AIX-MARSEILLE UNIVERSITÉ

ECOLE DOCTORALE 184

Institut de Mathématiques de Marseille, UMR 7373

Thèse présentée pour obtenir le grade universitaire de docteur

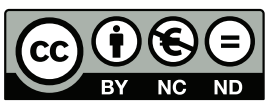
Discipline: Mathématiques appliquées
Spécialité: Statistiques

JEBREEN Kamel

Titre de la thèse: Modèles Graphiques pour la Classification et les Séries Temporelles

Soutenue le October 10, 2017 devant le jury composé de:

Anne FRANCOISE YAO	Professeur - Université Clermont Ferrand	Rapporteur
Abdulhakeem EIDEH	Professeur associé - Université Al Quds	Rapporteur
Denis ALLARD	Directeur de Recherche - INRA Avignon	Président
Liliane BEL	Professeur - Agro ParisTech	Examinateur
Badih GHATTAS	Maître de Conférences - Université d'Aix-Marseille	Directeur


BY NC ND

AIX-MARSEILLE UNIVERSITY

ECOLE DOCTORALE 184

Mathematical Institute of Marseille, UMR 7373

Submitted in partial fulfillment for the degree of

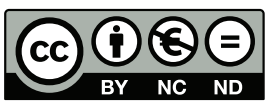
Discipline: Applied Mathematics
Specialty: Statistics

JEBREEN Kamel

Thesis title: Graphical Models for Classification and Time Series

Defended on October 10, 2017 before the committee:

Anne FRANCOISE YAO	Professor - Clermont Ferrand University	Reviewer
Abdulhakeem EIDEH	Associate Professor - Al Quds University	Reviewer
Denis ALLARD	Professor - INRA Avignon	President
Liliane BEL	Professor - Agro ParisTech	Examiner
Badih GHATTAS	Professor - Aix-Marseille University	Advisor


BY NC ND

Résumé

Dans cette thèse nous nous intéressons aux méthodes de classifications supervisées utilisant les réseaux Bayésiens. L'avantage majeur de ces méthodes est qu'elles peuvent prendre en compte les interactions entre les variables explicatives. Elles reposent cependant sur des hypothèses de loi et leur complexité algorithmique augmente considérablement avec le nombre de variables explicatives.
Dans une première partie nous proposons une procédure de discrétisation spécifique et une procédure de sélection de variables qui permettent d'améliorer considérablement les classifieurs basés sur des réseaux bayésiens. Cette procédure a montré de très bonnes performances empiriques sur un grand choix de jeux de données connus de l'entrepôt d'apprentissage automatique (UCI Machine Learning repository). Une application pour la prévision de type d'épilepsie à partir de de caractéristiques des patients extraites des images de Tomographie par émission de positrons (TEP) confirme l'efficacité de notre approche comparé à des approches communes de classifications supervisées.

Dans la deuxième partie de cette thèse nous nous intéressons à la modélisation des interactions entre des variables dans le contexte de séries chronologiques en grande dimension. Après avoir fait l'état de l'art des méthodes adaptées à ce contexte nous avons proposé deux nouvelles approches. La première, similaire à la technique "neighborhood Lasso" remplace la technique Lasso par des machines à vecteurs de supports. La motivation principale est que la méthode Lasso, bien qu'efficace pour la sélection des variables, elle ne l'est que dans le cas linéaire et le nombre de variables sélectionnées ne peut dépasser la taille de l'échantillon. La deuxième approche est un réseau bayésien restreint: les variables observées à chaque instant et à l'instant précédent sont utilisées dans un réseau dont la structure est restreinte. Nous montrons l'efficacité de ces approches par des simulations utilisant des donnés simulées issues de modèles linéaires, non-linéaires et un mélange des deux. Les approches proposées donnent en général de meilleurs résultats dans les cas non linéaires en grande dimension.

Mots-clés: Réseaux Bayésiens; Classification; Sélection de Variables; Discrétisation; Modèles Graphiques; Séries Temporelles.

