.4**	**0.226**	**43.7**
Path 6	**55.8**	**0.305**	**39.4**
Path 7	**56.6**	**0.285**	**40.9**
Path 8	**62.4**	**0.264**	**43.3**
Path 9	**62.5**	**0.254**	**43.6**

Table 2.2: Models for each path step

In the table 2.2 we have the results of the **Step** 2. The algorithm computes the cross validation for each path-step, and then it selects in **Step** 3 the best path in terms of *MSE*. Finally, in *Step 4* the algorithm removes the variables that are not significant. The model suggested by the algorithm is the follow:

$$Y = \beta_0 + \mathcal{M}_f \times \beta + \epsilon_i \tag{2.9}$$

where Y is *Sales in thousands* and the variables that belong to $\mathcal{M}_f$ are: *Wheelbase,Curb weight, Horsepower* and *Vehicle type*.

2.5 Result

In this section, we show a comparison in terms of prediction with the benchmark method in statistical learning: *LASSO* [140]. This algorithm is a powerful method that performs two main tasks: regularization and feature selection. The table 2.3 shows for each column 100 times of comparison of mean square error between *LASSO* and the *best path algorithm*. For each iteration, we divide the dataset into 70% training and 30% test. In that way, we have the same training data and the same test data for the comparison in each iteration. The percentages indicate the times where the MSE is lower in one of the methods with respect to the other. As the table suggests, the *best path algorithm* performs better than the *LASSO* method for this dataset. The main advantage of respect to the *Statistical learning* methods is that we can interpret the coefficients for each variable.

Computation of Mean Square Error					
Algorithm	**84%**	**83%**	**92%**	**91%**	**89%**
LASSO	**16%**	**17%**	**8%**	**9%**	**11%**

Table 2.3: Comparison between the best path algorithm and LASSO method

2.6 A new representation of the Graphical Model

The algorithm that we have proposed, can be see as a strategy to build a *Factor Graphical Model*. A *Factor Graph* is a bipartite graph where the vertices, **V**, can be divided into two independent sets, V_1 and V_2, and every edge of the graph connects one vertex in V_1 to one vertex in V_2 [12]. This means that a bipartite graph is a graph that does not contain any odd-length cycles [39]. In detail, the Factor Graph express how a global function of many variables factors into a product of local function. For instance, we can write some real valued function $g = (x_1, x_2, x_3, x_4, x_5)$ of five variables can be written as the product of five functions, $f_A, f_B, ..., f_E$.

$$g = (x_1, x_2, x_3, x_4, x_5) = f_A(x_1) f_B(x_2) f_C(x_1, x_2, x_3) f_D(x_3, x_4) f_E(x_3, x_5) \tag{2.10}$$

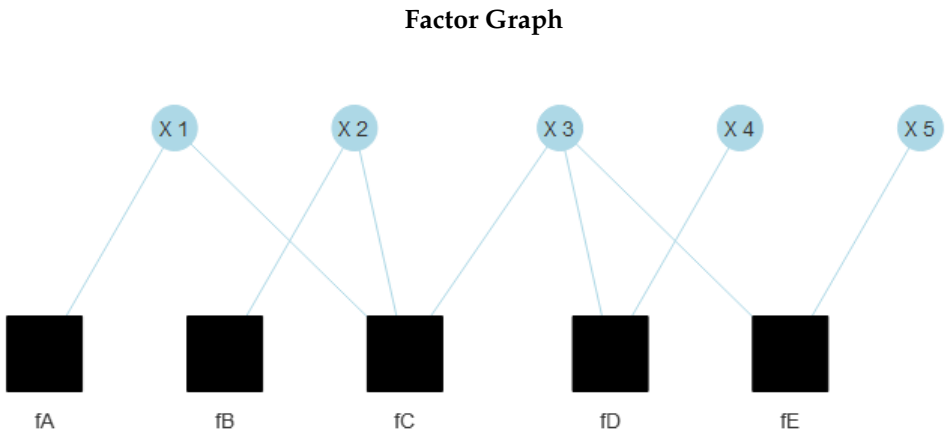


Figure 2.4: A factor graph that expresses that a global function factors as the product of local functions

The corresponding factor graph is shown in Figure 2.4, where we have for each variable a circle and for each factor a square. The variable x_i is connected to the function node for f if and only if x_i is an argument of f. Let $X_S = \{x_i : i \in S\}$ be a collection of the variables indexed by a finite set S, where S is a linearly ordered. For each $i \in S$, the variable x_i takes on value from some set A_i. If E is a subset of S, then we denote by $X_E = \{x_i : i \in E\}$ to be the subset of variables indexed by E. A particular assignment of a value to each of the variables of X_S will be referred to as a configuration of the variables. A configurations of the variables can be viewed as being elements of the

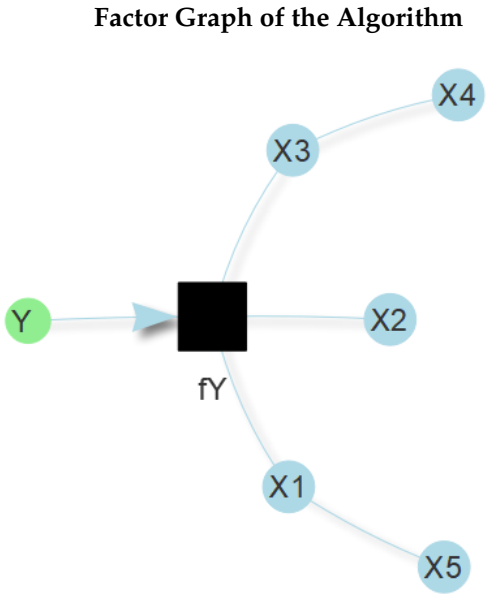


Figure 2.5: An example of factor graph from the algorithm

Cartesian product $A_S \triangleq \prod_{i \in S} A_i$ called *configuration space*. In this context, we should consider also *subconfigurations*: if $E = \{j_1, j_2, ..., j_M\} \subset S$. We can define A_E as all subconfigurations with respect to E; clearly $A_E = \prod_{i \in E} A_i$ [49]. Suppose for some collection Q of subsets of S, that the function g factor as:

$$g(X_S) = \prod_{E \in Q} f_E(X_E) \tag{2.11}$$

where, for each $E \in Q, f_E : A_E \to R$ is a function of the subconfigurations with respect to E. The authors in [49] refer to each factor $f_E(X_E)$, in 2.11 as a *local function.*
A factor graph representation is a bipartite graph denoted by $F(S, Q)$, with vertex set $S \cup Q$ and edge set $\{\{i, E\} : i \in S, E \in Q, i \in E\}$. In words, $F(S, Q)$ contains an edge $\{i, E\}$ if and only if $i \in E$, i.e., if and only if x_i is an argument of the local function f_E. Those vertices that are element of S are called variable nodes and those vertices that are elements of Q are called function nodes. For example the factor graph $F(S, Q)$ in the Figure 2.4 is composed by $S = \{X_1, X_2, X_3, X_4, X_5\}$ and $Q = \{\{X_1\}, \{X_2\}, \{X_1, X_2, X_3\}, \{X_3, X_4\}, \{X_3, X_5\}\}$. In some way, the *best path algorithm* can see as a strategy to build a factor graph, that it maintains the same property of the original graphical models. The authors in [29] showed that joint probability distribution of a dataset $P(\mathbf{X})$ can be approximate efficiently with a second order product $P_a(\mathbf{X})$, as showed in the section 2.2. If we apply the algorithm to find the best subset of variables to explain variable Y we can build a factor graph as in Figure 2.5. In that case we could express the joint probability of entire dataset as:

$$P(Y, X_1, ..., X_5) = P(f_Y(X_1, X_2, X_3))P(X_4|f_Y(X_1, X_2, X_3))P(X_5|f_Y(X_1, X_2, X_3))$$

In other words, a factor graph $F(S, Q)$ produced by the algorithm, started by the graphical model $G(V, E)$ can be factorized the function:

$$g(X_S) = \prod_{E \in Q} f_E(X_E)$$

inside to the dataset the following way:

$$P_a(\mathbf{X}) = \prod_{i=1}^{p} P(X_i, g(X_S))$$

This results gives us the opportunities to explain the variable through a function and in particular to understand the relationship between the variables inside the dataset with the support of the factor graph.

2.7 Conclusion

In the last decade the availability of large dataset it was a reason of the development of methods for the dimensional reduction. Most of them works through the transformation of the variables that not permit to infer directly over the variables. Most of the statistical learning methods belong at this vein [50], but it is not easy to interpret their results. On the other hand,the use of the mutual information for the variable selection was introduced in different paper [19] [16] but these methods suffer in the contest of *big data*. The main motivation for this research was to find a useful way for the automatic selection of the variables. The idea to combine the use of the graphical models as start point, for this selection is is dictated by the need to develop an algorithm for variable selection for very large dataset. Furthermore, the algorithm presented in this research can be adapt with other forecasting models and can be used with discrete and continuous variables.

2.8 Availability

The analyses were performed using the R library gRapHD which we have made available to the R community via the CRAN repository (de Abreu GCG, Labouriau R, Edwards D: High-dimensional Graphical Model Search with gRapHD R package, submitted to J. Stat. Software).

2.9 Appendix

The result of the *LASSO* model are shown in the Figure 2.6. In the abscissa are reported different values of λ. When λ changes the importance of the variables changes consequently. Each line represent on the explanatory variables and its role in the model. In order to choose the most appropriate value for λ, we implement a "*gird*" of λ values,

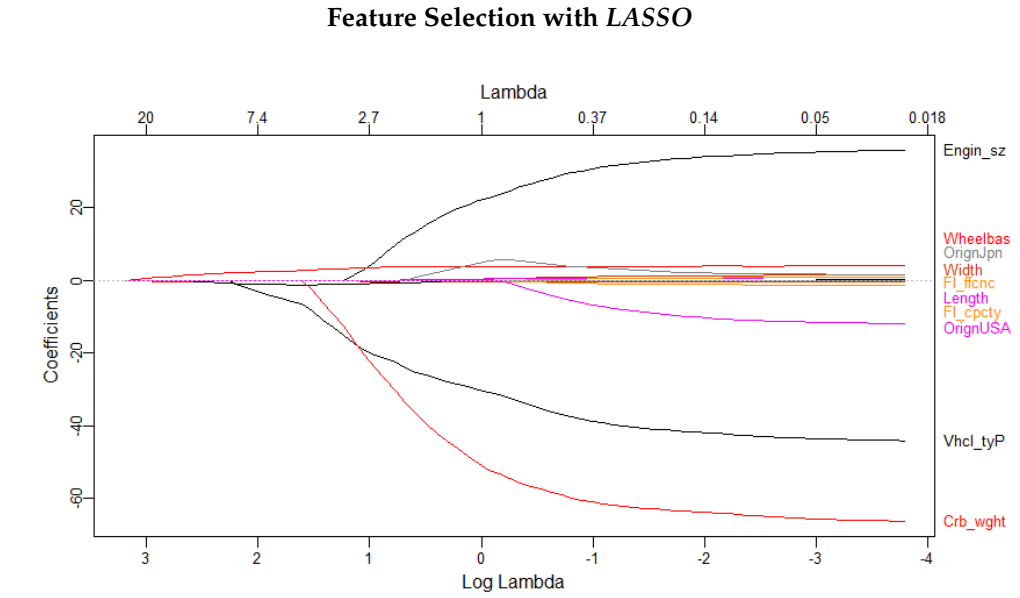


Figure 2.6: Selection of the variables with *LASSO* method

and compute the *cross-validation* error for each value of λ, eventually we choose the value with the smallest *cross-validation* error. Figure 2.7 reported this procedure, the first vertical dotted line represent the value of λ choose for this analysis.

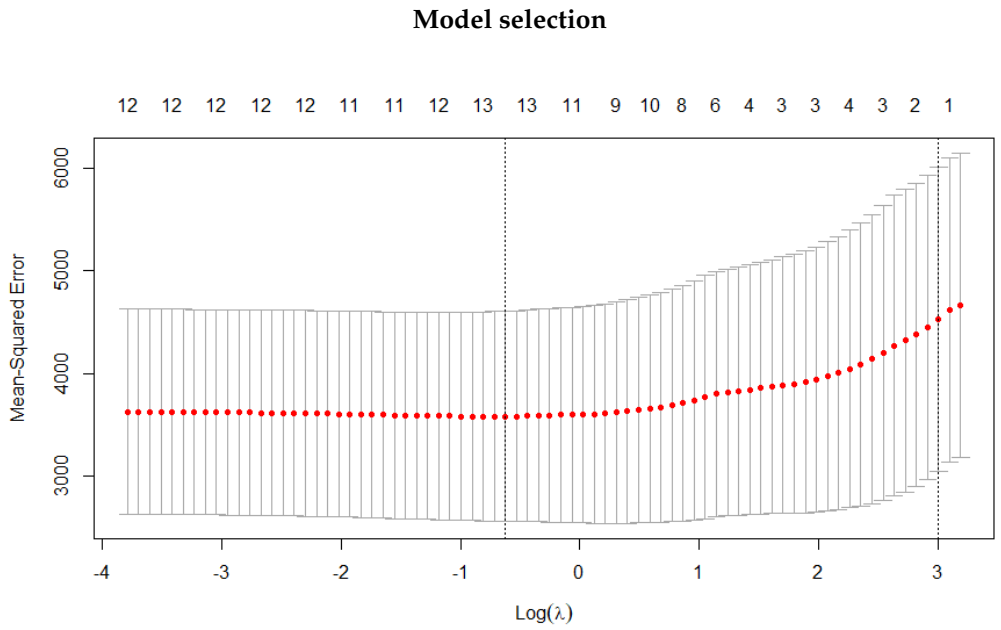


Figure 2.7: Cross-Validation, $n - fold = 10$

Chapter 3

Drift Estimation with Graphical Models

This paper deals with the issue of concept drift in supervised machine learning. We make use of graphical models to elicit the visible structure of the data and we infer from there changes in the hidden context. Differently from previous concept-drift-detection methods, this application does not depend on the supervised machine learning model in use for a specific target variable, but it tries to assess the concept drift as independent characteristic of the evolution of a data set. Specifically we investigate how a graphical model evolves by looking at the creation of new links and the disappearing of existing ones in different time periods. The paper suggests a method that highlights the changes and eventually produce a metric to evaluate the stability over time. The paper evaluate the method with real world data on the Australian Electric market.

3.1 Introduction

In the last decades, both the increasing availability of digitised information and the improvement in the algorithms made the use of machine learning widespread across different industries. Specifically, supervised machine learning became a standard tool for predicting key information in various organization processes such as for instance to mention a few risk default of firms and individual, fraudulent claims, customers churn, and machine failures. The assessment of model uncertainty within a supervised machine learning exercise is based on testing the goodness on a test-set, whose observations have not been employed in the model training. This practice allows for flexibility in the choice of the model and prevents from the risk of over-fitting. However, this analysis relies on the assumption the data generating structure is similar between the test-set and the future observations. While this assumption is rarely debatable in physical process, social process change overtime and a model trained on past data might see a deterioration of its predictive power [55]. This phenomenon is known as concept- or model- drift and describes the situation in which there exists an hidden context of data generative structure, that is any effect of the outcome variable not captured by the model features, which changes over time abruptly, incrementally, or periodically [150,

147]. Scholars addressed this issue and developed a battery of techniques for concept drift detection and early detection. As reviewed in [83] and [44], traditional techniques in concept drift detection typically relies by adopting different time windows or size of the training data [84] or in explaining how the weights of different features change overtime in the outcome prediction [84, 138, 82]. A recent review [10] surveys also methods which can also deal with model update with stream data [23]. However, all of these techniques rely on some sort of computation or statistical comparison of the changes on classification error overtime and from this evidence they deduct the presence of concept drift [150]. In this paper, we approach the problem from a different angle. We make use of graphical models [92] to elicit the visible structure of the data and we infer from there changes in the hidden context with use of statistical measure. To compute the drift we use the Bayesian Regression. This enables us to discern contexts where we have domain knowledge with respect to contexts where we do not have any estimates ahead of time. Thus, differently from previous concept-drift-detection methods, this application does not depend on the supervised machine learning model in use, but it tries to assess the concept drift as an independent characteristic of the evolution of a data set.

3.2 Graphical Models, background

Consider a dataset, composed by p random variables $\mathbf{X}_p$, where p can be divided in d discrete and q continuous random variables. Graphical Models are a method to display the conditional independence relationships between random variables in a dataset. The conditional independence relationships can be showed as a networks of variables with an undirected graph, that is mathematical object $G = (V, E)$, where V is a finite set of nodes, one-to-one correspondence with the p random variables present in the dataset, and $E \subset V \times V$, is a subset of ordered couples of V. Links represent interactions between the nodes. If a link between two nodes is absent, the two variables represented by the node are conditional independent given the dependence of the remaining variables.

Pairwise, local and global Markov properties are the connections between graph theory and statistical modeling [92]. As said before, there exist a one-to-one correspondence between the variables and the nodes in the graph and, for this reason, the sets of nodes is Δ and Γ, where $V = \{\Delta \cup \Gamma\}$. Let the corresponding random variables be $(\mathbf{Z}, \mathbf{Y})$ where $\mathbf{Z} = (Z_1, ..., Z_d)$ and $\mathbf{Y} = (Y_1, ..., Y_q)$ and a i-observation be $(\mathbf{z}_i, \mathbf{y}_i)$. This means that $\mathbf{z}$ is a d-tuple containing the values of discrete variables, and $\mathbf{y}$ is a real vector of length q. Our interest is to estimate the joint probability distribution $P(\mathbf{x})$ for the random variables $(\mathbf{Z}, \mathbf{Y})$ to build a conditional (undirected) graph from the data. A product approximation of $P(\mathbf{x})$ is defined to be a product of several of its component distribution of lower order $P_a(\mathbf{x})$. As suggest [29], we can consider the class of second-order

approximation, i.e:

$$P_a(\mathbf{x}) = \prod_{i=1}^{p} P(x_i, x_{j(i)}), \quad 0 \leqslant j(i) \leqslant p \tag{3.1}$$

where $(j_1, ..., j_p)$ is an unknown permutation of integers $(1, 2, ..., p)$, where p=d+q. Chow and Liu in [29] proved that for discrete random $\mathbf{Z}$, the problem of finding the goodness of approximation between $P(\mathbf{z})$ and $P_a(\mathbf{z})$ with the minimization of the closeness measure:

$$I(P, P_a) = \sum_{z} P(\mathbf{z}) \log \frac{P(\mathbf{z})}{P_a(\mathbf{z})} \tag{3.2}$$

where $\sum_z P(\mathbf{z})$ is nothing more than the sum over all levels of discrete variables. The equation (2), is equivalent to maximizing the total branch (link) weight $\sum_{i=1}^{p} I(z_i, z_{j(i)})$, where:

$$I(z_i, z_{j(i)}) = \sum_{z_i, z_{j(i)}} P(z_i, z_{j(i)}) \log \left(\frac{P(z_i, z_{j(i)})}{P(z_i) P(z_{j(i)})} \right) \tag{3.3}$$

The task is to build a tree or forest (different trees) of maximum weight. We make use of the Kruskal's algorithm [88] to compute trees with the minimum of total length. To choose a tree of maximum total branch weight, we first index the $d(d-1)/2$ according to decreasing weight. This algorithm starts from a square weighted matrix $d \times d$, where a weight for a couple of variables (Z_i, Z_j) is given by the mutual information $I(z_i, z_j)$. In the real world the probability distributions are no given explicitly, for this reason we have to estimate the mutual information. Let $\mathbf{z}^1, \mathbf{z}^2, ..., \mathbf{z}^N$ be independent samples of finite discrete variables $\mathbf{z}$. Then the mutual information is given by:

$$\hat{I}(z_i, z_j) = \sum_{u,v} f_{u,v}(i, j) \log \frac{f_{u,v}(i, j)}{f_u(i) f_v(j)}, \tag{3.4}$$

where $f_{u,v}(i, j) = \frac{n_{uv}(i,j)}{\sum_{uv} n_{uv}(i,j)}$ and $n_{uv}(i, j)$ is the number of samples such that their ith and jth components assume the values of u and v, respectively. It was showed that with this estimator we also maximize the likelihood for a dependence tree [29]. This procedure works only with the discrete random variables, but it can be extended to data with both discrete and continuous random variables [42]. To present this extension, we have to consider the distributional assumption of our random variables $\mathbf{X}$ i.e. the distribution of $\mathbf{Y}$ given $\mathbf{Z} = \mathbf{z}$ is a multivariate normal $\mathcal{N}(\mu_i, \Sigma_i)$ so that both the conditional mean and covariance may depend on ith component.