Abstract

First, in this dissertation, we will show that Bayesian networks classifiers are very accurate models when compared to other classical machine learning methods. Discretising input variables often increase the performance of Bayesian networks classifiers, as does a feature selection procedure.
Different types of Bayesian networks may be used for supervised classification. We combine such approaches together with feature selection and discretisation to show that such a combination gives rise to powerful classifiers. A large choice of data sets from the UCI machine learning repository are used in our experiments, and the application to Epilepsy type prediction based on PET scan data confirms the efficiency of our approach.

Second, in this dissertation we also consider modelling interaction between a set of variables in the context of time series and high dimension. We suggest two approaches; the first is similar to the neighbourhood lasso where the lasso model is replaced by Support Vector Machines (SVMs); the second is a restricted Bayesian network for time series. We demonstrate the efficiency of our approaches simulations using linear and nonlinear data set and a mixture of both.

Keywords: Bayesian Networks; Classification; Feature Selection; Discretisation; Graphical Models; Time Series.

Dedication

To God,
my parents,
my supervisor, Badih Ghattas,
and my brothers and sisters.

Acknowledgment

I would like to express my sincere gratitude to my advisor Professor Badih GHATTAS of Aix Marseille university, France, who gave me the chance to work on one of the most important topics in the recent century, for his patience, motivation, and immense knowledge. His guidance helped me throughout my research, and his encouragement helped me to work in advanced and futuristic topics of machine learning.

Moreover, I wish to express my sincere thanks to the panel of expert reporters and examiners before whom I defended this work and for their remarks and questions that enriched this work.

In detail, I wish to thank Professor Anne FRANCOISE YAO of Clermont Ferrand University, France, and Associative Professor Abdulhakeem EIDEH of Al Quds University, Palestine, for their reports that highlighted the ideas and contribution of this work. I also thank Professor Denis ALLARD of INRA Avignon, France, for accepting to preside the defense committee, and Professor Liliane BEL of Agro ParisTech, France, for accepting to examine my work.

In particular, I extend my sincere thanks and appreciation to my parents, brothers, and sisters for their continuous encouragement throughout my studies.

Finally, I am so grateful to the "HERMES" project and its President Mr. Michel AUTRIC, Erasmus Mundus European programme, Action 2, Marseilles, France, for the financial support and administrative help extended during my PhD study. Also, I am so grateful to my university, An-Najah National University, Palestine, for allowing me to compete for this grant and supporting me to represent them.

Kamel JEBREEN

Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

Adj Adjacency or Score Matrix

BIC Bayesian Information Criterion

BN A Bayesian Network

$BN =< G, \Theta >$ A Bayesian Network BN with graph G and set of parametres Θ

Ber Bernoulli Distribution

$Beta$ Beta Function

$Child(X)$ Childern set of variable X

DAG A Directed Acyclic Graph

$E(\mathbf{\Theta})$ Expectation of $\mathbf{\Theta}$

G Graph

$G =< \mathbf{X}, \mathbf{E} >$ Graph G over the set of nodes $\mathbf{X}$ and the set of edges $\mathbf{E}$

$L(\mathbf{X})$ Likelihood Function of the set $\mathbf{X}$.

P Probability Function

PGM Probabilistic Graphical Model

$Pa(X)$ or $\pi(X)$ Parents set of variable X

X, Y, Z Variables (uppercase and not boldface).

Γ Gamma Function

Σ Variance Covariance Matrix

Θ Set of parametres $\{\theta_1, \theta_2, ...\}$

$\perp\!\!\!\perp$ Independent

$\mathbb{N}$ Set of natural numbers

$\mathbf{E}$ Set of edges

$\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ Sets of variables (nodes or features) (uppercase and boldface)

$\mathcal{D}$ Training data set

$\mathcal{N}$ Normal Distribution

$\mathcal{R}$ Set of real numbers

μ Mean

ρ^2 Correlation

$\rightarrow$ Arc

σ^2 Variance

$\subseteq$ Subset

θ Parameter

$d - separartion$ Direct Separation

n Number of instances (cases)

$nebr(X)$ Neighbors set of variable X

p Number of variables

var Variance

1.Introduction

Graphical models (Lauritzen (1996)) are a mixture of graph theory studies and probability. They are employed in various statistical studies, as they give an explicit, graphical, and interpretable representation of uncertain knowledge, based on the concept of conditional independence. The graph representing and visualising the relationships between many variables allows us to answer many queries, as it extracts the conditional independence relationships between the variables from their parametric forms. Hence, graphical models employ various algorithms to implement probabilistic inference efficiently. These models have been applied by a wide range of technicians in supervised and unsupervised learning, such as regression, classification, probabilistic expert models, image processing, and inferring genes networks.

The probabilistic graphical model is a graph in which the nodes correspond to random variables, and the edges in the graph represent the qualitative dependencies between the variables. The missing edges between two nodes mean that the corresponding random variables are conditionally independent given the other variables. The edges are parametric conditional distributions encoding the joint probability distribution over all the variables in the graph. We refer to the pattern of edges as the *structure* of the graph and the conditional probabilities as the *parameters* of joint probability distribution over all the variables in the graph.

There are two main types of graphical models: *undirected* and *directed* graphical models. Undirected graphical models, also known as *Markov Networks* or *Markov random fields*, are used more frequently by physics and vision communities, and directed graphical models, also known as *Bayesian Networks* (BNs), *Belief Networks*, *Generative Models*, *Causal Models*, among others, are used more by the machine learning communities and for Artificial Intelligence. It is also possible to have a model with both directed and undirected arcs that is called a *chain graph* or *partially directed graph*.

A *Gaussian Graphical Model* (GGM) is a graphical model in which the data belong to the Gaussian distribution. It is intensively used in the biology and engineering fields, such as for learning causal networks from systems biology. A graphical model can be static or dynamic. Dynamic graphical models use time course data to model the interaction between the variables where the arcs have only one direction, from time point t to its next time $t + 1$.

Bayesian networks (BNs), whether they are static models (Pearl (1988, 2009); Koller and Friedman (2009); Lauritzen (1996); Jebreen and Ghattas (2016); Jordan (1998); Friedman et al. (1997); Mitchell (1990); Murphy and Mian (1999); Korb and Nicholson (2010)) or dynamic models (Friedman et al. (1998); Meinshausen and Bühlmann (2006); Koller and Friedman (2009)), are directed acyclic graphs (DAGs) and among the most important supervised and unsuper-

vised model. This importance comes from the fact that they encode the dependencies among all variables; they deal with missing data; they are used to study the causality relations between variables and hence perform prediction; they are a good representation for combining prior knowledge and data, as they encode causal and probabilistic relations, and this prior knowledge can be used with statistical methods on data to improve the models accuracy; they encode the strength of causal relationships with probabilities and enable an effective representation and computation of the probability density function (pdf) over a set of random variables and are easy to adapt feature selection methods in Bayesian networks (Heckerman et al. (1995); Heckerman (1991)). As before, the edges between the nodes represent the probabilistic dependencies among the corresponding random variables based on the concept of a Markov blanket. These conditional dependencies in the graph are often estimated by using known statistical and computational methods.

The structure of a DAG is defined by the set of nodes (vertices) and the set of a directed edges. The nodes represent random variables and are drawn as circles labelled by the variable names. The edges represent a direct dependence among the variables and are drawn by arrows between the nodes. Moreover, an edge from node X_i to node X_j represents a statistical dependence between the corresponding variables. Hence, the arrow indicates that a value taken by variable X_j depends on the value taken by variable X_i, or the variable X_i's effect on the variable X_j. Here X_i is a parent of X_j, and, similarly, X_j is the child of X_i. Additionally, the sets of "descendants" is the set of nodes which can be reached on a direct path from the node, and "ancestor" nodes are the set of nodes from which the node can be reached on a direct path. The structure of the acyclic graph guarantees that there is no node that can be its own ancestor or its own descendant.