We distinguish between homogenous and heterogeneous case, if Σ depend on i we are in the homogenous case, otherwise we are in the heterogeneous case. More details this conditional Gaussian distribution can be found in [136]. Before to apply the Kruskal's algorithm, we need to find an estimator of the mutual information $I(z_u, y_v)$ between each couple of variables in the mixed case.

For a couple of variables (Z_u, Y_v) we can write the sample cell count, mean, and finally the variance, respectively, $\{n_i, \bar{y}_v, s_i^{(v)}\}_{i=1,...,|Z_u|}$. An estimator of mutual information, in

the homogenous case is give by:

$$\hat{I}(z_u, y_v) = \frac{N}{2} \log\left(\frac{s_0}{s}\right), \tag{3.5}$$

where $s_0 = \sum_{k=1}^{N}(y_v^{(k)} - \hat{y}_v)/N$ and $s = \sum_{i=1}^{|Z_u|} n_i s_i / N$. $k_{z_u,y_v} = |Z_u| - 1$ are the degree of freedom associated to the mutual information in the homogenous case.
While, in the heterogeneous case an estimator of the mutual information is equal to

$$\hat{I}(z_u, y_v) = \frac{N}{2} \log(s_0) - \frac{1}{2} \sum_{i=1,\ldots,|Z_s|} n_i \log(s_i) \tag{3.6}$$

with $k_{z_u,y_v} = 2(|Z_u| - 1)$ degrees of freedom. According [42] it is useful to use either $\hat{I}^{AIC} = \hat{I}(x_i, x_j) - 2k_{x_i,x_j}$ or $\hat{I}^{BIC} = \hat{I}(x_i, x_j) - \log(n)k_{x_i,x_j}$, where k_{x_i,x_j} are the degree of freedom, to avoid inclusion of links not supported by the data. This aspect is suggested by the algorithm to find the best spanning tree, because it stop when it has added the maximum number of edges. Furthermore the algorithm avoid inside the tree a forbidden path. The definition of forbidden path is a path between tow not adjacent discrete nodes which passes through continuous nodes [2]. However, we can start from the best spanning tree and determine the best strongly decomposable graphical model. A strongly decomposable graphical model whose graph neither contains cycles of length more than three nor forbidden path. Strongly decomposable model is an important class of model that can be used to analyze mixed data. This class restrict the class of possible interaction model which would be to huge to be explored [1]. The graph build to find the best spanning tree, can be see with a symmetric adjacency matrix AM, with dimension $V \times V$, in which each element takes value of 1 if an edge exists between two of the V variables, and zero otherwise. Elements in the main diagonal are zeros, since self-loops are not allowed.

3.3 A measure of dynamic stability as proxy for the model drift

Considering the additional dimension of time t to the dataset of N observations and p variables as a tensor $\mathbf{X}$ with dimension $(N \times p \times T)$, we are interested in modeling the evolution of the joint probability $P(X_1, ..., X_p)$ over T time periods. In other words, considering the graph G, with $V = p$ vertices of the maximum spanning tree with mutual information as express in Eq. 3.6 for each period $t = 1, ..., T$ and the corresponding T adjacency matrices AM_t. The aim of the paper is to describe how the graphs, as represented by their adjacency matrix AM_t with $t = 1, 2, ...T$, change over time.

3.3.1 Transition Matrix Processes

In order to accomplish this task, we analyse the transition process which connects the original adjacency matrix AM_1 to any adjacency matrices in a subsequent period AM_T.

We first introduce a function which maps any possible state of AM_t into a transition matrix $TM = f(AM_t)$ with $t = 1, 2, 3, ..T$, noted TM_T, of dimension $V \times V$. Its generic element $w_{i,j}$ registers all possible states of dependence of any couple of variable V_i and V_j in T periods. Specifically, the function takes the following form:

$$TM_t = \sum_{t=1}^{T} 2^{(T-t)} AM_t \tag{3.7}$$

For the sake of clarity, the following paragraph describes the process up to $T = 3$ and, thereafter, generalizes for T periods.

	AM_1	AM_2	AM_3	TM_3
	0	0	0	0
	1	0	0	4
	1	1	0	6
	1	1	1	7
	0	1	0	2
	0	0	1	1
	1	0	1	5
	0	1	1	3

Table 3.1: All possible AM_t values for two nodes i and j and the resulting $w_{i,j}$ in TM_T function for $T = 3$

As a starting point, in $t = 1$ the transition matrix TM_1 is equal to the adjacency matrix $AM_{t=1}$, where $w_{i,j;1} = 0$ means that the i-node and j-node are not connected, while when $w_{i,j;1} = 1$ means that the i-node and j-node are connected. At $t = 2$ existing links can persist or not, while non-existing links can appears or not. From Eq. 3.7,

$$TM_2 = 2 \times AM_1 + AM_2 \tag{3.8}$$

Thus, TM_2 maps any possible evolution of connections $w_{i,j;2}$ with values $\{0, 1, 2, 3\}$. When V_i and V_j are never connected,that is $AM_{i,j;t=1} = AM_{i,j;t=2} = 0$, then $TM_{i,j;2} = 0$. If V_i and V_j stay connected, that is $AM_{i,j;t=1} = AM_{i,j;t=2} = 1$, then $w_{i,j;2} = 3$. For $AM_{i,j}$ changing from 0 in $t = 1$ to 1 in $t = 2$ and viceversa, we have $w_{i,j;2} = 2$ and $w_{i,j;2} = 1$, respectively. At time $t = 3$ the possible evolution of AM can be described has 8 levels, given by:

$$TM_3 = 2^2 \times AM_1 + 2^1 \times AM_2 + 2^0 \times AM_3 \tag{3.9}$$

Table 3.1 summarizes all possible combinations between two nodes of binary values of the AM_t in the three periods, mapped on TM_3. Generally, for time T we can derive Eq. 3.7:

$$
\begin{aligned}
TM_2 &= 2 \times AM_1 + AM_2 \\
TM_3 &= 2 \times TM_{1,2} + AM_3 \\
TM_3 &= 2 \times (2 \times AM_1 + AM_2) + AM_3 \\
TM_3 &= 2^2 \times AM_1 + 2^1 \times AM_2 + 2^0 \times AM_3 \\
TM_3 &= \sum_{t=1}^{T} 2^{(3-t)} AM_t \\
&\dots \\
TM_T &= \sum_{t=1}^{T} 2^{(T-t)} AM_t
\end{aligned}
\tag{3.10}
$$

In general, the value of the generic element $w_{i,j;t} \in \mathcal{W} \subset \mathbb{N}$ of TM_t can be considered as a discrete random variable with density $f(w_{i,j;t})$

$$
f(w_{i,j;t}) = P(\mathcal{W}_{i,j;t} = w_{i,j;t}), \quad t = 2, ..., T \tag{3.11}
$$

Thus, $w_{i,j;t}$ represents the evolution of the connection between i-node with j-node at time T, for each node V. The numerosity of the set $\mathcal{W}_{i,j;T} = \{0, 1, 2.., 2^T - 1\}$ is 2^T.

3.3.2 From the transition process to stability

The main idea of the paper is to consider as a proxy for the model drift the appearance or disappearance of connections between nodes, that is changes of the conditional independence structure of a dataset over time. For this reason, we are specifically interested in two specific levels. The one describing the state of the word in which a connection between two nodes never exists, that is $AM_{i,j;t} = 0 \; \forall \; t$ and the one describing a stable connection over time, that is $AM_{i,j;t} = 1 \; \forall \; t$. For the case $T = 3$, the two cases map into $w_{i,j;3} = 0$ and $w_{i,j;3} = 7$, as showed in Table 3.1. In general for a generic T, we have a stability of connections when connections are always absent, with $w_{i,j;T} = 0$, or always existing, with $w_{i,j;T} = 2^T - 1$. This transition process is a partition process (Fig 3.1) of the set of $\mathcal{V}$ possible connections between the **V** nodes in the undirected graph: $\mathcal{V} = \frac{V(V-1)}{2}$. Each transition in time t generates a subsequent partition of $\mathcal{V}$, one of whose will always contain elements for which $w_{i,j;t} = 0$ or always $w_{i,j;t} = 2^t - 1$. This *transition processes* is a special case of the Tail-free processes [74]. Consider a sequence $\mathcal{T}_0 = \{\mathcal{V}\}$, $\mathcal{T}_1 = \{A_0, A_1\}$, $\mathcal{T}_2 = \{A_{00}, A_{01}, A_1\}$, and so on, of measurable partitions of the $\mathcal{V}$ elements, obtained by slitting every set in the preceding partition into two new sets for the node on left and maintain the same node for the others.

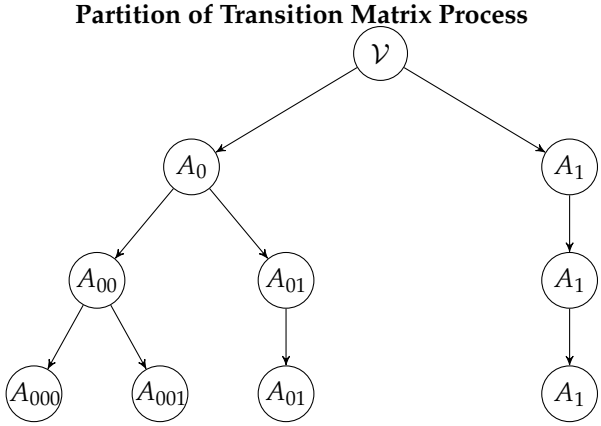


Figure 3.1: Representation of *Transition Matrix process* with Tail-free processes

Specifically, at each time t we can partition the elements between stable and unstables ones. Fig. 3.1 shows a tree diagram that represents the distribution of mass over time $\mathcal{V} = A_0 \cup A_1 = (A_{00} \cup A_{01}) \cup A_{10}$ of the elements at each time. A_0 contains elements for $w_{i,j,2} = \{0, 3\}$, that is stable connections while A_1, the remaining ones. At the subsequent period, A_0 is partitioned between A_{00}, in which connection remain stable with $w_{i,j,3} = \{0, 7\}$,while $A_{01} = \{1, 6\}$ and A_1 the remaining ones.

Clearly, every partition is composed by the union of all possible evolution of the connection given by the levels of $\mathcal{W}$, and, by construction, there is always a partition with elements $w_{i,j;t} = 0$ and $w_{i,j;t} = 2^t - 1$, that containing stable links between the i-node and the j-node until time t.
We describe this process as a variable $Y_{i,j;t}$ with values :

$$Y_{i,j;t} = \begin{cases} y_{i,j;t} = 1 & \text{if } w_{i,j;t} = 0 \vee w_{i,j;t} = 2^t - 1 \\ y_{i,j;t} = 0 & \text{otherwise} \end{cases}, \quad t = 2, ..., T \tag{3.12}$$

Thus, $Y_{i,j;t}$ is indicate persistent status of dependence over time $Y_{i,j;k} = 1$ or not $Y_{i,j;k} = 0$. Be Y_t the vectorization of $Y_{i,j;t}$, $vec(Y_{i,j;t}) = Y_t$ with length $\mathcal{V} = \frac{V \times (V-1)}{2}$, that is at each time we observe the stability of the $\mathcal{V}$ connection between each possible pair of nodes. The structure of the *transition matrix process* depend by the spanning forest at time $t = 1$, and for each period we have a partition of $\mathcal{V}$ given by $\mu_t = \sum_{i=1}^{N} Y_{i,t}$ with $t = 1, ..., T - 1$.

Therefore, we pool together the $T - 1$ periods and define *Stability*, the resulting variable Y with length $n = \mathcal{V} \times (T - 1)$. *Stability* is the cornerstone of our strategy to estimate an empirical measure of model drift.

3.3.3 The stability index

In this section we introduce the *Stability* as a latent variable, which capture stability of connection of a graph overtime.
Consider the following variable with same length $i = 1, ..., n$:

- Y, *Stability* as defined above
- $W = vec(TM_{i,j,t})$ that is the vectorization of the value $w_{i,j;t}$ of TM.
- T the corresponding time for each Y_i.

We build a dataset with this variables and call it **D**. Note that by construction the observations of **D** is exchangeable since we have built **D** respecting the temporal period of the *adjacent matrices*, thus:

$$P(\mathbf{D}_1, ..., \mathbf{D}_n) = P(\mathbf{D}_{\sigma(1)}, ..., \mathbf{D}_{\sigma(n)})$$

for all $n \geq 1$ and all permutations σ of $(1, ..., n)$. In other words, the order of appearance of the observation does not matter in terms of their joint distribution. In order to exploit this property, we implement a Bayesian Regression Model over the dataset **D** [7]. One advantage of a Bayes perspective is the opportunity to consider the contest of analysis with the support of the prior distribution. This approach is functional to discern the situations when the data generating structure is similar between the test-set and the future observations as a physical process, with respect to a social process that changes more fast overtime [146].

Let θ_i the probability of a realization of $Y_i = 1$ of *Stability* with odds of stability $\frac{\theta_i}{1-\theta_i}$. Thus the dichotomous variable Y can be described by a Bernoulli distribution with probability of success θ_i:

$$Y_i|\theta_i \overset{ind}{\sim} Bern(\theta_i), \quad i = 1, .., n$$

Consider a logistic regression model[1], which writes that the logit of the probability θ_i, or the log of the its odd is a linear function of some predictor variables $\mathbf{x}_i$:

$$\text{Logit}(\theta_i) = \log\left(\frac{\theta_i}{1-\theta_i}\right) = \beta_0 + \sum_{j=1}^{2^T} \beta_j \mathbf{x}_{j,i} \tag{3.13}$$

where the j predictors are T, that is the time of the realization of Y and W, that is the corresponding value. Since W has 2^t levels, we regress $2^t - 1$ dummy variable and keep $W = 0$ as the reference category:

$$\log\left(\frac{\theta_i}{1-\theta_i}\right) = \beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i} \tag{3.14}$$

By construction, the intercept of this model β_0 can be interpreted as the baseline risk for *Stability*. A high β_0 suggests that the underlying graphical model is not changing much over time. β_t captures the effect of the drift over time. It can be shown

[1]The logistic regression seem the most natural way to describe this phenomenon. However, according to the type of expected drift, we could employ other function, without loss of generalization.

that *Stability* is weakly decreasing over time and, thus β_1 define the speed of convergence towards the absence of stability. Finally, since the variable Y takes value 1 for $W_{i,j} = (0; 2^T - 1)$, the coefficient β_{2^T-1}, that is the coefficient for $W_{i,j} = 2^T - 1$ with reference $W_{i,j} = 0$ captures which component of *Stability* originates in the persistence of existing connections, rather than on the persistence of absence of connections.

The computation is straightforward: by rearranging the logistic regression Equation 3.13. It is possible to express the regression as a nonlinear equation for the probability of success θ_i :

$$\begin{aligned} \log\left(\frac{\theta_i}{1-\theta_i}\right) &= \beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i} \\ \frac{\theta_i}{1-\theta_i} &= \exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\} \\ \theta_i &= \frac{\exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}}{1 + \exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}} \end{aligned} \tag{3.15}$$

From the Equation 3.15 we can define the likelihood for the sequence of Y_i over data set of n subjects is then

$$\begin{aligned} p(\mathbf{D}|\beta_0, \beta_1, \boldsymbol{\beta}_j) = \prod_{i=1}^{n} \Bigg[&\left(\frac{\exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}}{1 + \exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}}\right)^{y_i} \\ &\left(1 - \frac{\exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}}{1 + \exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}}\right)^{(1-y_i)} \Bigg] \end{aligned} \tag{3.16}$$

where $\mathbf{D}$ is the dataset composed by T_i and the corresponding dummy variables generated by the level of W_i. The set of unknown parameters consists of $\beta_0, \beta_T, ..., \beta_{2^T-1}$. In general, any prior distribution can be used, depending on the available prior information. The literature suggests the use of informative prior distributions if something is known about the likely values of the unknown parameters, otherwise, the use of non-informative prior if either little is known about the coefficient values or if one wishes to see what the data themselves provide as inferences. In this case, we will use the most common priors for logistic regression parameters:

$$\beta_j \sim N(\mu_j, \sigma_j^2) \tag{3.17}$$

The most common choice for μ is zero with σ large enough to considered as non-informative in the range from $\sigma = 10$ to $\sigma = 100$. The posterior distribution of $\boldsymbol{\beta}_j$ is extrapolated by combining likelihood Eq. 3.16, with the prior in Eq. 3.17:

$$p(\beta_0, \beta_1, \boldsymbol{\beta}_j|\mathbf{D}, \sigma_j, \mu_j) = \prod_{i=1}^{n} \left[\left(\frac{\exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}}{1 + \exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}} \right)^{y_i} \right.$$
$$\left. \left(1 - \frac{\exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}}{1 + \exp\left\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^T-1} \beta_j \mathbf{w}_{j,i}\right\}} \right)^{(1-y_i)} \right] \tag{3.18}$$
$$\times \prod_{j=0}^{p} \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left\{ -\frac{1}{2} \left(\frac{\beta_j - \mu_j}{\sigma_j} \right)^2 \right\}$$

Now, we are not that much interested in the regression parameters $\boldsymbol{\beta}_j$, we want to find the posterior probability distribution of the *stability*. Furthermore, this model gives us the opportunity to compute the prediction of the *stability* over a specific time t. If $\widetilde{y}_i$ represents the number of similarity connection between n nodes at time t, then one would be interested in the posterior predictive distribution of the fraction $\widetilde{y}_i/n$ One represents this predictive density of $\widetilde{y}_i$ as:

$$f(\widetilde{Y}_i|y) = \int p(\beta_0, \beta_1, \boldsymbol{\beta}_j|\mathbf{D}, \sigma_j, \mu_j) p(\widetilde{y}_i, \mathbf{D}|\beta_0, \beta_1, \boldsymbol{\beta}_p) d\boldsymbol{\beta} \tag{3.19}$$

where $p(\beta_0, \beta_1, \boldsymbol{\beta}_j|\mathbf{D}, \sigma_j, \mu_j)$ is the posterior density of $\boldsymbol{\beta}$ and $p(\widetilde{y}_i, \mathbf{D}|\beta_0, \beta_1, \boldsymbol{\beta}_j)$ is the Binomial sampling density of $\widetilde{y}_i$ conditional of regression vector $\boldsymbol{\beta} = (\beta_0, \beta_1, \boldsymbol{\beta}_p)$. Figure 3.2 represents the Bayesian graphical model of the stability, in particular, we can see all process that describes from the adjacent matrix to the coefficients of the logistic, that say us how changes the relationship between the variables over the time. Where we have an adjacent matrix (AM) for each time t, for each pair sequential of the AM we have a transition matrix TM. From the TM we can build the dataset to compute the stability with n observation, where $n = \mathcal{V} \times (T - 1)$, and three variables: **W,T,Y**.