A Bayesian network reflects an important conditional independence statement to the factorisation of the joint probability of a set of nodes. That is, each variable is independent of its non-descendants in the graph, given its parents. This important property is used to reduce the number of parameters required to characterise the probability density function of the variables. Reducing the number of parameters is an efficient way to compute the posterior probabilities given the evidence.

Learning Bayesian network (Koller and Friedman (2009); Korb and Nicholson (2010); Heckerman et al. (1995)) is performed through two steps: *learning parameters* (the conditional probabilities among variables) and *learning structure*. Learning the structure is a more challenging problem than estimating the parameters that depend on the correct structure. The goal of learning BNs is to find a Bayesian network BN that approximates the joint distribution over the set of variables $\mathbf{X}$. Bayesian networks learning algorithms depend on the type of input data : *discrete data* (the variable takes the values from a finite set) or *continuous data*. When the training data is discrete the natural choice for the

joint probability distribution is a multinomial distribution, and when the training data is continuous, the natural choice for the joint probability distribution is multivariate normal distributions. In the context of Bayesian networks, this joint distribution is called the *global distribution*.

There are two approaches for learning parameters in Bayesian networks: *maximum likelihood estimation* (MLE) and *Bayesian estimation*. The simplest approach to learn the parameters in BNs is maximum likelihood estimation, which finds the set of parameters which maximises the likelihood function $L(\theta)$ over the observed data. It asymptotically converges toward the true probability if the proposed structure is correct. The Bayesian estimation method calculates the most probable parameters given the data. This is enough to weigh the parameters with an a priori knowledge. The most used prior is the Dirichlet distribution in discrete Bayesian networks.

After learning a Bayesian network, one can obtain various probability queries from the model. The computation of probability from a model is known as probabilistic *inference*. Inference using a Bayesian network, also called as belief updating, is based on the Bayes theorem. Exact inference in high dimension discrete networks becomes complex, so we introduce approximation algorithms for that, such as *forward* and *likelihood weighting* sampling algorithms.

Learning the structure of a Bayesian network can be considered as a specific example of selecting a probabilistic model that explains a given set of data. The structural learning of a Bayesian network in this dissertation is divided into two approaches: *conditional independence-based approach* and *score-based approach*.

The conditional independence based approach finds the causal relationships between the random variables and infers the structure of the graph. It performs a number of conditional independence tests on the data. The tests are usually done by using statistical or information theoretic measures. Most of these approaches need an exponential number of conditional independence tests that is unreliable.

The score-based approach maps each Bayesian network structure to a score and searches through all structures to find the best Bayesian network that fits the data set. The optimisation method is used to search for the best network structure, such as greedy search, iterated hill climbing, and simulated annealing.

Bayesian networks classifiers (Friedman et al. (1997); Heckerman (1991, 1997); Heckerman et al. (1995)) are types of Bayesian networks that aim to assign labels, levels, or categories to the instances in training data. Statisticians investigate many approaches for learning Bayesian classifiers in order to improve accuracy.

Classification using Bayesian networks form a big challenge, especially with regard to high dimensional data. Thus *feature selection* algorithms were proposed in order to remove the irrelevant and redundant variables. This dissertation illustrates various types of Bayesian networks classifiers: *Naive Bayes* (NB) (Maron and Kuhns (1960); Minsky (1961)), *Tree Augmented Naive Bayes* (TAN) (Friedman et al. (1997)), *Multinets classifier* (MN) (Friedman et al. (1997); Geiger and

Heckerman (1996)) and *Unrestricted Bayesian Network classifier* (UBN) (Heckerman et al. (1995); Heckerman (1997); Koller and Friedman (2009)).

In NB, the variables are conditionally independent given the class variable. It calculates the probability of each class, given the other variables, by Bayes rule. The TAN classifier takes into account the correlation between predictor variables. It allows each variable in the network to have at most one other parent. MN classifier estimates multiple Bayesian networks separately for each label of class variable, and these are then used to estimate the conditional probabilities. Bayes rule is then used to estimate the posterior probabilities. In UBN, the graph has no restriction like NB, TAN, or MN.