3.4 Empirical experiment

As a test bed for this theoretical approach, we apply the stability index to the *ELEC2* dataset [64], a benchmark for drift evaluation [15, 90, among the many]. It holds information on the Australian New South Wales (NSW) Electricity Market, containing 27552 records dated from May 1996 to December 1998, each referring to a period of 30 minutes. These records have 5 fields: a binary class label Y and four covariates X_1, X_2, X_3 and X_4 capturing different aspects of electricity demand and supply. In order to compute the empirical evolution of the drift over time, we group observations in one week period. Thus, for each week we have a panel dataset of 5 variables and 336 observation. Thus, we have a tensor $\mathbf{X}$ with dimension $(N \times p \times T)$ with $N = 336$ records for a week, $p = 5$ the variables as described above and $T = 82$ temporal periods.

First, we realize a Graphical Models for each period t as the start point of our strategy to compute the drift. Figure 3.3 portraits the graphs for some selected periods

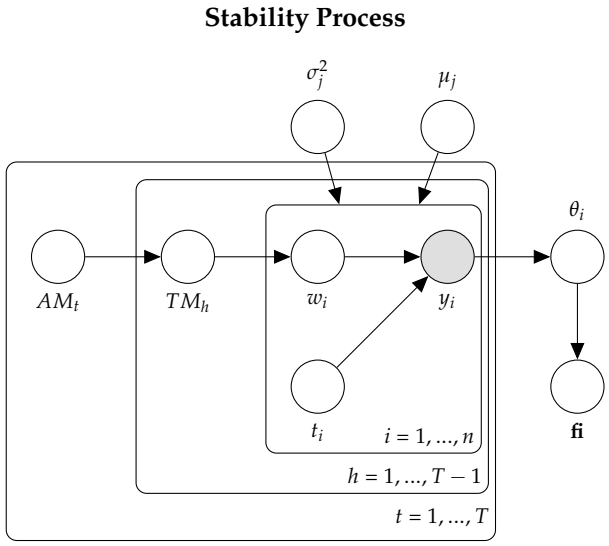


Figure 3.2: Bayesian Graphical Model of the *stability*

and shows that the structure of the graph changes overtime. We thus expect a presence of the drift.

Figure 3.4 depicts the evaluation of the drift overtime. The red dots are the percentage of stable relations among variables, that is the the sum of variable $Y_{i,t}$ in Equation 3.12, while the blue line is the estimation of the Equation 3.18 with its related confidence interval as the gray contour. The figure highlights 6 periods of drift. The different *Stability* values are reported in the table 3.2. In the table 3.3 are reported the magnitude of the coefficients for the baseline β_0 or intercept, β_{2^T-1} for the $W = 2^T - 1$ with reference level $W = 0$ and for the time β_{time}.

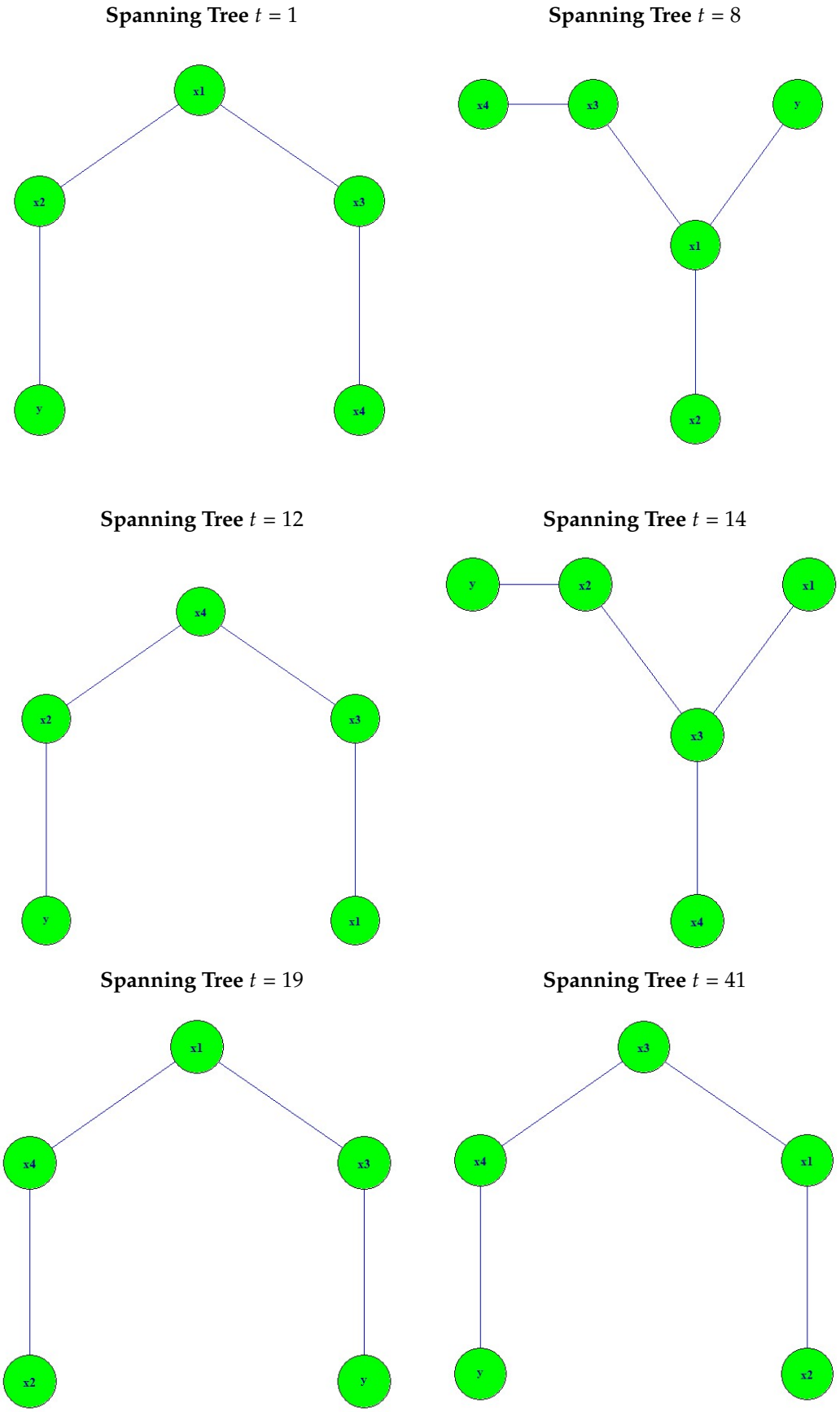


Figure 3.3: Graph over the time

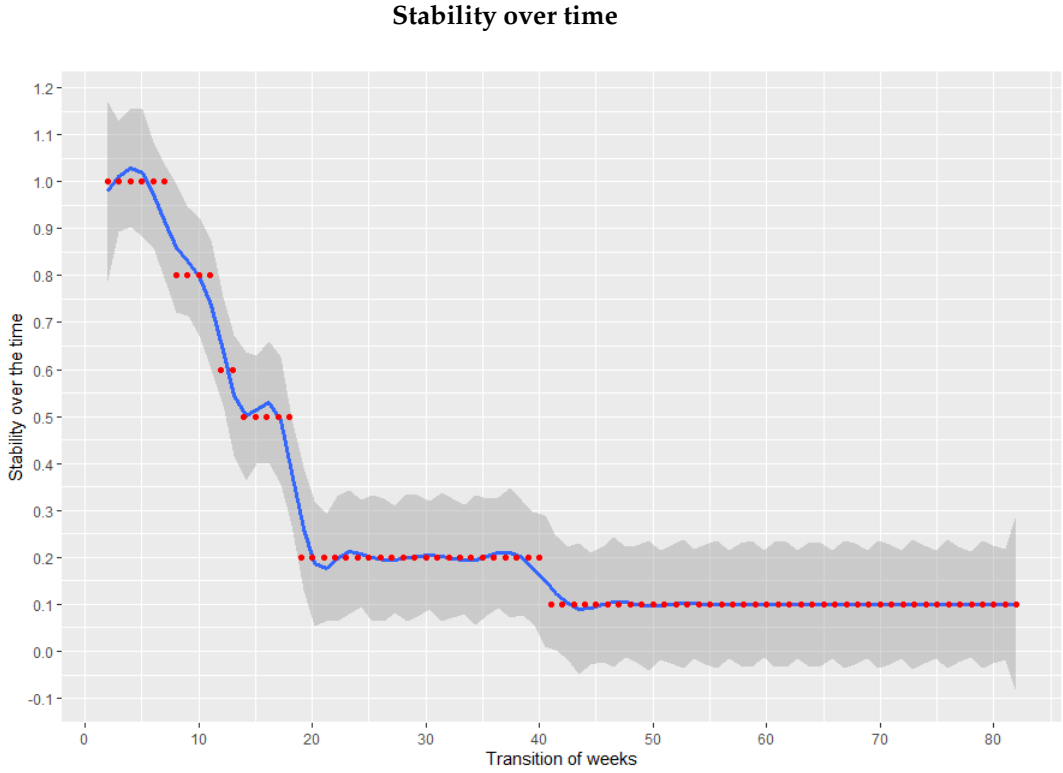


Figure 3.4: Evolution of Stability

Percent of Stability	**Evolution of the Drift**					
	$ty = 2$	$ty = 8$	$ty = 12$	$ty = 14$	$ty = 19$	$ty = 41$
$\frac{\sum_{i=1}^{N} Y_{i,t}}{N}$	1.0	0.8	0.6	0.5	0.2	0.1

Table 3.2: Approximation of the drift for selected period

Regression Summary

Coefficients	**Estimation**
β_0	**7.66**
β_{2^T-1}	**19.75**
β_{Time}	**-0.30**

Table 3.3: Coefficients of logistic regression

3.5 Conclusion

This paper presented an algorithm to estimate the magnitude of a model drift in a context of machine learning. While past solutions relies on how the classification errors of a specific target variable changes over time, the present method tries to describe the underlying hidden context with the use of graphical models and to estimate how the observable context changes over time. Specifically, we provide not only an assessment

of the drift, which is independent from the model in use, but also an estimation of the confidence interval of this prediction. These two characteristics combined together allow to signal when a data driven process shows an excessive risk due to the drift and needs to be retrained or re-calibrated. Possible applications are countless such as predicting defaults, online recommendations systems, or spam filtering. More specific, any prediction which involves human behaviour is prone to constant changes in the data generating process, while biological and physical phenomena tend to be more stable over time. Further lines of research in this area include a fine tuning for estimating different type of drift, allowing for temporary drift, and testing the index on a wider array of applications.

Chapter 4

Variable selection and dynamic stability with graphical models

A survival analysis of Italian start-ups

The paper investigates the economic performance of new firms born during the 2009 crisis with a focus on survival and value creation. Since it leverages on a very large dataset traditional econometrics techniques require an ex-ante sound variable selection to avoid excessive loss of relevant information. The most important vantage of this approach is that we can identify the most relevant variables for each years. These selections are justified by the presence of the drift. We make use of graphical models to elicit the visible structure of the data and we infer from there changes in the hidden context. Differently from previous concept-drift-detection methods, this application does not depend on the supervised machine learning model in use, but it tries to assess the concept drift as independent characteristic of the evolution of a data set. Specifically we investigate how a graphical model evolves by looking at the creation of new links and the disappearing of existing ones in different time periods. The paper suggests a method that highlights the changes and eventually produce a metric to evaluate the stability over time. Moreover, the use of graphical model allows us to understand the differences in the relationships among variables in firms subgroups, for instance survived firms vs non-survived or firms from different geographical location.

4.1 Introduction

In this paper we address the issue of variable selection in survival models. Although survival models root in biomedical research, they have been widely employed in the analysis of firms' survival. Survival models have three characteristics: the dependent variable is the time span before the realization of an event; data are right censored, since some observations do not experience the event; there are covariates explaining

the average waiting time for the event to occur and they can be considered either risk factors to be analyzed or control variables, depending on the problem at hand.

The present digital revolution presents the researcher with very rich data-sets in survival analysis which can greatly improve both prediction capability and the analysis of specific risk factors of interest. However, the vast availability of data might create an embarrassment of riches [11, 43] in the choice of the variables since the requirements of clarity, exogeneity of the covariates, degrees of freedom, and collinearity issues force the researcher to make an educated selection.

Although in the age of digital information the issue of variable selection is not limited to survival models only, we believe that the nature of multivariate duration analysis offers specific challenges worth to address. [72] proposed a method based on random survival forest, which they claim is superior to any modification of the standard cox regression framework [110, 113, 97, 17], but also to alternative methods [70, 31]. However, both [72] and cited literature are prediction exercises using gene arrays, that is biological variables. The exercise presented here however aims also at using survival model for causal explanation applying economic theory to socio-economic variables, which, contrary to biological and physical ones, describe socio-economic phenomena prone to change abruptly even in the short run. Therefore, the process of variable selection needs both to explore how stable is the data generation process leading to the event and, possibly, operates a different selection of variable in each time period. This selection should also allow for the man-in-the-loop, that it should embed educated and theory-driven knowledge of the researcher.

They key issue in the framework propose is the use of data science algorithms to empower the research and not not as substitutes. With this aim in mind, the next section discuss the theoretical approach and present 4.2, graphical models as a general framework 4.2.1, for variable selection in a context which combines an automatic approach with theory-driven approach of the researchers, a Bayesian inference tool to estimate the stability of the relation of variables over time 4.2.2, the precise algorithm which operate the selection 4.2.3. Sections 4.4 and 4.5 perform an exercise of firms survival and highlight the difference between a traditional approach and the present one in the variable selection.

4.2 Survival analysis and data science

This paper takes up the challenge of introducing new emerging tools from data science with the aim of improving already established methods of analysis [145]. Specifically, we address the issue of variables selection, which, in the presence of many variables, creates a trade-offs between the need of minimizing the loss of information and the constraints of econometrics model which mandate specific requirements on esogeneity of the variables, degrees of freedom, and coherence with the theory.

Traditionally, variable choice in an econometrics exercise is theory driven and operated by the researcher based on both its educated guess and, in some cases, a process of trail-and-errors. While this process exploits all theoretical knowledge, it presents some drawbacks. If the variables' set is extremely large, the task can go beyond the cognitive ability of the researcher, who will opt for cognitive shortcuts [58]. Thus this process might be influenced by cognitive biases and possibly be subject to scientific malpractices such as the p-hacking behavior[26, 66]. Moreover, the process of selection can be opaque and usually motivated ex-post such in the case of reverse p-hacking and selective report [30]. On the contrary, an automated purely data-driven process is fast, transparent and free from biases, but it does not allow to leverage the information coming from theory, expertise, and literature: indeed in many cases, there is a clear theoretical expectation that a variable will influence an outcome and the exercise is either to test this hypothesis or infer its strength [112].

Along this line, [60] propose a general methodology to build few new variables out of many under theoretical constraints set by the researcher. Specifically, they show how information in many variables can be automatically reduced in a new one with supervised machine learning, but still obeying some rules set ex-ante which encode the expertise of the researcher as the man-in-the-loop.

In this paper, we maintain the idea that data science methods in the context of econometrics should empowered the researcher, rather than substitute her, but we overturn the [60]'s methodology. While they automatized a process with rules set by the researcher ex-ante, in the present exercise the data-driven process serves as a first step to simplify the space of possible choices and allow the researcher to operate both within her cognitive ability and with an accountable and reproducible method to select variables. Specifically, we introduce graphical models as a tool to compute and visualize the conditional dependency structure of a data-set. Graphical models allow the researcher to appreciate any variable of interested and its direct determinants. Thus, graphical models reduce the information to be elaborated by the the researcher, but at the same time, controlling for the dependency structure, allow her to select variables without omitting covariates with a mediating effects and incurring in estimating spurious relationships. The role of theory and expertise in the selection of the variable occurs only ex-post exposing with clarity the rationale behind the selection. Since the set of possible variables can still be very large, we add a further layer of automation, which occurs in a set of meaningful relations controlled by the-man-in-the-loop. In sum, the

process, graphically depicted as a flowchart in Figure 4.1, is described by the following steps:

- unsupervised production of a graphical model;
- researcher's analysis of the dependency structure of the graphical model and theory-driven transparent intervention removing or adding variables and subsequent transformation of the graphical model;
- stability analysis, to detect a possible model drift which would hinder the possibility of a panel analysis;
- automatic variables selection on the transformed graphical model for each year in case of low stability;
- econometric analysis.

This procedure allows for a transparent selection of the variables, by blending automatic algorithms and theory, and checks for the stability overtime of the relation. The next section briefly recalls the main element of graphical models, the analysis of stability, and variable selection as introduced in the previous chapters of this thesis.

4.2.1 Graphical models

Graphical models are a method to display the conditional independence relationships between the variables through a network or a graph. A Graph is a mathematical object $G(V, E)$, where V is a finite set of nodes with direct correspondence with the variables and $E \subset V \times V$, is a subset of ordered couples of V [92]. Graphical Models used in this paper belong to *classes of multivariate distributions*, whose conditional independence properties are encoded by a graph in the following way: the random variables have a direct representation with the nodes of the graph, while the absence of the edges between nodes represents conditional independence between the corresponding variables. We define a graphical models as an undirected graph $G = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V} = \{v_1, ..., v_p\}$ is the of vertices and $\mathbf{E}$ is the set of edges. Furthermore, an edge $e = (u, v) \in \mathbf{E}$ indicates that the variables associated to u and v are not conditionally independent given the rest of the dataset. We restrict the analysis to undirected decomposable graphs, for which two non-adjacent nodes are separated by a set of (at most) size one (tree). A *tree* is a special undirected graph, that does not allow cycles. If a dataset is represented by a collection of *tree*, we call this graph a *forest*. This simplification allows us to calculate the maximum spanning tree, that is the tree that reduces information redundancy, with the mean of the *Chow-Liu Algorithm* [29]. This algorithm was developed to approximate optimally n-dimensional discrete probability distribution by a product of second-order distribution or the distribution of the first-order tree. In particular, we adopt the extension of the *Chow-Liu Algorithms* by [42], which allows also for continuous variables[1].

[1]For further details and a review on graphical models see Chapter 1

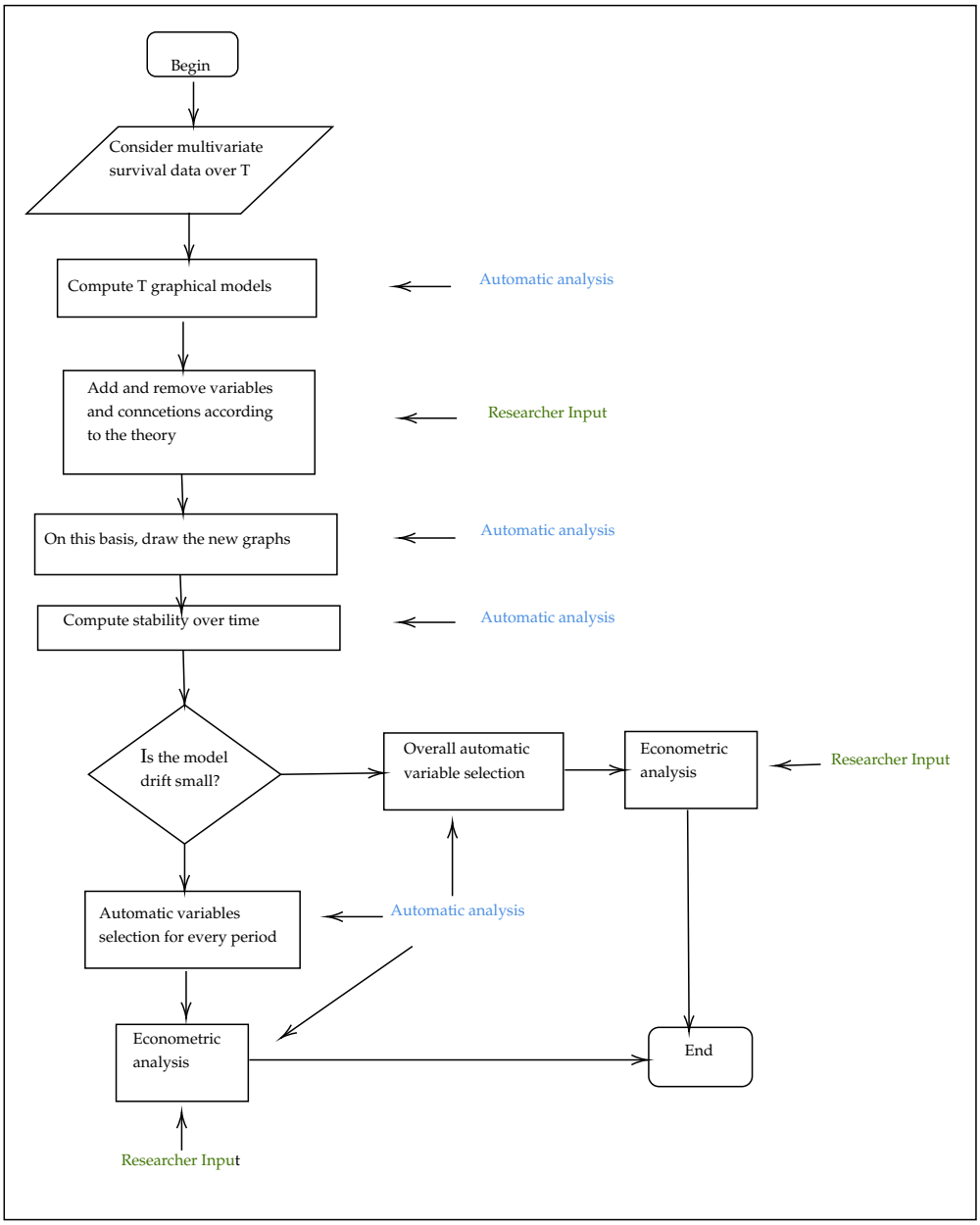


Figure 4.1: Flowchart: combining automatic analysis with research expertise

4.2.2 Graphical model and stability

In the context of time dependent variables, such as the case for survival analysis, the (in)dependence relationship among variables, as encoded in a graph does not remain necessarily stable. Chapter 3 of this thesis is devoted to the analysis of the drift, that is the estimation of the stability of connection of a graph over time.

First, given a collection of datasets composed by p variables, T different time of observation and n_t observations that correspond to the number of observation we compute a random variable *stability*, base on the change of connection among variables

described by a *Transition matrix* (TM). Starting from the adjacency matrix (AM), obtained from the graphical models for each year, *Transition matrix process* is a function that maps any possible change of state between two variables, that is a change of state between AM_0 to AM_t with $t = 1, 2, ..., T$. Specifically, the generic element $w_{i,j;t}$ of TM_t, with dimension $V \times V$, registers the evolution of the conditional dependency of any couple of nodes V_i and V_j in T period. Specifically, the TM takes the following form:

$$TM_t = \sum_{t=1}^{T} 2^{(T-t)} AM_t \tag{4.1}$$

Chapter 3 shows how we use the $w_i, j; t$ to identify the distribution of stable connections between two variables and the describe it as a Bernoulli distribution:

$$Y_i|\theta_i \overset{ind}{\sim} Bern(\theta_i), \quad i = 1, .., n$$

where Y is the long form vector of all possible connection over the T years and takes value of 1 if the connection is stable up to that period and 0, otherwise. On this basis, it is possible to implement a logistic regression model that is the odds of Y as a linear function of both the history of past connections encoded in W and T. Since W has 2^t levels, we regress $2^t - 1$ dummy variable and keep $W = 0$ (stable absence of connections) as the reference category, it that way the logistic regression is:

$$\log\left(\frac{\theta_i}{1-\theta_i}\right) = \beta_0 + \beta_1 \times T + \sum_{j}^{2^t-1} \beta_j \mathbf{w}_{j,i} \tag{4.2}$$

While in the section 3.4, we discussed the meaning of the coefficient, here we focus on the intercept of this model β_0, which, by construction can be is baseline risk for *Stability*. We can express the Eq.4.2 in term of probability of stability of θ_i:

$$\theta_i = \frac{\exp\{\beta_0 + \beta_1 \times T + \sum_{j}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}}{1 + \exp\{\beta_0 + \beta_1 \times T + \sum_{j}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}} \tag{4.3}$$

From the Eq. 4.3 we can define the likelihood for the sequence of Y_i over dataset of n subjects is then

$$p(\mathbf{D}|\beta_0, \beta_1, \boldsymbol{\beta}_j) = \prod_{i=1}^{n} \left[\left(\frac{\exp\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}}{1 + \exp\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}} \right)^{y_i} \left(1 - \frac{\exp\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}}{1 + \exp\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}} \right)^{(1-y_i)} \right] \tag{4.4}$$

where **D** is the dataset composed by the corresponding dummy variables by the level of W for each Y_i and the time of the transition T. The set of unknown parameters consists of $\beta_0, \beta_T, ..., \beta_{2^T-2}$. Empirically, we use the most common priors for logistic regression parameters, which are of the form:

$$\beta_j \sim N(\mu_j, \sigma_j^2) \tag{4.5}$$

with μ is zero with σ large enough to considered as non-informative in the range from $\sigma = 10$ to $\sigma = 100$. The posterior distribution of $\boldsymbol{\beta}_j$ is extrapolated by combining likelihood Eq. 4.4 and prior 4.5:

$$\begin{aligned} p(\beta_0, \beta_1, \boldsymbol{\beta}_j | \mathbf{D}, \sigma_j, \mu_j) = \prod_{i=1}^{n} \Bigg[\left(\frac{\exp\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}}{1 + \exp\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}} \right)^{y_i} \\ \left(1 - \frac{\exp\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}}{1 + \exp\{\beta_0 + \beta_1 \times T + \sum_{j=1}^{2^t-1} \beta_j \mathbf{w}_{j,i}\}} \right)^{(1-y_i)} \Bigg] \\ \times \prod_{j=0}^{p} \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left\{ -\frac{1}{2} \left(\frac{\beta_j - \mu_j}{\sigma_j} \right)^2 \right\} \end{aligned} \tag{4.6}$$

4.2.3 Graphical models and variable selection

We briefly recall here the main result of THE Chapter 2, in which we use the decomposable graph property for the development of of an algorithm of variable selection. This algorithm belongs to belongs to *Sequential Forward Search* [114]. In this case, we fit the algorithm for a binary outcome. The strategy to identify the appropriate number of regressors is the following: at **Step 0** the algorithm produces a rank of the importance for all variables, then the from **Step 1** to **Step 4**, it maximizes the relevance and at **Step 5** it and minimizes the redundancy of information[2]. Specifically, the algorithm operates according the following steps:

- **Step 0**: the algorithm finds the best spanning tree or forest, and call this model $\mathcal{M}_0$;
- **Step 1**: the algorithm identifies all path steps w_i, starting form the variable of interest identified by the researcher;
- **Step 2**: the algorithm divides the dataset in training set(75%) and validation set (25%)
- **Step 3**: the algorithm for $i = 1, ..., N$:

    (a) fits the model with k variables present in path step w_i

    (b) computes the predicted probability of the outcome of interest in the test set;

[2] details in Chapter 2 of this thesis

(b) measures the overall performance with the area under the ROC curve (AUC) [73] .

- **Step 4**: Select the best among N models (largest AUC), and call it $\mathcal{M}_w$
- **Step 5**: Fit the model $\mathcal{M}_w$ and select only the significant variables. Call this model $\mathcal{M}_f$

It is simple to note that $\mathcal{M}_0 \subset \mathcal{M}_w \subseteq \mathcal{M}_f$. The output of the model $\mathcal{M}_f$ can be written as:

$$y_i = f(\mathbf{w}_i) + \epsilon_i$$

where $\mathbf{w}_i$ represent a vector of the variables selected by the algorithm. In order to control the temporal dependence, we have included in the models the probability to fail of the previous year. In that way, we can see if the performances of the previous year can affect the probability to fail in the year of observation.

4.3 Data

For this analysis we use the AIDA (Analisi Informatizzata delle Aziende) database provided by the Bureau van Dijk. This database contains comprehensive information on all Italian firms required to file account. Each firm is described by a large number of variables in the following categories: identification codes and vital statistics; activities and commodities sector; legal and commercial information; index, share accounting and financial data; shareholders, managers, company participation. From this database we consider variables with lowest percentage of missing data and that describe all macro categories. Specifically, we observe all firm funded in 2009 and we observe them along a time span of 10 years. Details of the percentage of the missing variables are recorded in Table 4.3. Since there is still a percentage of missing variables, we use *Random Forest Missing Algorithm* as data imputation strategy. This method has some desirable properties, since it is able to handle mixed types of missing data. Furthermore, it is adaptive to interactions and non-linearity and it has the potential to scale to big data settings [137]. We implemented the *Random Forest Missing Algorithm* with the support of Open Computing Cluster for Advanced data Manipulation (OCCAM) at the University of Turin [8, 9].

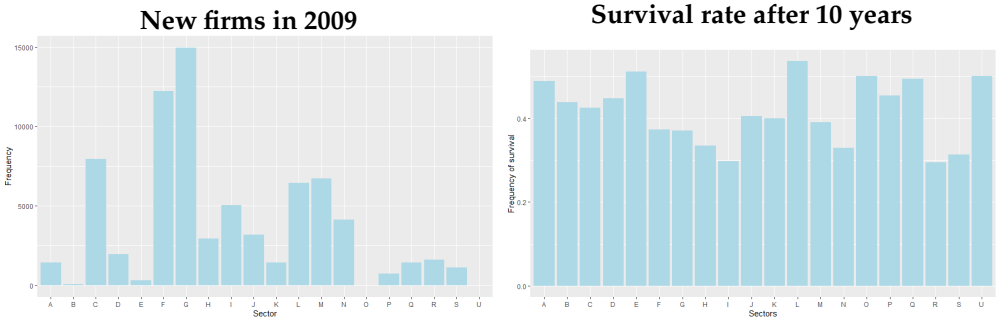
4.4 Traditional Survival Analysis

Traditional approach to survival analysis in such a context relies on an extensive literature which overtime highlight both empirically and theoretically the main determinants of survival [59, 118]. It is not the purpose of the paper to survey the broad literature, but we recall here the six main elements which emerged in the last decades as the fundamental ones together with few selected reference, specifically:

- Age and size [118, 13, 57];

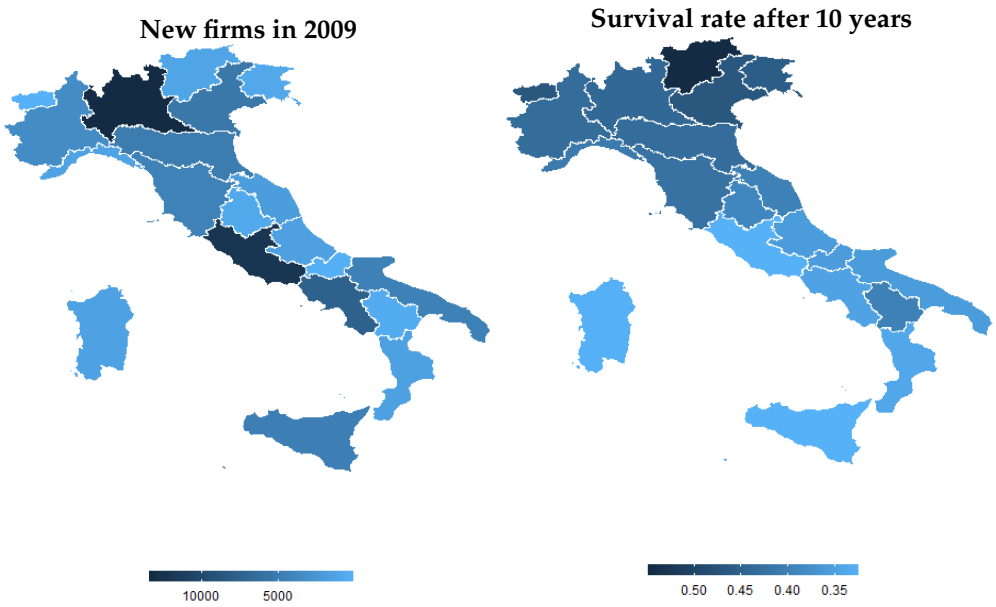
- Sector or Industry of belonging; [101, 81, 57]
- Geographical localization; [4, 134, 133]
- Profitability; [37]
- Liquidity constraints; [69, 107]
- Innovativeness and Entrepreneurship. [60, 71, 123]

The traditional approach of variables selection consists in deriving from both theory and literature the most promising hypotheses to be tested. Insofar, among the variables in the dataset, we selected *Region*, *Sector*, *Total from sales* and *Production cost* as a proxy for profitability, *Index liquidity* to capture liquidity constraints, *Employees* and *Sales* for the size, and for the innovativeness *Innovative Startups*, that is whether a firm is registered after 2012 in the register of Italian innovative start-ups. The variable *Sector* is *Nace Rev.2* and Fig. 4.2 shows the distribution across sectors and the survival rate after 10 years, while the maps in Fig. 4.3 displays the distribution of the observations and they survival across Italian regions. More details are reported in Appendices in the Tables 4.4 and 4.5.



(a) Frequency **(b)** Survival Rate

Figure 4.2: Histograms of the Sectors in the *start-ups*



(a) Map of the Frequency

(b) Map of the Survival Rate

Figure 4.3: Maps of the *start-ups*

We can thus consider the set of selected variables and define $X_i = \{X_{i,1}, ..., X_{i,p}\}$ as the realized values of the covariates for firm i and Y_i the correspondence status, of survival or not. We adopt semi-parametric hazard models, that are specifically designed to examine the duration phenomena to ascertain survival determinants by explaining the time period between a firm´s start-up and its cessation of economic activity. The most commonly used models for survival data describe the transition rate from one state to another, wherein this case the transition is represented by the death of the firm [91]. These models belong to Poisson regression, in particular, the *Cox* proportional hazard models used in this analysis: These models belong to Poisson regression, in particular, the *Cox* proportional hazard models used in this analysis:

$$\lambda(t|\mathbf{X}_i) = \lambda_0(t) \exp \sum_{j=1}^{p} \beta_j \mathbf{X}_{i,j} \tag{4.7}$$

It is simple to note that some variables that are time variant. Following the standard approach to survival analysis, we consider the time dimension according to:

$$\lambda(t|\mathbf{X}(t)_i) = \lambda_0(t) \exp \sum_{j=1}^{p} \beta_j \mathbf{X}(t)_{i,j} \tag{4.8}$$

where the covariate $X(t)$ is the value of time-varying covariate for the $i_t h$ subject at time t, with $t = 1, ..., T$. The partial likelihood, in general, is as follows:

$$L(\beta) = \prod_{t=1}^{T} \left[\frac{\lambda(\mathbf{Y}_i|\mathbf{X}_i(t))}{\sum_{i \in R_i(t)} \lambda(\mathbf{Y}_i|\mathbf{X}_i(t))} \right] \tag{4.9}$$

where the expression $i \in R_i$ indicates that the sum is taken over all subject in the risk set R_i at time t. Figure 4.4 shows the survival cur for the firms born in 2009, while the figure 4.5 shows the survival curs with the stratification for *Macro-Region*. Figure 4.11 shows the survival cur for the *Innovativave* stratification.