In this dissertation we describe and evaluate a *feature selection* approach for classification problems. Reunanen (Reunanen (2003)) views that there can be many reasons for selecting only a subset of the variables: it is less expensive to measure only a subset of the variables;the prediction accuracy may be improved through removing the irrelevant variables; the predictor to be built is usually simpler and potentially faster when less variables are used to build the model, and knowing which variables are relevant can give insight into the nature of the prediction problem. Therefore, the problem of focusing on the most relevant information has become increasingly important (Hruschka et al. (2004)).

We also describe and evaluate the *discretisation approach* on Bayesian networks classifiers, as data sets are often described by continuous variables. If the number of continuous variables is large, the model building for such data can be difficult and/or highly inefficient. Moreover, many data mining algorithms can only handle discrete attributes (Martínez (2010)).

We combine such approaches along with feature selection and discretisation and show that such combination gives rise to powerful classifiers.

Fitting the models using positron emission tomography (**PET**) scanning data (Guedj et al. (2015)) have an important position in field of classification. This is due to the predictions problems. These models study the essence of the relations between the regions of interest (ROI) in the brain and the causal relations between these variables as well as the direction of these relations if they exist (Ramsey et al. (2010); Smith et al. (2011)). Interpreting these causal relations requires complex and multivariate data. We view how to use Bayesian classifiers on PET scan data and introduce solutions for the failures that researchers have faced in previous studies to obtain the best performance within these large numbers of variables and their strong connectivities. The Bayesian network has the ability to search for the best structure among high dimensional data contrary to other methods. We focus on how to find the present connection relations between the regions to perform accuracy and correct its causality. We follow the approaches mentioned above: feature selection approach and discretisation approach with different Bayesian networks models to overcome the difficulties of fitting the models to these type of data sets, e.g, normality assumption on the data and redundant or irrelevant variables.

Dynamic models are the extensions of static models to temporal processes. This dissertation discusses the statistical methods used to reconstruct variable regulatory networks using time series data. Dynamic graphical model (DGM) is based on a dynamic system by discretising time and provide a static model that represents the probabilistic transition of the node at time t to the node at time $t+1$,; i.e, we assume that the variable at a given time $t+1$ only depends on the past variable observed at the previous time t. Also, we assume that the variables observed simultaneously are conditionally independent, given the past variables. These assumptions allow the existence of a dynamic graphical model representation. The DAG in dynamic model contains all the edges pointing out from a variable observed at time t towards a variable observed at the next time $t+1$. The direction of the edges according to time guarantees the acyclicity of the graph G.

We focus on estimating variables interactions, inferring causalities, and modelling the temporal changes of regulation behaviours. The problem in the practical application of DGMs is the high dimension of the data compared to the small sample size. Hence, if the number of variables is large, the parameters describing the graphical model (edge probabilities) quickly outnumber the data points. For this reason the graphical model in this case almost requires some form of regularised inference, such as penalised maximum likelihood or other shrinkage procedures. In modelling dynamic networks, a researcher is faced with the choice of whether to include extra features such as causality and temporal behaviours into the model. This choice of modelling paradigm is largely dependent on the type and quality of data available, relevant questions to be addressed, and statistical and computational considerations.

We are looking for simple relationships, such as variable X_i activating variable X_j. Also, we want to capture more complex approaches such as auto-regulations. Moreover most of the variables are not taking part in the temporal evolution of the model, so we want to determine the few active variables involved in the regulatory machinery and the relationships between them. The purpose of studying the dynamic models is that we want to infer a network representing the dependence relationships which govern a system composed of several agents from the observation of their activity across the short time series.

In this dissertation, the model is considered to be governed by the same rules during the whole experiment, i.e, the process is homogeneous. Many DGM representations have been proposed, multivariate autoregressive process (Opgen-Rhein and Strimmer (2007)), State Space or Hidden Markov Models (Beal et al. (2005)), nonparametric additive regression model (Imoto et al. (2002)), among others.