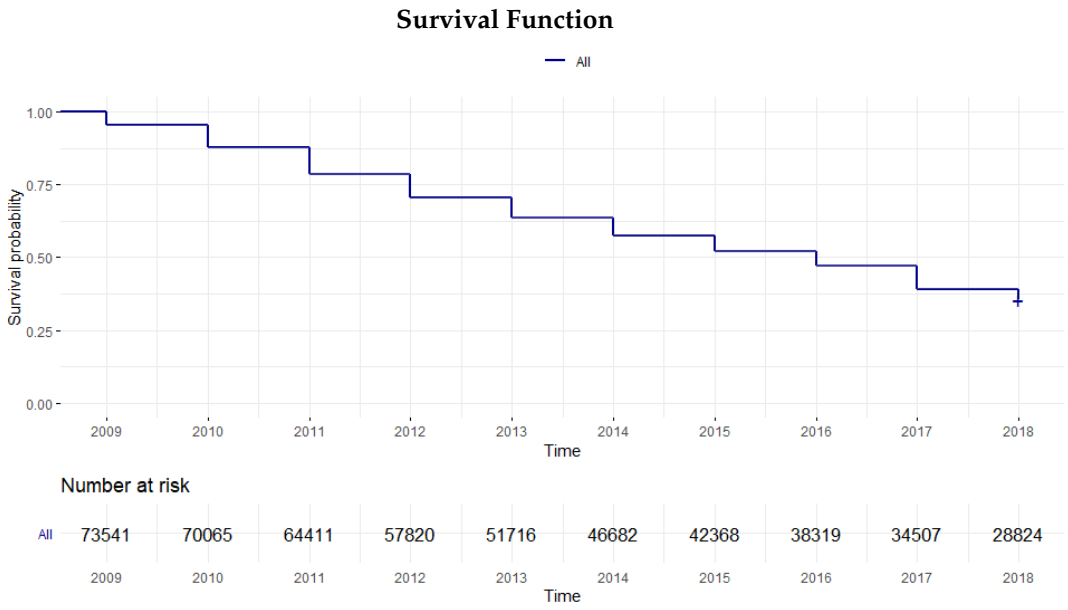


Figure 4.4: Survival function for the firms born in 2009 and table risk

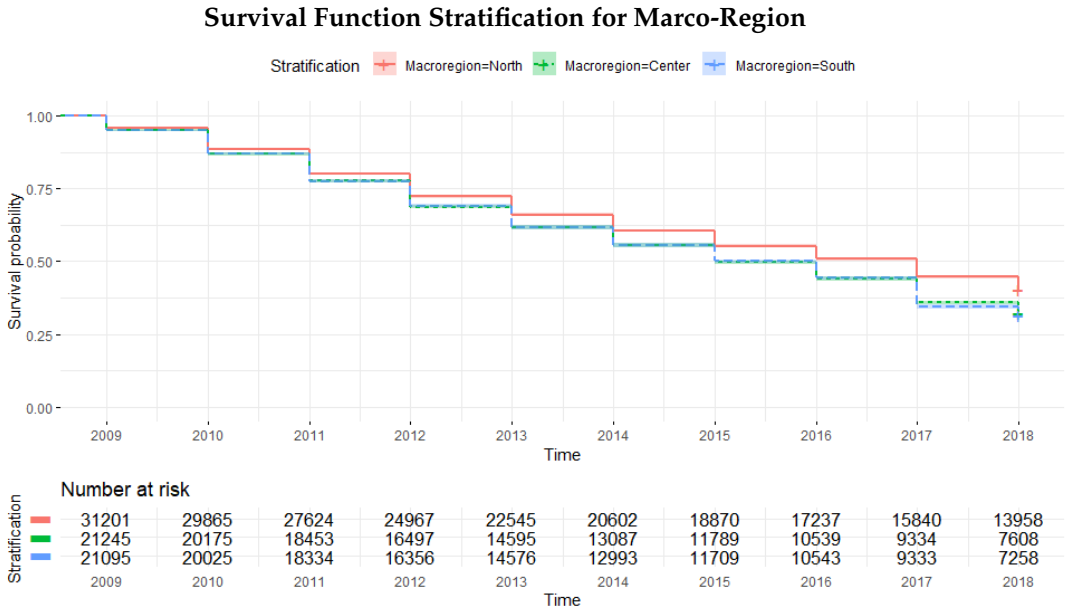


Figure 4.5: Survival Function Stratification for Marco-Region and table risk

Figure 4.5 corroborates the role play by the regions, in Italy along the social-economic divide between the North and the rest of Italy [3]. Table 4.6 shows the result of the mul-

Macro-Region	**N**	**Observed**	**Expected**	$\frac{(O-E)^2}{E}$
North	31201	18634	241	241
Center	21245	14366	83	83
south	21095	14409	107	107
χ^2 = 480, on 2 degrees of freedom, *p-value*$\leq 2e-16$				

Table 4.1: log-rank test for Macro-Region

tivariate *Cox* regression, in which sector *G* (wholesale and retail trade; repair of motor vehicles and motorcycles) is the reference level for the variable *Sector*, while for the variable *Region* the the reference is *Lombardia*. The results consistent with the literature and the selected variable are mainly significant. These result assume both a stable relation among variables over a time span of ten years and stable coefficients as well. In the next section we apply the theoretical framework presented in Section 4.2 to the same data and present an alternative view in which the variable selection is computer-aided and stability is not considered for granted.

4.5 Graphical model and survival

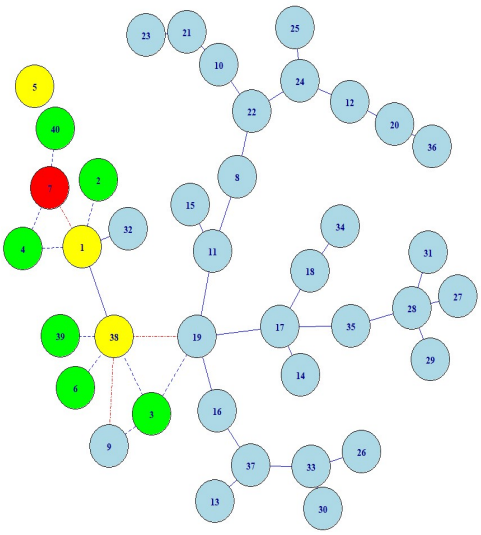
Section 4.2.1 introduced graphical models a method to map he conditional dependence structure of a dataset, to evaluate its stability overtime, and to select the variable to include in a regression, accordingly. In this section we apply the method to the data at hand and show whether this approach produce qualitatively different results.

4.5.1 A graphical model of Italian Start-ups

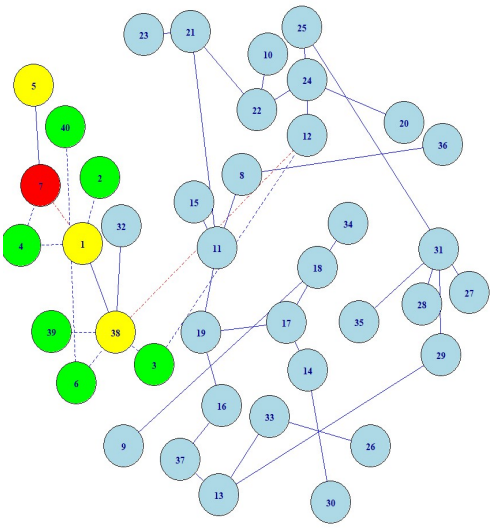
Figure 4.6 presents the graphical model for ten years: continuous variables are light blue, the discrete ones green, region and sector are yellow, and survival is red.The label of the nodes are reported in table 4.2.

[3]The table 4.1 reports the *log-rank* test, that is the most widely used method of comparing different survival curves. The *log-rank* is approximately distributed as a χ^2 test statistic and is a non-parametric test, which makes no assumptions about the survival distributions

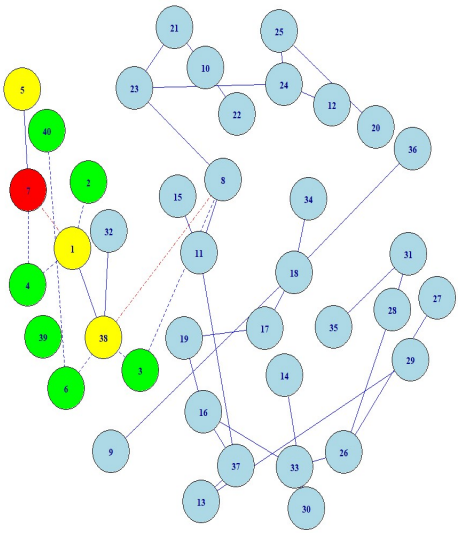
Spanning Tree 2009



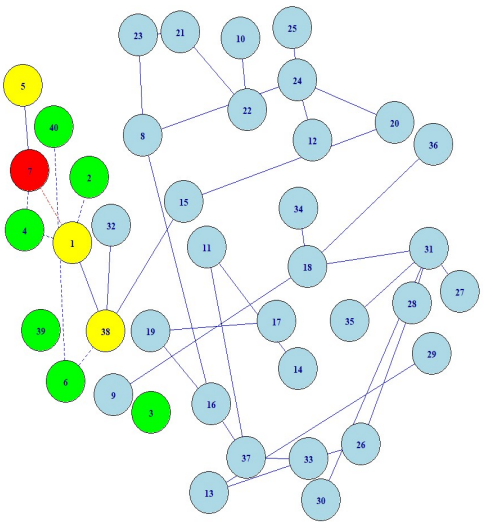
Spanning Tree 2010



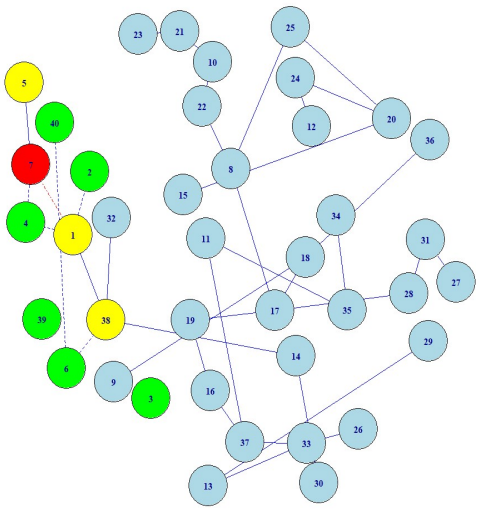
Spanning Tree 2011



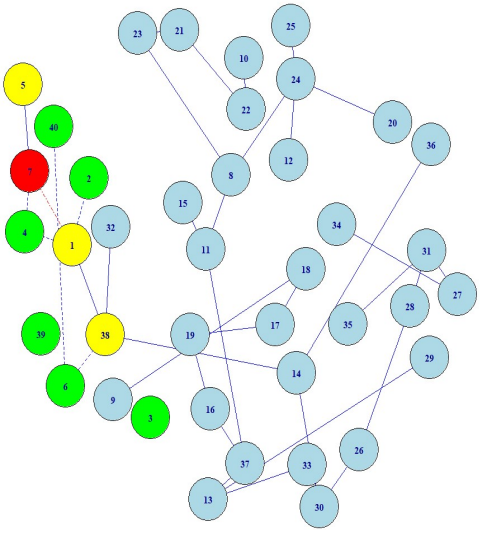
Spanning Tree 2012



Spanning Tree 2013



Spanning Tree 2014



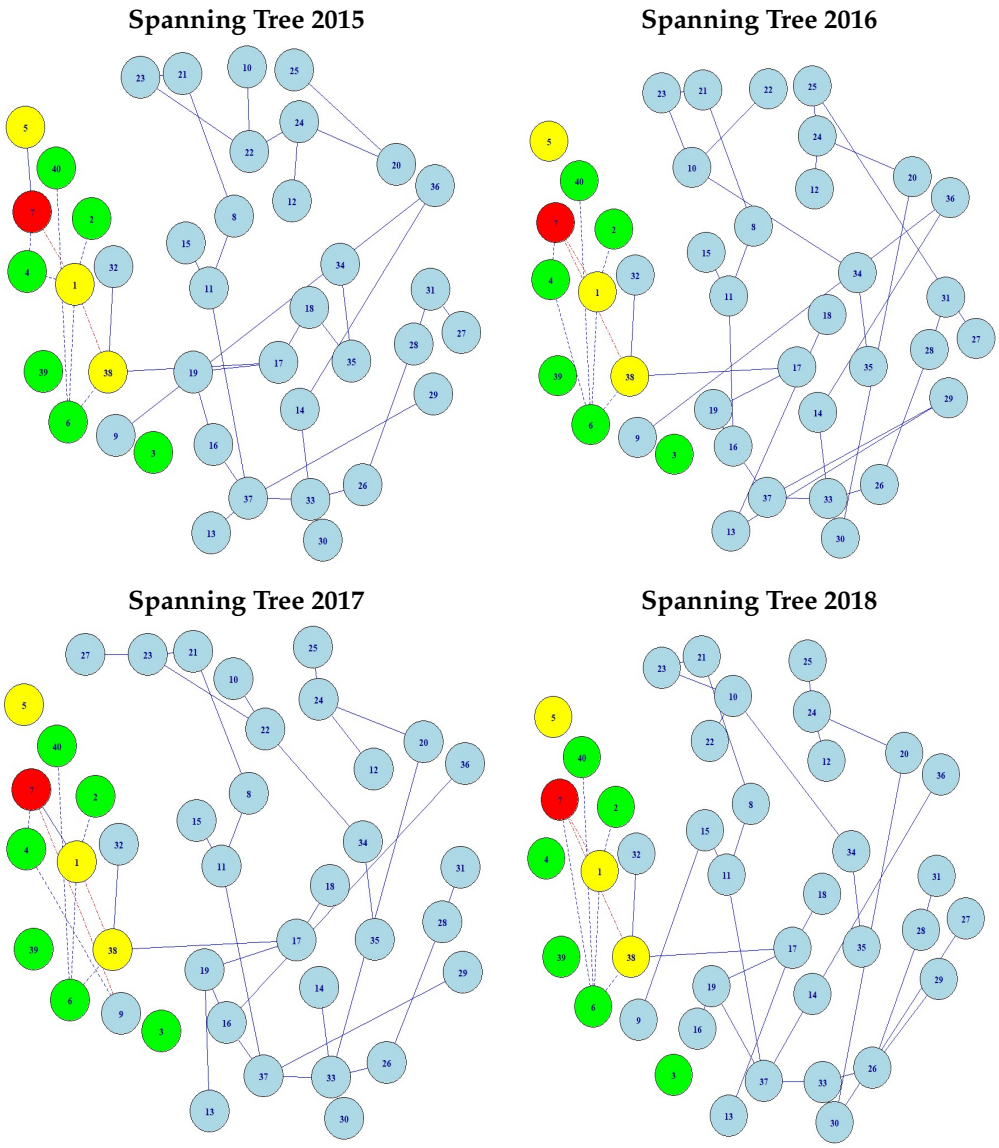


Figure 4.6: Graphical Models over ten years

Figure 4.6 can help the researcher in identifying the variable to include or not in the selection process for the survival analysis with a high level of accountability. This step, which introduces "the-man-in-the-loop" of the automatic process serves to include the researcher's knowledge with scientific justifications.

In the case at hand we decide the exclude 6 variables, which two precise characteristics: they both generate noises in the data and they are not theoretical relevant. Specifically, the following motivations explain the reason of exclusion:

- *Province* (2) direct dependence with the variable *Region*, presence of many levels.
- *Legal form* (3) presence of many levels, among which many are not informative. Moreover the variable is rarely.

- *Legal status* (4) Presence of many non informative levels, not mentioned in the literature, data non reliable since they do derive from administrative process.
- *Artisan Companies* (6): zero-inflated, non informative, not mentioned in the literature.
- *Constitution quarter* (40): not informative since in Italy, the timing of constitution relates more with administrative deadlines.
- *Sae description* (39): many non-informative levels, not mentioned in the literature.

Based on this new set of variables the selection algorithm will identify each years a different number of regressor (k) each year, hindering the possibility of a standard duration analysis which exploits the panel structure. We estimate the each year the probability of survival with a logit regression:

$$\log\left(\frac{\rho_{i,t}}{1-\rho_{i,t}}\right) = \beta_0 + \sum_{j=1}^{k} \beta_j x_{t,j} \quad \forall \quad t \tag{4.10}$$

where on the left hand side, we have the odd ratio of surviving explained by the set of selected variable X_t. Unfortunately, in this way we loose the information of the variable in this previous and we cannot safely assumed that it is uncorrelated with with the present probability of surviving. Indeed the likelihood of surviving depends very much also on the past history of the firms. In order not to incurred in the omitted variable bias, we add in the set probability of surviving in $t-1$, in this way we test the impact of the variable in t, controlling for past history of the firm, which in this case is unobservable. Thus also add a variable *Predicted probability*: $p_{i,t-1}$.

We prune all the connection relating to this variables and compute the new edges represented as red connections in Fig. 4.6. The resulting graphs highlight that variable *Survival* node (7), which is the variable of interest belongs to the main component in every year. Thus, as expected, this variables exhibits multiple relationships with the remaining variables in the data-sets. The analyses were performed using the R library *gRapHD* which we have made available to the R community via the *CRAN* repository ([2]).

4.5.2 Stability

To evaluate how changes the relationship between the variables over the observation time, we introduce the *stability*, a measure developed to evaluate the *drift* [4]. Figure 4.7 shows the result of the logistic regression for the *Stability* of the graphs in figure 4.6.

[4] for more details we suggest the Chapter 3

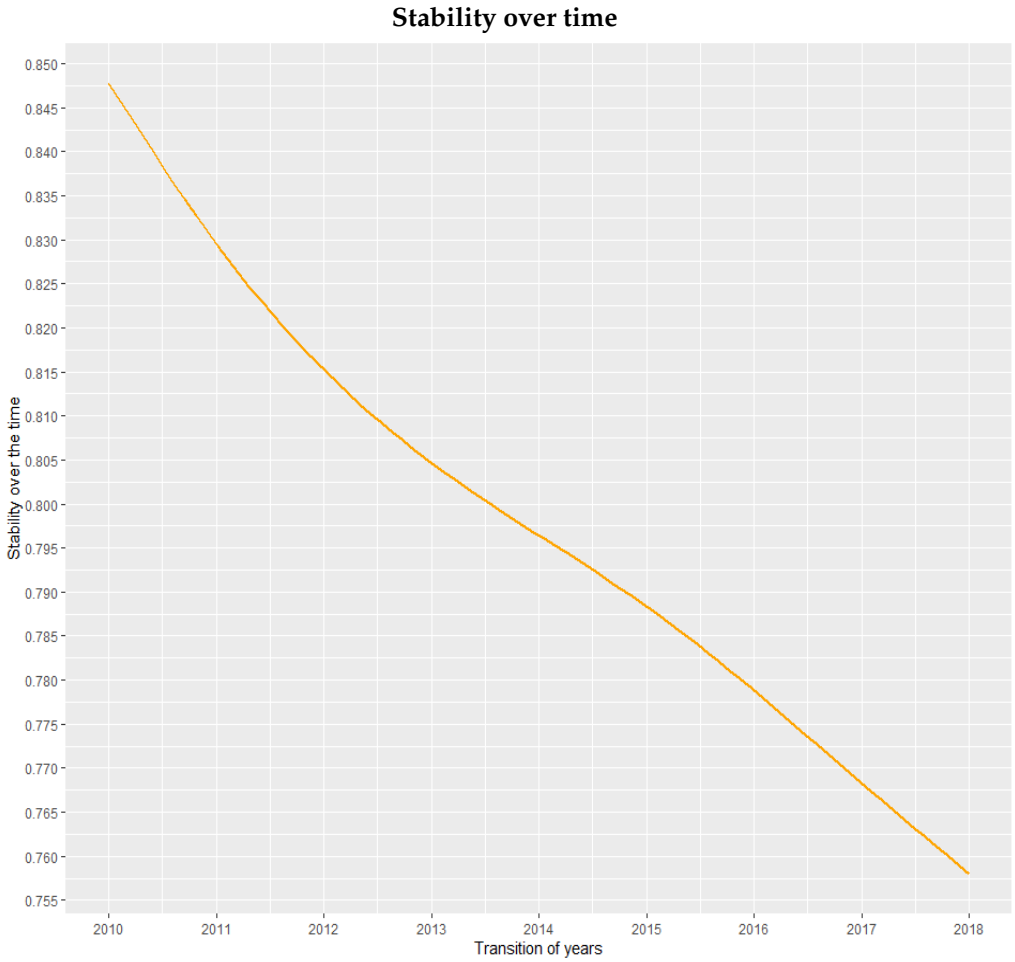


Figure 4.7: Stability

As we can see from the figure 4.7, the *Stability* constantly decreasing during the years, for this reason , it is important to identify a model that is dynamic over the time. Details of the coefficient are recorded in the table 4.7, while figure 4.12 shows the result of the Eq. 4.6.

4.5.3 Variables Selection

In this section we present the results of the variables selection carried out through the algorithm presented in the section 4.2.3. The results in evidence in the figures 4.8, 4.9 and 4.10 show an evolution of the relevant variables over the time and they are compared with the coefficient of the traditional survival exercise. Gray color stays for non significant coefficient or not selected for the analysis. Traditional survival analysis, highlight the role firm innovativeness, size, cost of production, revenues and as usually predicted in the literature, region and sectors. The alternative approach does not dismiss these variables, but overall it provides a richer analysis which changes remarkably year by year. At the general level, survival in 2010 and 2017 does not seem to be

captured by the current variables as if in these years other events were responsible for firms' exit.

More in details, innovativeness [5] is still recognized as an important variable, but only in few specific years. For categorical sectors and regions, the new method seems to explain the mild significance of the traditional supervisor as a mean between years with a strong impact of regions and sectors and other, with a minimum or absent one.

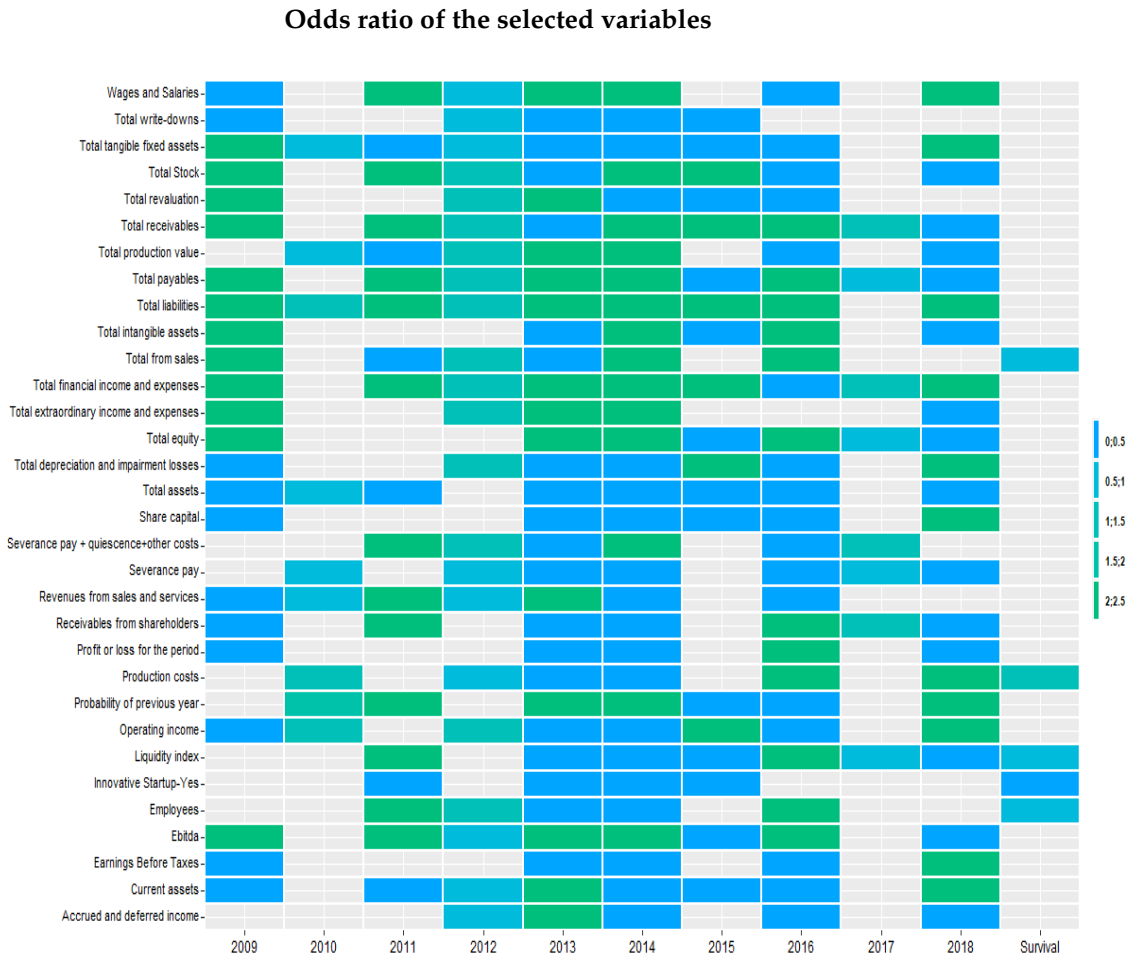


Figure 4.8: Impact of significant continuous variables

[5] the survival curve for innovativeness firms is reported in the Figure 4.11

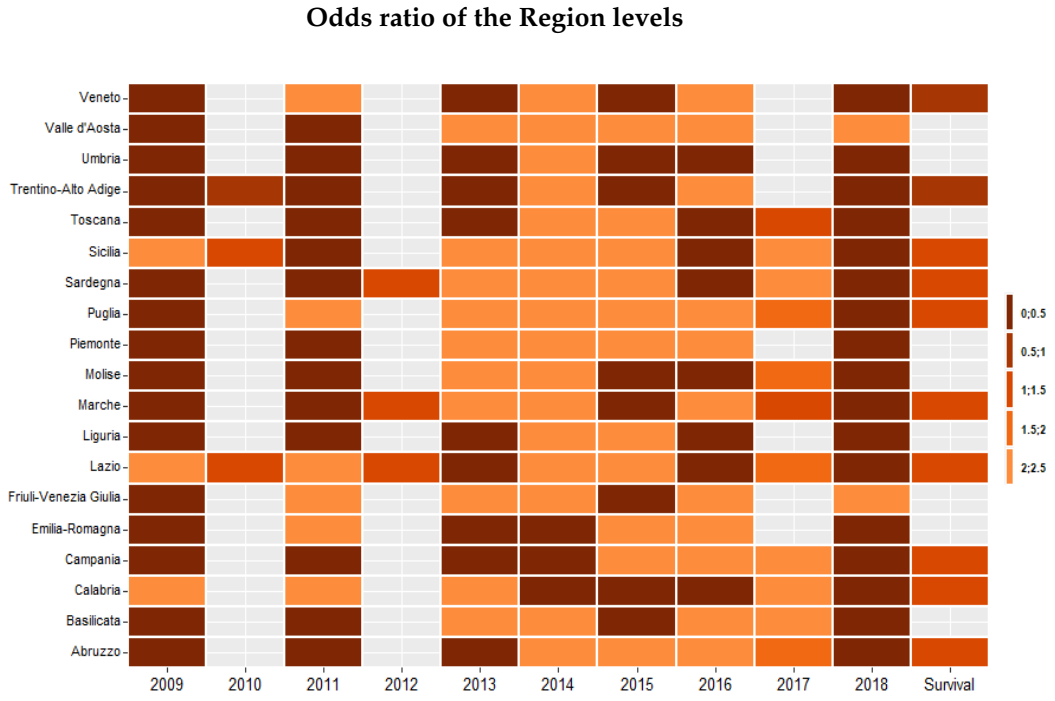


Figure 4.9: Impact of significant *Regions*
Reference level: region *Lombardia*

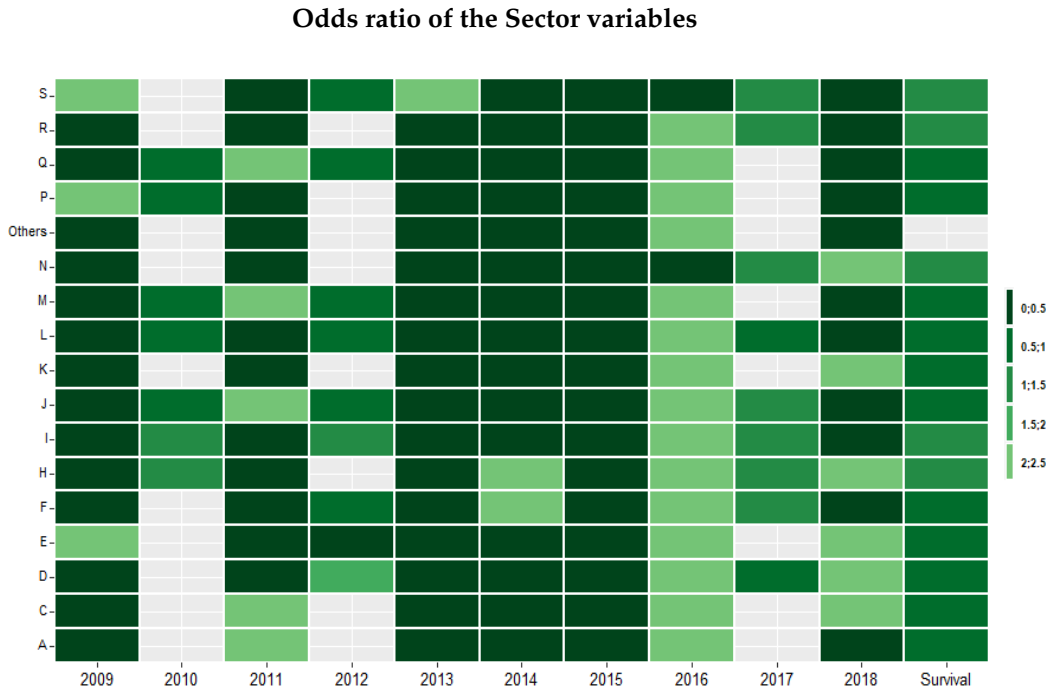


Figure 4.10: Impact of significant *Sector*
Reference Level: sector *G*

4.6 Conclusions

This paper proposes a methodology to address two issues in survival models. First, the increasing availability of data with a large number of variables makes the process of variable selection both cumbersome and, in some cases, opaque. Secondly, the hidden context not captured by the observable data of a socio-economic phenomenon might change over time and it is not possible to assume that the generative process of the data remains constant. We employ graphical models as a tool for both empowering the research in the process of variable selection and testing the stability over time of the underlying dependency structure of the variable. When applying this method and comparing it with a traditional survival exercise, results are striking. While the traditional methodology highlights the role of some variables, as for instance firms' innovativeness, the approach suggested in this paper shows that this effect explains survival in few selected year only. Moreover, this process of selection is fully accountable and remove any risk of selective reporting and p-hacking. This paper contribute to the ongoing attempt in the literature of combining data science tools with traditional econometrics, in order to enhance a researcher's possibilities, without substituting its role.

4.7 Appendix

In this section are reported the details of missing data 4.3, the frequencies of the Regions 4.4 and of the Sectors 4.5. It is possible also to find the coefficients of the Cox Regression. Furthermore in the table 4.7 are reported the coefficients values of the Bayesian Logist Regression for the *Stability*, where each coefficients of the factor **w** are negative respect to $w = 0$, except the coefficient $w = 1023$ that identify the stability of the connection of the edges at time $t = 10$. Figure 4.12 shows the result of the Equation 4.6, in particular we can see the distribution of the posterior and the evolution of the edges over the time. We can see for each years N points. Furthermore, Figure 4.12 shows also as the edges between the variables change over time. Indeed, the ordinate indicates the sorted levels of the variable $w_{i,j}$.

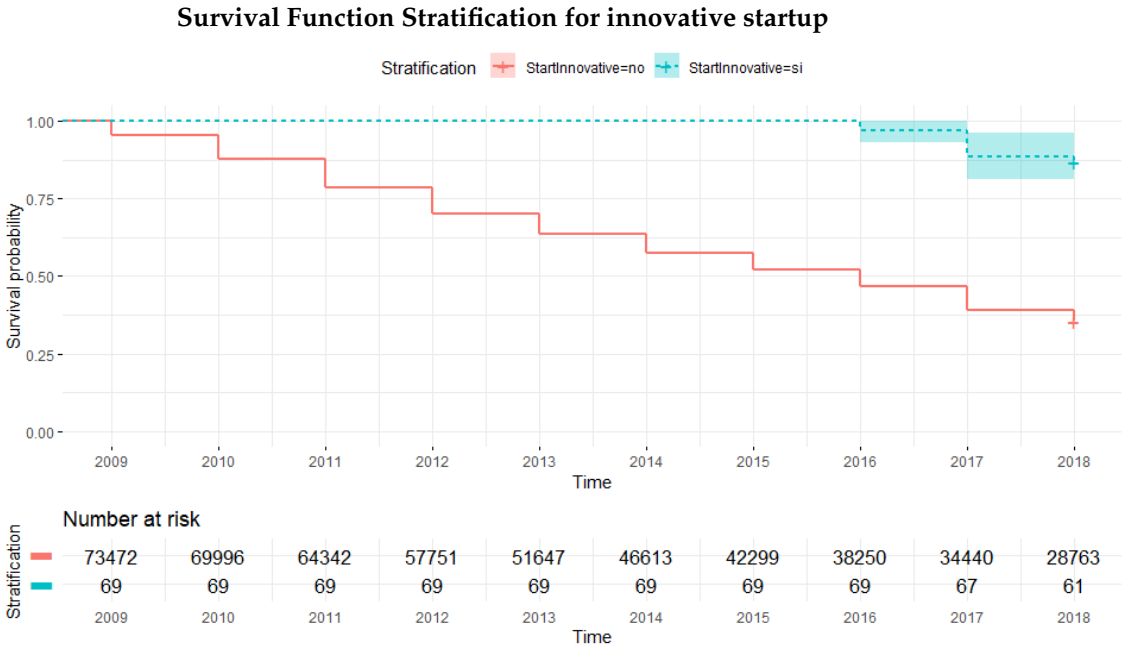


Figure 4.11: Survival Function Stratification for innovative startup and table risk

Name of Variables	Label Node	Type of variable
Region	1	Discrete
Province	2	Discrete
Legal form	3	Discrete
Legal State	4	Discrete
Innovative Startup	5	Dichotomous
Artisan Companies	6	Dichotomous
Death	7	Dichotomous
Total receivables	8	Continue
Receivables from shareholders	9	Continue
Total from sales	10	Continue
Current assets	11	Continue
Employees	12	Continue
Total tangible fixed assets	13	Continue
Total intangible assets	14	Continue
Total Stock	15	Continue
Total assets	16	Continue
Total equity	17	Continue
Share capital	18	Continue
Total liabilities	19	Continue
Severance pay	20	Continue
Total production value	21	Continue
Revenues from sales and services	22	Continue
Production costs	23	Continue
Wages and Salaries	24	Continue
Severance pay + quiescence+other costs	25	Continue
Operating income	26	Continue
Total extraordinary income and expenses	27	Continue
Earnings Before Taxes	28	Continue
Total financial income and expenses	29	Continue
Total depreciation and impairment losses	30	Continue
Profit or loss for the period	31	Continue
Liquidity index	32	Continue
Ebitda	33	Continue
Total revaluation	34	Continue
Total write-downs	35	Continue
Accrued and deferred income	36	Continue
Total payables	37	Continue
Sector	38	Discrete
Sae description	39	Discrete
Constitution quarter	40	Discrete

Table 4.2: Name of the variables

Label Node	Percentage of missing data									
	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
1	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
2	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
3	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
4	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
5	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
6	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
7	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
8	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
9	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
10	20%	5%	4%	4%	4%	3%	3%	2%	1%	0%
11	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
12	28%	26%	12%	8%	7%	4%	4%	4%	3%	3%
13	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
14	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
15	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
16	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
17	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
18	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
19	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
20	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
21	20%	5%	4%	4%	4%	3%	3%	2%	1%	0%
22	20%	5%	4%	4%	4%	3%	3%	2%	1%	0%
23	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
24	20%	5%	4%	4%	4%	3%	3%	2%	1%	0%
25	20%	5%	4%	4%	4%	3%	3%	2%	1%	0%
26	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
27	20%	5%	4%	4%	4%	3%	3%	2%	1%	0%
28	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
29	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
30	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
31	20%	5%	4%	4%	4%	3%	3%	2%	1%	0%
32	27%	10%	9%	9%	9%	8%	8%	7%	7%	6%
33	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
34	45%	30%	24%	17%	9%	3%	2%	2%	1%	0%
35	20%	5%	4%	4%	4%	3%	3%	2%	1%	0%
36	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
37	17%	4%	4%	4%	4%	3%	3%	2%	1%	0%
38	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
39	1%	1%	1%	0%	0%	0%	0%	0%	0%	0%
40	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table 4.3: Percent of missing for year

Region	**Number of fimrs born in 2009**
Lombardia	13430
Veneto	5453
Emilia-Romagna	5079
Piemonte	3552
Liguria	1408
Trentino-Alto Adige	1204
Friuli-Venezia Giulia	943
Valle D'Aosta	132
North	**31201**
Lazio	12217
Toscana	4435
Marche	1975
Abruzzo	1631
Umbria	987
Center	**21245**
Campania	7645
Sicilia	4766
Puglia	4580
Calabria	1614
Sardegna	1583
Basilicata	590
Molise	317
South	**21095**

Table 4.4: Distribution of the firms for Region

Label	Frequency	Description
A	1426	agriculture, forestryand fishing
B	64	mining and quarrying
C	7933	manufacturing
D	1936	electricity, gas, steam and air conditioning supply
E	317	water supply;sewerage, waste management and remediation activities
F	12214	construction
G	14955	wholesale and retail trade; repair of motor vehicles and motorcycles
H	2923	transportation and storage
I	5053	accommodation and food service activities
J	3192	information and communication
K	1433	financial and insurance activities
L	6422	real estate activities
M	6703	professional, scientific and technical activities
N	4117	administrative and support service activities
O	2	public administration and defence; compulsory social security
P	716	education
Q	1428	human health and social work activities
R	1588	arts, entertainment and recreation
S	1117	other service activities
U	2	activities of extraterritorial organizations and bodies