Such dynamic networks were described using static modelling; e.g, correlation network (Butte et al. (2000)), Covariance selection networks (Dempster (1972)), also known as concentration graph or graphical Gaussian model, (Whittaker (1990)) and dynamic Bayesian networks (Friedman et al. (1998); Murphy

and Mian (1999)).

Moreover, we suggest two approaches. The first is similar to the neighbourhood lasso when the lasso model is replaced by a support vector machine (SVM). The second is a restricted Bayesian network adapted for the time series.

It is interesting to apply this work to genetics and systems biology due to the high dimension of the data sets under study, which often contain several thousand variables and only a few tens or hundreds of observations. Such kinds of data are gene expression, protein signalling, and sequence data. This raises problems in both computational complexity and the statistical significance of the resulting networks, which are known as the "curse of dimensionality". Moreover, the data itself is difficult to model correctly due to the limited understanding of the underlying phenomena. Thus, we show the efficiency of our approaches by simulations using linear, nonlinear data set, and a mixture of both.

1.1. Dissertation Contribution

The contributions viewed in this dissertation are presented in two main parts. Firstly, the contributions related to classification using Bayesian networks, focusing on Epilepsy type prediction using PET Scan Data. Secondly, the contributions related to inferring linear and nonlinear interaction networks using neighbourhood support vector machines.

1.2. Dissertation Overview

This dissertation consist of four chapters. Chapter two presents the general definitions and concepts related to Bayesian networks. Chapter three presents the classification using Bayesian networks with discretisation and features selection approaches. Chapter four presents the previous approaches used to infer the interaction networks. Chapter five presents our approaches used to infer the interaction networks using concept of graphical models.

Chapter two is an illustration of the definitions and basics related to graphical models, Bayesian networks, conditional independencies in Bayesian networks, Markov property, learning both parameters and structure in Bayesian networks and inference on Bayesian networks.

Chapter three illustrates the classification using Bayesian networks classifiers (naive Bayes, tree augmented naive Bayes, multinet classifiers, and unrestricted Bayesian networks classifiers) with the methodologies used to improve the accuracy when using Bayesian network classification as application on PET scan data. **This chapter is based on the paper published in International Conference on Machine Learning and Applications (ICMLA 2016) CA, USA.**

Chapter four introduces the methodology to use vector autoregressive model, ridge regression, and static Bayesian network in dynamic process. Next, we introduce the approaches based on VAR model, such as Shrinkage approach, low order conditional dependence, among others.

Chapter five introduces our approaches: neighbourhood support machines and restricted Bayesian networks with experimental results on linear, nonlinear, and a mixture of both simulated data sets. **This chapter is based on the paper accepted in International Conference on Machine Learning and Applications (ICMLA 2017) CANCUN, MEXICO.**

2. Bayesian Networks

2.1. Introduction

Graphical models (Lauritzen (1996)) are a mixture of graph theory studies and probability. Bayesian networks (Pearl (1988)) are a type of directed graphical model used to study the causality relations between variables. A Bayesian network reflects an important conditional independence statement to the factorisation of the joint probability of a set of nodes.

Bayesian networks (Friedman et al. (1997); Heckerman (1991, 1997); Heckerman et al. (1995)) are powerful graphical models for representing the joint distribution of a random vector **X**. Bayesian networks encode the dependencies among all variables, and they are a good representation for combining prior knowledge and data, as they encode causal and probabilistic relations, and these prior knowledge can be used with statistical methods on data to improve the models accuracy. (Heckerman et al. (1995); Heckerman (1991)).

We use capital letters such as $X; Y; Z$ for variable names and lower-case letters such as $x; y; z$ to denote the specific values taken by those variables. Sets or vectors of variables are denoted by boldface capital letters such as $\mathbf{X}; \mathbf{Y}; \mathbf{Z}$, and the assignments of values to the variables in these sets are denoted by boldface lowercase letters $\mathbf{x}; \mathbf{y}; \mathbf{z}$. The variables are sometimes called **features**, **nodes**, or **vertices**, and the instances are sometimes called **cases** or **examples** (Friedman et al. (1997)).