Table 4.5: Frequency of the sector

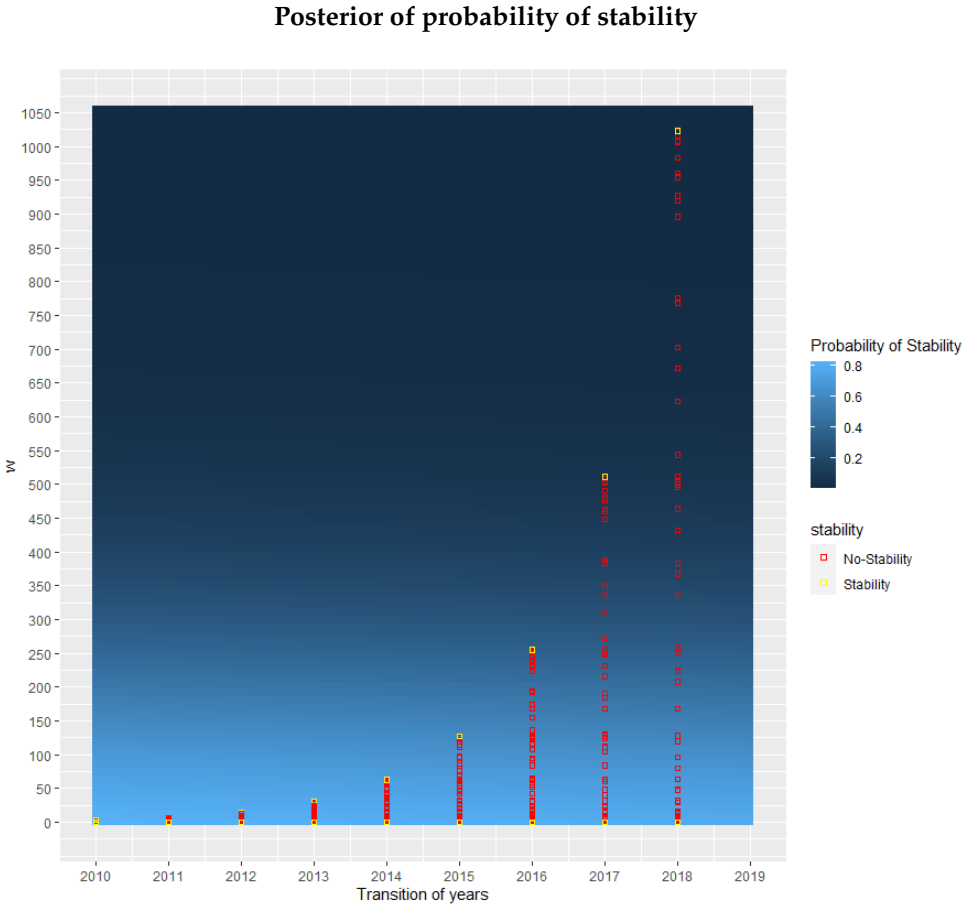


Figure 4.12: Evolution of the connections

Variables	Coefficient	exp(Coef)	se(Coef)	robust se	z	Pr(>|z|)	Signif.
Region Abruzzo	1.18E-01	1.12E+00	3.23E-02	3.26E-02	3.609	0.000307	***
Region Basilicata	-8.21E-03	9.92E-01	5.29E-02	5.15E-02	-0.159	0.873346	
Region Calabria	1.26E-01	1.13E+00	3.23E-02	3.30E-02	3.809	0.00014	***
Region Campania	1.37E-01	1.15E+00	1.79E-02	1.86E-02	7.357	1.89E-13	***
Region Emilia-Romagna	4.09E-03	1.00E+00	2.11E-02	2.13E-02	0.192	0.847482	
Region Friuli-Venezia Giulia	-6.11E-02	9.41E-01	4.39E-02	4.39E-02	-1.393	0.163483	
Region Lazio	2.13E-01	1.24E+00	1.56E-02	1.61E-02	13.209	2E-16	***
Region Liguria	5.57E-02	1.06E+00	3.53E-02	3.61E-02	1.543	0.122827	
Region Marche	6.59E-02	1.07E+00	3.02E-02	3.07E-02	2.144	0.032037	*
Region Molise	8.57E-02	1.09E+00	6.96E-02	6.76E-02	1.268	0.204727	
Region Piemonte	1.46E-02	1.01E+00	2.42E-02	2.45E-02	0.597	0.550658	
Region Puglia	1.28E-01	1.14E+00	2.13E-02	2.22E-02	5.78	7.46E-09	***
Region Sardegna	1.95E-01	1.22E+00	3.21E-02	3.29E-02	5.936	2.92E-09	***
Region Sicilia	1.92E-01	1.21E+00	2.07E-02	2.18E-02	8.8	2E-16	***
Region Toscana	-6.24E-03	9.94E-01	2.21E-02	2.23E-02	-0.28	0.779621	
Region Trentino-Alto Adige	-3.55E-01	7.01E-01	4.31E-02	4.28E-02	-8.292	2E-16	***
Region Umbria	5.44E-02	1.06E+00	4.13E-02	4.15E-02	1.312	0.189687	
Region Valle d'Aosta	-9.14E-02	9.13E-01	1.14E-01	1.17E-01	-0.78	0.435531	
Region Veneto	-8.76E-02	9.16E-01	2.10E-02	2.12E-02	-4.13	3.63E-05	***
Sector A	-4.14E-01	6.61E-01	3.77E-02	4.01E-02	-10.311	2E-16	***
Sector Others	-1.63E-01	8.50E-01	1.57E-01	1.55E-01	-1.053	0.292157	
Sector C	-6.59E-02	9.36E-01	1.77E-02	1.93E-02	-3.42	0.000626	***
Sector D	-1.72E-01	8.42E-01	3.18E-02	3.50E-02	-4.897	9.75E-07	***
Sector E	-3.66E-01	6.93E-01	7.90E-02	7.93E-02	-4.616	3.91E-06	***
Sector F	-6.60E-02	9.36E-01	1.51E-02	1.91E-02	-3.448	0.000564	***
Sector H	1.23E-01	1.13E+00	2.46E-02	2.75E-02	4.474	7.66E-06	***
Sector I	1.54E-01	1.17E+00	1.95E-02	2.70E-02	5.704	1.17E-08	***
Sector J	-1.28E-01	8.80E-01	2.46E-02	2.79E-02	-4.584	4.56E-06	***
Sector K	-8.20E-02	9.21E-01	3.46E-02	3.80E-02	-2.157	0.031038	*
Sector L	-5.30E-01	5.89E-01	2.06E-02	2.60E-02	-20.369	2E-16	***
Sector M	-6.11E-02	9.41E-01	1.85E-02	2.26E-02	-2.701	0.006913	**
Sector N	8.03E-02	1.08E+00	2.14E-02	2.68E-02	2.99	0.002788	**
Sector P	-2.78E-01	7.57E-01	5.03E-02	5.38E-02	-5.158	2.49E-07	***
Sector Q	-3.96E-01	6.73E-01	3.77E-02	4.18E-02	-9.467	2E-16	***
Sector R	1.65E-01	1.18E+00	3.09E-02	3.44E-02	4.788	1.69E-06	***
Sector S	1.14E-01	1.12E+00	3.67E-02	4.16E-02	2.731	0.006322	**
Innovative Startup	-1.99E+00	1.36E-01	3.33E-01	3.19E-01	-6.252	4.05E-10	***
Production costs	2.96E-05	1.00E+00	4.43E-06	8.48E-06	3.495	0.000474	***
Total from sales	-1.13E-04	1.00E+00	6.52E-06	3.51E-05	-3.208	0.001339	**
Index Liquidity	-2.58E-02	9.74E-01	3.03E-03	3.35E-03	-7.717	1.19E-14	***
Employees	-7.58E-03	9.92E-01	7.35E-04	3.18E-03	-2.384	0.017148	*

Signif. code: $p^{***} < 0.0001, p^{**} < 0.001, p^{*} < 0.05, p^{\cdot} < 0.1$

Table 4.6: Cox regression summary

β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8
69.51	-11.68	-11.54	-1.84	-11.1	-10.45	-10.33	-1.67	-10.89
β_9	β_{10}	β_{11}	β_{12}	β_{13}	β_{14}	β_{15}	β_{16}	β_{17}
-8.13	-9.69	-9.47	-10.26	-9.38	-10.2	-1.51	-10.98	-8.32
β_{19}	β_{20}	β_{21}	β_{23}	β_{24}	β_{26}	β_{27}	β_{28}	β_{29}
-8.25	-8.09	-9.32	-9.1	-10.05	-8.06	-8.89	-9.57	-9.49
β_{30}	β_{31}	β_{32}	β_{34}	β_{38}	β_{40}	β_{42}	β_{43}	β_{46}
-9.27	-1.11	-11.09	-8.3	-8.16	-8.31	-9.35	-8.3	-8.39
β_{47}	β_{48}	β_{52}	β_{54}	β_{55}	β_{56}	β_{57}	β_{58}	β_{59}
-8.91	-9.76	-8.38	-8.17	-8.43	-8.8	-9.34	-7.99	-8.28
β_{60}	β_{61}	β_{62}	β_{63}	β_{64}	β_{65}	β_{68}	β_{77}	β_{80}
-8.96	-8.18	-9.55	-0.43	-10.76	-8.26	-8.34	-8.42	-8
β_{84}	β_{87}	β_{92}	β_{95}	β_{96}	β_{97}	β_{104}	β_{108}	β_{111}
-9.19	-8.2	-8.28	-8.53	-9.63	-8.15	-8.28	-8.26	-8.16
β_{112}	β_{115}	β_{116}	β_{119}	β_{120}	β_{122}	β_{124}	β_{125}	β_{126}
-8.8	-9.22	-8.41	-8.13	-9.07	-8.06	-8.43	-9.09	-9.06
β_{127}	β_{128}	β_{130}	β_{136}	β_{155}	β_{168}	β_{175}	β_{184}	β_{191}
-0.03	-10.71	-8.49	-8.51	-8.15	-9.26	-8.47	-8.3	-8.06
β_{192}	β_{194}	β_{208}	β_{216}	β_{223}	β_{224}	β_{230}	β_{231}	β_{232}
-9.22	-8.32	-8.37	-8.15	-8.27	-8.81	-8.26	-8.92	-8.25
β_{238}	β_{240}	β_{245}	β_{248}	β_{250}	β_{251}	β_{252}	β_{254}	β_{255}
-8.17	-8.49	-8.47	-8.28	-8.21	-8.94	-9.29	-8.24	-0.07
β_{256}	β_{260}	β_{272}	β_{311}	β_{336}	β_{351}	β_{368}	β_{383}	β_{384}
-10.34	-8.35	-8.21	-8.44	-8.72	-8.25	-8.26	-8.33	-9.16
β_{388}	β_{432}	β_{448}	β_{460}	β_{463}	β_{464}	β_{477}	β_{480}	β_{491}
-8.23	-8.46	-8.22	-8.39	-8.91	-8.39	-8.13	-8.05	-8.36
β_{496}	β_{500}	β_{503}	β_{504}	β_{508}	β_{511}	β_{512}	β_{544}	β_{623}
-8.04	-8.07	-8.33	-8.82	-8.32	-0.06	-10.24	-8.2	-8.21
β_{672}	β_{703}	β_{768}	β_{776}	β_{896}	β_{920}	β_{927}	β_{955}	β_{960}
-8.21	-8.15	-9.43	-8.25	-8.52	-8.18	-8.72	-8.05	-8.4
β_{983}	β_{1007}	β_{1008}	β_{1016}	β_{1022}	β_{1023}	β_t		
-8.24	-8.31	-7.98	-8.07	-8.58	6.85	-0.03		

Table 4.7: Coefficients of Bayesian logistic regression

Bibliography

[1] A. Abbruzzo and A. M. Mineo. "Inferring networks from high-dimensional data with mixed variables". In: *Advances in Complex Data Modeling and Computational Methods in Statistics*. Springer, 2015, pp. 1–15.

[2] G. C. de Abreu, R. Labouriau, and D. Edwards. "High-dimensional graphical model search with gRapHD R package". In: *arXiv preprint arXiv:0909.1234* (2009).

[3] S. Acid, L. M. de Campos, J. M. Fernández-Luna, S. Rodrıguez, J. M. Rodrıguez, and J. L. Salcedo. "A comparison of learning algorithms for Bayesian networks: a case study based on data from an emergency medical service". In: *Artificial intelligence in medicine* 30.3 (2004), pp. 215–232.

[4] Z. J. Acs, C. Armington, and T. Zhang. "The determinants of new-firm survival across regional economies: The role of human capital stock and knowledge spillover". In: *Papers in Regional Science* 86.3 (2007), pp. 367–391.

[5] C. C. Aggarwal and P. Zhao. "Towards graphical models for text processing". In: *Knowledge and information systems* 36.1 (2013), pp. 1–21.

[6] H. Akaike. "A new look at the statistical model identification". In: *IEEE transactions on automatic control* 19.6 (1974), pp. 716–723.

[7] J. Albert and J. Hu. *Probability and Bayesian Modeling*. CRC Press, 2019.

[8] M. Aldinucci, S. Bagnasco, S. Lusso, P. Pasteris, S. Rabellino, and S. Vallero. "OCCAM: a flexible, multi-purpose and extendable HPC cluster". In: *Journal of Physics: Conference Series*. Vol. 898. 8. 2017, p. 082039.

[9] M. Aldinucci, S. Rabellino, M. Pironti, F. Spiga, P. Viviani, M. Drocco, M. Guerzoni, G. Boella, M. Mellia, P. Margara, et al. "HPC4AI: an AI-on-demand federated platform endeavour". In: *Proceedings of the 15th ACM International Conference on Computing Frontiers*. 2018, pp. 279–286.

[10] M. Althabiti and M. Abdullah. "CLASSIFICATION OF CONCEPT DRIFT IN EVOLVING DATA STREAM". In: *Emerging Extended Reality Technologies for Industry 4.0: Early Experiences with Conception, Design, Implementation, Evaluation and Deployment* (2020), p. 189.

[11] N. Altman and M. Krzywinski. "The curse (s) of dimensionality". In: *Nat Methods* 15.6 (2018), pp. 399–400.

[12] A. S. Asratian, T. M. Denley, and R. Häggkvist. *Bipartite graphs and their applications*. Vol. 131. Cambridge university press, 1998.

[13] D. B. Audretsch. "Innovation, growth and survival". In: *International journal of industrial organization* 13.4 (1995), pp. 441–457.

[14] A. Azzalini and B. Scarpa. *Data analysis and data mining: An introduction*. OUP USA, 2012.

[15] M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno. "Early drift detection method". In: *Fourth international workshop on knowledge discovery from data streams*. Vol. 6. 2006, pp. 77–86.

[16] G. Bagallo and D. Haussler. "Boolean feature discovery in empirical learning". In: *Machine learning* 5.1 (1990), pp. 71–99.

[17] E. Bair and R. Tibshirani. "Semi-supervised methods to predict patient survival from gene expression data". In: *PLoS Biol* 2.4 (2004), e108.

[18] M. Balasubramanian, E. L. Schwartz, J. B. Tenenbaum, V. de Silva, and J. C. Langford. "The isomap algorithm and topological stability". In: *Science* 295.5552 (2002), pp. 7–7.

[19] R. Battiti. "Using mutual information for selecting features in supervised neural net learning". In: *IEEE Transactions on neural networks* 5.4 (1994), pp. 537–550.

[20] H.-U. Bauer and W. Schöllhorn. "Self-organizing maps for the analysis of complex movement patterns". In: *Neural Processing Letters* 5.3 (1997), pp. 193–199.

[21] R. E. Bellman. *Adaptive control processes: a guided tour*. Princeton university press, 2015.

[22] D. Bertsimas, A. King, R. Mazumder, et al. "Best subset selection via a modern optimization lens". In: *Annals of statistics* 44.2 (2016), pp. 813–852.

[23] R. J. C. Bose, W. M. van der Aalst, I. Žliobaitė, and M. Pechenizkiy. "Handling concept drift in process mining". In: *International Conference on Advanced Information Systems Engineering*. Springer. 2011, pp. 391–405.

[24] A. Brun, H.-J. Park, H. Knutsson, and C.-F. Westin. "Coloring of DT-MRI fiber traces using Laplacian eigenmaps". In: *International conference on computer aided systems theory*. Springer. 2003, pp. 518–529.

[25] C. Carota, A. Durio, and M. Guerzoni. "An application of graphical models to the innobarometer survey: A map of firms' innovative behaviour". In: *Department of Economics and Statistics" Cognetti de Martiis" Working Paper Series* (2014).

[26] C. Carota, A. Durio, and M. Guerzoni. "AN APPLICATION OF GRAPHICAL MODELS TO THE INNOBAROMETER SURVEY: A MAP OF FIRMS'INNOVATIVE BEHAVIOUR". In: *Italian Journal of Applied Statistics, 25 1* 25.1 (2015).

[27] K. M. Carter, R. Raich, W. G. Finn, and A. O. Hero III. "Information-geometric dimensionality reduction". In: *IEEE Signal Processing Magazine* 28.2 (2011), pp. 89–99.

[28] H. Choi and S. Choi. "Robust kernel isomap". In: *Pattern Recognition* 40.3 (2007), pp. 853–862.

[29] C. Chow and C. Liu. "Approximating discrete probability distributions with dependence trees". In: *IEEE transactions on Information Theory* 14.3 (1968), pp. 462–467.

[30] P. J. Chuard, M. Vrtılek, M. L. Head, and M. D. Jennions. "Evidence that nonsignificant results are sometimes preferred: Reverse P-hacking or selective reporting?" In: *PLoS biology* 17.1 (2019), e3000127.

[31] J. Clarke and M. West. "Bayesian Weibull tree models for survival analysis of clinico-genomic data". In: *Statistical methodology* 5.3 (2008), pp. 238–262.

[32] J. A. Costa and A. Hero. "Classification constrained dimensionality reduction". In: *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.* Vol. 5. IEEE. 2005, pp. v–1077.

[33] D. R. Cox and N. Wermuth. *Multivariate dependencies: Models, analysis and interpretation*. Vol. 67. CRC Press, 2014.

[34] M. d. C. Cunha and J. Sousa. "Water distribution network design optimization: simulated annealing approach". In: *Journal of water resources planning and management* 125.4 (1999), pp. 215–221.

[35] J. P. Cunningham and Z. Ghahramani. "Linear dimensionality reduction: Survey, insights, and generalizations". In: *The Journal of Machine Learning Research* 16.1 (2015), pp. 2859–2900.

[36] P. Dellaportas, J. J. Forster, and I. Ntzoufras. "On Bayesian model and variable selection using MCMC". In: *Statistics and Computing* 12.1 (2002), pp. 27–36.

[37] F. Delmar, A. McKelvie, and K. Wennberg. "Untangling the relationships among growth, profitability and survival in new firms". In: *Technovation* 33.8-9 (2013), pp. 276–291.

[38] A. P. Dempster. "Covariance selection". In: *Biometrics* (1972), pp. 157–175.