In this chapter, we illustrate many definitions and algorithms that are used in graphical models to understand the concept of Bayesian networks, e.g, conditional independencies in graphical models and Markov property, which leads to the most important criterion in Bayesian networks known as *d-separation*. In the next sections, we introduce more details on how to learn the Bayesian model in both cases: discrete Bayesian networks and continuous Bayesian networks. In the next chapter, we draw an overview of our work of Bayesian classification as an application on PET scan data.

2.2. Graphical model: Introduction and definitions

In this section, we introduce the general notations used in the dissertation and some basic concepts about graphical models. To understand the concept of a probabilistic graphical model, the definition of conditional independence is essential.

Definition 2.2.1. [**Conditionally Independent, (Neapolitan (2003))**]. **Let $\mathbf{X}, \mathbf{Y}$ and $\mathbf{Z}$ be three disjoint sets of random variables, then $\mathbf{X}$ is condi-**

tionally independent of Y given Z and denoted by X $\perp\!\!\!\perp$ Y|Z if and only if

$$P(\mathbf{x}|\mathbf{y},\mathbf{z}) = P(\mathbf{x}|\mathbf{z}), \tag{2.1}$$

for any possible configuration $\mathbf{x}, \mathbf{y}$ and $\mathbf{z}$.

From equation 2.1, we can see that there is no important information offered by **Y** if the value of **Z** is known.

The variables may be **qualitative** (**categorical**) or **numerical** (**measurement**). When the case under study concerns a qualitative type that is only classified in categories and not numerically measured, the resulting data is known as categorical data. If on the other hand, the case is measured on a numeric scale, the resulting data consists of a set of numbers and is known as measurement data. In categorical data, if the variables X_j, $j = 1, ..., p$ have values from a finite set $\{1, ..., J\}$, then the data is called **discrete data**. Also, in numerics data, if the values of X_j's belong to the set of real numbers $\mathcal{R}$ then the data is called **continuous** data, (Neapolitan (2003); Johnson and Bhattacharyya (2014)).

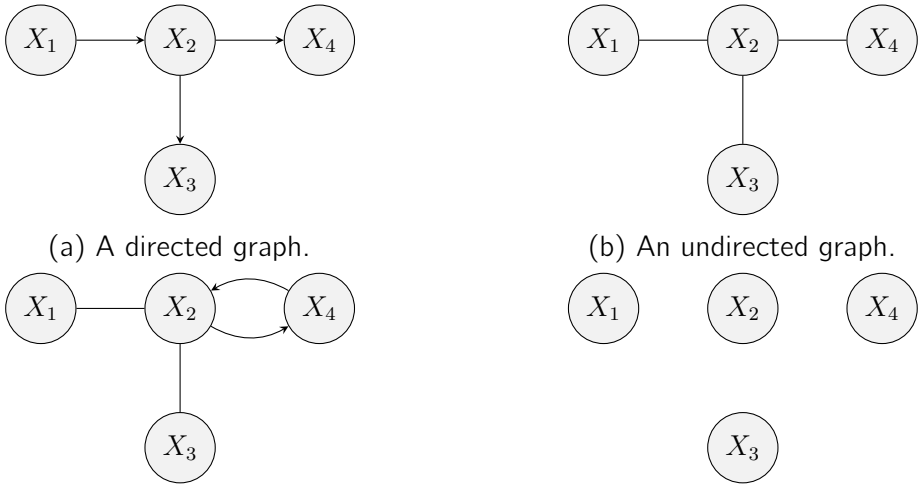
A graph in statistics means a **network** with nodes connected by links or edges. If the links are directed (arrows), then the graph is a **directed graph**, figure 2.1a. If the links are not directed (edges), the graph is an **undirected graph**, figure 2.1b. And when the graph has both directed and undirected links (edges and arrows), it is called a **partially directed graph**, figure 2.1c.