[39] R. Diestel. "Graph Theory (Graduate Texts in Mathematics)(Springel-Verlag, Heidelberg)". In: (2005).

[40] S. N. Dorogovtsev, A. V. Goltsev, J. F. Mendes, and A. N. Samukhin. "Spectra of complex networks". In: *Physical Review E* 68.4 (2003), p. 046109.

[41] K.-B. Duan and S. S. Keerthi. "Which is the best multiclass SVM method? An empirical study". In: *International workshop on multiple classifier systems*. Springer. 2005, pp. 278–285.

[42] D. Edwards, G. C. De Abreu, and R. Labouriau. "Selecting high-dimensional mixed graphical models using minimal AIC or BIC forests". In: *BMC bioinformatics* 11.1 (2010), p. 18.

[43] J. Eklund, S. Karlsson, et al. *An embarrassment of riches: Forecasting using large panels*. Citeseer, 2007.

[44] R. Elwell and R. Polikar. "Incremental learning of concept drift in nonstationary environments". In: *IEEE Transactions on Neural Networks* 22.10 (2011), pp. 1517–1531.

[45] B. Feil and J. Abonyi. "Introduction to fuzzy data mining methods". In: *Handbook of research on fuzzy information processing in databases*. IGI Global, 2008, pp. 55–95.

[46] M. Fiore and M. D. Campos. "The algebra of directed acyclic graphs". In: *Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky*. Springer, 2013, pp. 37–51.

[47] D. Fouskakis, I. Ntzoufras, D. Draper, et al. "Bayesian variable selection using cost-adjusted BIC, with application to cost-effective measurement of quality of health care". In: *The Annals of Applied Statistics* 3.2 (2009), pp. 663–690.

[48] D. Fouskakis, I. Ntzoufras, D. Draper, et al. "Power-expected-posterior priors for variable selection in Gaussian linear models". In: *Bayesian Analysis* 10.1 (2015), pp. 75–107.

[49] B. J. Frey, F. R. Kschischang, H.-A. Loeliger, and N. Wiberg. "Factor graphs and algorithms". In: *Proceedings of the Annual Allerton Conference on Communication Control and Computing*. Vol. 35. Citeseer. 1997, pp. 666–680.

[50] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York, 2001.

[51] N. Friedman. "Inferring cellular networks using probabilistic graphical models". In: *Science* 303.5659 (2004), pp. 799–805.

[52] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. "Using Bayesian networks to analyze expression data". In: *Journal of computational biology* 7.3-4 (2000), pp. 601–620.

[53] K. Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.

[54] R. A. Gallego, R. Romero, and A. J. Monticelli. "Tabu search algorithm for network synthesis". In: *IEEE Transactions on Power Systems* 15.2 (2000), pp. 490–495.

[55] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia. "A survey on concept drift adaptation". In: *ACM computing surveys (CSUR)* 46.4 (2014), pp. 1–37.

[56] D. Geiger, T. Verma, and J. Pearl. "Identifying independence in Bayesian networks". In: *Networks* 20.5 (1990), pp. 507–534.

[57] P. A. Geroski. "What do we know about entry?" In: *International Journal of Industrial Organization* 13.4 (1995), pp. 421–440.

[58] G. Gigerenzer and R. Selten. *Bounded rationality: The adaptive toolbox*. MIT press, 2002.

[59] P. Giot and A. Schwienbacher. "IPOs, trade sales and liquidations: Modelling venture capital exits using survival analysis". In: *Journal of Banking & Finance* 31.3 (2007), pp. 679–702.

[60] M. Guerzoni, C. R. Nava, and M. Nuccio. "Start-ups survival through a crisis. Combining machine learning with econometrics to measure innovation". In: *Economics of Innovation and New Technology* (2020), pp. 1–26.

[61] I. Guyon and A. Elisseeff. "An introduction to variable and feature selection". In: *Journal of machine learning research* 3.Mar (2003), pp. 1157–1182.

[62] J. F. Hair. "Multivariate data analysis". In: (2009).

[63] A. U. Haq, J. Li, M. H. Memon, M. H. Memon, J. Khan, and S. M. Marium. "Heart disease prediction system using model of machine learning and sequential backward selection algorithm for features selection". In: *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*. IEEE. 2019, pp. 1–4.

[64] M. Harries. *Splice-2 Comparative Evaluation: Electricity Pricing*. PANDORA electronic collection. University of New South Wales, School of Computer Science and Engineering, 1999. URL: https://books.google.it/books?id=1Zr1vQAACAAJ.

[65] X. He, D. Cai, S. Yan, and H.-J. Zhang. "Neighborhood preserving embedding". In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Vol. 2. IEEE. 2005, pp. 1208–1213.

[66] M. L. Head, L. Holman, R. Lanfear, A. T. Kahn, and M. D. Jennions. "The extent and consequences of p-hacking in science". In: *PLoS Biol* 13.3 (2015), e1002106.

[67] S. Højsgaard, D. Edwards, and S. Lauritzen. *Graphical models with R*. Springer Science & Business Media, 2012.

[68] D. E. Holmes. *Innovations in Bayesian networks: theory and applications*. Vol. 156. Springer, 2008.

[69] D. Holtz-Eakin, D. Joulfaian, and H. S. Rosen. "Sticking it out: Entrepreneurial survival and liquidity constraints". In: *Journal of Political economy* 102.1 (1994), pp. 53–75.

[70] J. Huang, S. Ma, and H. Xie. "Regularized estimation in the accelerated failure time model with high-dimensional covariates". In: *Biometrics* 62.3 (2006), pp. 813–820.

[71] A. Hyytinen, M. Pajarinen, and P. Rouvinen. "Does innovativeness reduce startup survival rates?" In: *Journal of Business Venturing* 30.4 (2015), pp. 564–581.

[72] H. Ishwaran, U. B. Kogalur, E. Z. Gorodeski, A. J. Minn, and M. S. Lauer. "High-dimensional variable selection for survival data". In: *Journal of the American Statistical Association* 105.489 (2010), pp. 205–217.

[73] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[74] A. Jara and T. Hanson. "A class of mixtures of dependent tail-free processes". In: *Biometrika* 98.3 (2011), pp. 553–566.

[75] O. C. Jenkins and M. J. Mataric. "Deriving action and behavior primitives from human motion data". In: *IEEE/RSJ international conference on intelligent robots and systems*. Vol. 3. IEEE. 2002, pp. 2551–2556.

[76] L. O. Jimenez and D. A. Landgrebe. "Supervised classification in high-dimensional space: geometrical, statistical, and asymptotical properties of multivariate data". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 28.1 (1998), pp. 39–54.

[77] R. A. Johnson, D. W. Wichern, et al. *Applied multivariate statistical analysis*. Vol. 5. 8. Prentice hall Upper Saddle River, NJ, 2002.

[78] M. I. Jordan et al. "Graphical models". In: *Statistical science* 19.1 (2004), pp. 140–155.

[79] S. Kaski. "Dimensionality reduction by random mapping: Fast similarity computation for clustering". In: *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227)*. Vol. 1. IEEE. 1998, pp. 413–418.

[80] P. Khajehpour Tadavani. "Nonlinear dimensionality reduction by manifold unfolding". In: (2013).

[81] S. Klepper. "Entry, exit, growth, and innovation over the product life cycle". In: *The American Economic Review* (1996), pp. 562–583.

[82] R. Klinkenberg. "Learning drifting concepts: Example selection vs. example weighting". In: *Intelligent data analysis* 8.3 (2004), pp. 281–300.

[83] R. Klinkenberg and T. Joachims. "Detecting concept drift with support vector machines." In: *ICML*. 2000, pp. 487–494.

[84] R. Klinkenberg and I. Renz. "Adaptive information filtering: Learning in the presence of concept drifts". In: *Learning for text categorization* (1998), pp. 33–40.

[85] T. Kohonen. "Self-organized formation of topologically correct feature maps". In: *Biological cybernetics* 43.1 (1982), pp. 59–69.

[86] K. B. Korb and A. E. Nicholson. *Bayesian artificial intelligence*. CRC press, 2010.

[87] G. Kratzer and R. Furrer. "varrank: an R package for variable ranking based on mutual information with applications to observed systemic datasets". In: *arXiv preprint arXiv:1804.07134* (2018).

[88] J. B. Kruskal. "On the shortest spanning subtree of a graph and the traveling salesman problem". In: *Proceedings of the American Mathematical society* 7.1 (1956), pp. 48–50.

[89] S. Kullback. *Statistics and Information theory*. 1959.

[90] L. I. Kuncheva and C. O. Plumpton. "Adaptive learning rate for online linear discriminant classifiers". In: *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer. 2008, pp. 510–519.

[91] R. A. Kyle, M. A. Gertz, P. R. Greipp, T. E. Witzig, J. A. Lust, M. Q. Lacy, and T. M. Therneau. "A trial of three regimens for primary amyloidosis: colchicine alone, melphalan and prednisone, and melphalan, prednisone, and colchicine". In: *New England Journal of Medicine* 336.17 (1997), pp. 1202–1207.

[92] S. Lauritzen. "Graphical Models, ser". In: *Oxford Statistical Science Series. Oxford University Press* (1996).

[93] S. L. Lauritzen and D. J. Spiegelhalter. "Local computations with probabilities on graphical structures and their application to expert systems". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 50.2 (1988), pp. 157–194.

[94] S. L. Lauritzen and N. Wermuth. "Graphical models for associations between variables, some of which are qualitative and some quantitative". In: *The annals of Statistics* (1989), pp. 31–57.

[95] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007.

[96] J. A. Lee, A. Lendasse, N. Donckers, and M. Verleysen. "A robust nonlinear projection method". In: *European Symposium on Artificial Neural Networks (ESANN'2000)*. 2000.

[97] H. Li and Y. Luan. "Boosting proportional hazards models using smoothing splines, with applications to high-dimensional microarray data". In: *Bioinformatics* 21.10 (2005), pp. 2403–2409.

[98] J. Li and H. Liu. "Challenges of feature selection for big data analytics". In: *IEEE Intelligent Systems* 32.2 (2017), pp. 9–15.

[99] I. S. Lim, P. de Heras Ciechomski, S. Sarni, and D. Thalmann. "Planar arrangement of high-dimensional biomedical data sets by isomap coordinates". In: *16th IEEE Symposium Computer-Based Medical Systems, 2003. Proceedings.* IEEE. 2003, pp. 50–55.

[100] O. Maimon and L. Rokach. "Data mining and knowledge discovery handbook". In: (2005).

[101] F. Malerba and L. Orsenigo. "Technological regimes and sectoral patterns of innovative activities". In: *Industrial and Corporate Change* 6.1 (1997), pp. 83–118.

[102] D. Margaritis. *Learning Bayesian network model structure from data*. Tech. rep. Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science, 2003.

[103] L. D. Miller, J. Smeds, J. George, V. B. Vega, L. Vergara, A. Ploner, Y. Pawitan, P. Hall, S. Klaar, E. T. Liu, et al. "An expression signature for p53 status in human breast cancer predicts mutation status, transcriptional effects, and patient survival". In: *Proceedings of the National Academy of Sciences* 102.38 (2005), pp. 13550–13555.

[104] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. "Deepfool: a simple and accurate method to fool deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2574–2582.

[105] J. Munkres. "Topology, a first course. Prentice Hall Inc., Englewood Cliffs (NJ), 1975 (2000)." In: ().

[106] U. Murty and A. Bondy. *Graph Theory (Graduate Texts in Mathematics 244)*. 2008.

[107] P. Musso and S. Schiavo. "The impact of financial constraints on firm survival and growth". In: *Journal of Evolutionary Economics* 18.2 (2008), pp. 135–149.

[108] B. Nadler, S. Lafon, R. R. Coifman, and I. G. Kevrekidis. "Diffusion maps, spectral clustering and reaction coordinates of dynamical systems". In: *Applied and Computational Harmonic Analysis* 21.1 (2006), pp. 113–127.

[109] P. M. Narendra and K. Fukunaga. "A branch and bound algorithm for feature subset selection". In: *IEEE Computer Architecture Letters* 26.09 (1977), pp. 917–922.

[110] D. V. Nguyen and D. M. Rocke. "Tumor classification by partial least squares using microarray gene expression data". In: *Bioinformatics* 18.1 (2002), pp. 39–50.

[111] K. Nordhausen. "An Introduction to Statistical Learning—with Applications in R by Gareth James, Daniela Witten, Trevor Hastie & Robert Tibshirani". In: *International Statistical Review* 82.1 (2014), pp. 156–157.

[112] R. B. O'Hara, M. J. Sillanpää, et al. "A review of Bayesian variable selection methods: what, how and which". In: *Bayesian analysis* 4.1 (2009), pp. 85–117.

[113] M. Y. Park and T. Hastie. "L1-regularization path algorithm for generalized linear models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69.4 (2007), pp. 659–677.

[114] M. D. Patil and S. S. Sane. "Dimension reduction: A review". In: *International Journal of Computer Applications92* 16 (2014), pp. 23–29.

[115] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[116] K. Pearson. "LIII. On lines and planes of closest fit to systems of points in space". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[117] D. PEH. *RO Duda, PE Hart, and DG Stork, Pattern Classification*. 2001.

[118] S. E. Pérez, A. S. Llopis, and J. A. S. Llopis. "The determinants of survival of Spanish manufacturing firms". In: *Review of Industrial Organization* 25.3 (2004), pp. 251–273.

[119] A. C. Rencher. *Methods of multivariate analysis*. Vol. 492. John Wiley & Sons, 2003.

[120] J. Rissanen. *Information and complexity in statistical modeling*. Springer Science & Business Media, 2007.

[121] J. Rissanen. "Stochastic complexity". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 49.3 (1987), pp. 223–239.

[122] S. T. Roweis and L. K. Saul. "Nonlinear dimensionality reduction by locally linear embedding". In: *science* 290.5500 (2000), pp. 2323–2326.

[123] E. Santarelli and M. Vivarelli. "Entrepreneurship and the process of firms' entry, survival and growth". In: *Industrial and corporate change* 16.3 (2007), pp. 455–488.

[124] A. Saxena, A. Gupta, and A. Mukerjee. "Non-linear dimensionality reduction by locally linear isomaps". In: *International Conference on Neural Information Processing*. Springer. 2004, pp. 1038–1043.

[125] D. K. Saxena and K. Deb. "Dimensionality reduction of objectives and constraints in multi-objective optimization problems: A system design perspective". In: *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE. 2008, pp. 3204–3211.

[126] B. Schölkopf, A. Smola, and K.-R. Müller. "Nonlinear component analysis as a kernel eigenvalue problem". In: *Neural computation* 10.5 (1998), pp. 1299–1319.

[127] G. Schwarz et al. "Estimating the dimension of a model". In: *The annals of statistics* 6.2 (1978), pp. 461–464.

[128] D. W. Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

[129] D. W. Scott and J. R. Thompson. "Probability density estimation in higher dimensions". In: *Computer Science and Statistics: Proceedings of the fifteenth symposium on the interface*. Vol. 528. North-Holland, Amsterdam. 1983, pp. 173–179.

[130] M. Scutari. "Learning Bayesian networks with the bnlearn R package". In: *arXiv preprint arXiv:0908.3817* (2009).

[131] N. D. Sidiropoulos, R. Bro, and G. B. Giannakis. "Parallel factor analysis in sensor array processing". In: *IEEE transactions on Signal Processing* 48.8 (2000), pp. 2377–2388.

[132] M. Spivak. *Calculus on manifolds: a modern approach to classical theorems of advanced calculus*. CRC press, 2018.

[133] R. Sternberg et al. "Regional dimensions of entrepreneurship". In: *Foundations and Trends® in Entrepreneurship* 5.4 (2009), pp. 211–340.

[134] R. Sternberg and T. Litzenberger. "Regional clusters in Germany–their geography and their relevance for entrepreneurial activities". In: *European Planning Studies* 12.6 (2004), pp. 767–791.

[135] J. H. Stock and M. W. Watson. *Introduction to econometrics*. 2015.

[136] E. B. Sudderth, M. J. Wainwright, and A. S. Willsky. "Embedded trees: Estimation of Gaussian processes on graphs with cycles". In: *IEEE Transactions on Signal Processing* 52.11 (2004), pp. 3136–3150.

[137] F. Tang and H. Ishwaran. "Random forest missing data algorithms". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 10.6 (2017), pp. 363–377.

[138] C. Taylor, G. Nakhaeizadeh, and C. Lanquillon. "Structural change and classification". In: *Workshop Notes on Dynamically Changing Domains: Theory Revision and Context Dependence Issues, 9th European Conf. on Machine Learning (ECML'97), Prague, Czech Republic*. April. 1997, pp. 67–78.

[139] J. B. Tenenbaum. "Mapping a manifold of perceptual observations". In: *Advances in neural information processing systems*. 1998, pp. 682–688.

[140] R. Tibshirani. "The lasso method for variable selection in the Cox model". In: *Statistics in medicine* 16.4 (1997), pp. 385–395.

[141] W. S. Torgerson. "Multidimensional scaling: I. Theory and method". In: *Psychometrika* 17.4 (1952), pp. 401–419.

[142] Ø. D. Trier, A. K. Jain, and T. Taxt. "Feature extraction methods for character recognition-a survey". In: *Pattern recognition* 29.4 (1996), pp. 641–662.

[143] I. Tsamardinos, L. E. Brown, and C. F. Aliferis. "The max-min hill-climbing Bayesian network structure learning algorithm". In: *Machine learning* 65.1 (2006), pp. 31–78.

[144] L. Van Der Maaten, E. Postma, and J. Van den Herik. "Dimensionality reduction: a comparative". In: *J Mach Learn Res* 10.66-71 (2009), p. 13.

[145] H. R. Varian. "Big data: New tricks for econometrics". In: *Journal of Economic Perspectives* 28.2 (2014), pp. 3–28.

[146] J. Wakefield. *Bayesian and frequentist regression methods*. Springer Science & Business Media, 2013.

[147] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean. "Characterizing concept drift". In: *Data Mining and Knowledge Discovery* 30.4 (2016), pp. 964–994.

[148] E. J. Wegman. "Hyperdimensional data analysis using parallel coordinates". In: *Journal of the American Statistical Association* 85.411 (1990), pp. 664–675.

[149] H. Whitney. "Differentiable manifolds". In: *Annals of Mathematics* (1936), pp. 645–680.

[150] G. Widmer and M. Kubat. "Learning in the presence of concept drift and hidden contexts". In: *Machine learning* 23.1 (1996), pp. 69–101.

[151] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin. "Graph embedding and extensions: A general framework for dimensionality reduction". In: *IEEE transactions on pattern analysis and machine intelligence* 29.1 (2006), pp. 40–51.

[152] M. Yuan and Y. Lin. "Model selection and estimation in the Gaussian graphical model". In: *Biometrika* 94.1 (2007), pp. 19–35.

[153] Q. Zhang, L. T. Yang, Z. Chen, and P. Li. "A survey on deep learning for big data". In: *Information Fusion* 42 (2018), pp. 146–157.