Definition 2.2.2. [**Graphical model**, (Jordan (1998))]. A graphical model is a probabilistic graphical model (PGM), representing the conditional dependence relations between random variables and denoted by $G = < \mathbf{X}, \mathbf{E} >$, where **X** is the nonempty set of nodes (variables) and **E** is the set of edges connected (links or arcs) . For more details see (Koller and Friedman (2009); Heckerman (1997); Jordan (1998)).

There are many types of graphical models, e.g, Bayesian networks (Friedman et al. (1997)), Markovian model (Edwards (2000)), and dynamic probability models (Friedman et al. (1998)).

Definition 2.2.3. [**Structure**, (Jordan (1998))]. Let $G =< \mathbf{X}, \mathbf{E} >$ be a graphical model; the structure of G is the pattern that demonstrates the connections between the set of nodes **X** by the set of edges **E**.

We will denote $\mathbf{X} = \mathbf{X}(G)$ as the set of vertices of G, and $\mathbf{E} = \mathbf{E}(G)$ as the set of edges of G. The nodes are connected by arcs or edges adjacent to each other, giving the **structure** of graph G; so if **E** is an empty set, we will have an empty graphical model, see figure 2.1d. In figure 2.1c, the set of nodes is $\{X_1, X_2, X_3, X_4\}$ and the set of arcs or edges is $\{(X_1 - X_2), (X_2 - X_3), (X_2 \rightarrow X_4), (X_4 \rightarrow X_2)\}$. But the set of edges or arcs in figure 2.1d is ϕ.



(a) A directed graph. (b) An undirected graph.

(c) A partially directed graph. (d) An empty graph.

Figure 2.1. – Graphical models.

Each edge or arc can be represented by an ordered or unordered pair of nodes. For example, if $e = (X, Y)$ represents the edge between node X and node Y and e is an ordered pair, then there is a directed arc from node X to node Y; $X \rightarrow Y$. In this case, the node X is the tail of the arc and the node Y is the head of the arc. If $e = (X, Y)$ is an unordered pair, then there is an edge (undirected arc) between node X and node Y; $X - Y$.

There are many properties related to the concept of graph structure, and one of them is the **path**.

Definition 2.2.4. [**Path**, (Koller and Friedman (2009))]. A path in G is a set of edges that relate two vertices of G. For a pair of vertices, there may exist multiple paths.

For example, the directed graphs in figure 2.1a has the set of these paths $\{(X_1, X_2), (X_1, X_2, X_4), (X_2, X_4), (X_2, X_3)\}$. In each path of a directed graph, the arcs follow the same direction and the paths are ordered. In partially directed graphs, figure 2.1c, the paths may be directed or undirected arcs, e.g, $\{(X_1, X_2)$, (X_2, X_1), (X_1, X_2, X_4), (X_3, X_2, X_1), $(X_2, X_3)\}$. Also, when you consider figure 2.2, you can see that we can reach the node X_1 from X_4 by a multiple paths (colored in red), (X_4, X_1), (X_4, X_2, X_1), and (X_4, X_5, X_1).

Definition 2.2.5. [**Cycle or loop path**, (Scutari and Denis (2014))]. The paths $(X_i, ..., X_j)$ in which $X_i = X_j$ are cycles paths.

For example, the path (X_2, X_4, X_2) in figure 2.1c is a cycle path that is forbidden in Bayesian networks.

Remark 2.2.1. A path is not necessarily directed.

Definition 2.2.6. [**Ancestors, descendants**, (Koller and Friedman (2009))]. If there is a path from vertex X_i to vertex X_j in a directed graph G, denoted by the sequence of ordered nodes $(X_i, ..., X_j)$, then X_i is the ancestor of X_j, and X_j is the descendant of X_i.

Definition 2.2.7. [**Parents and child**, (Koller and Friedman (2009))]. If there is an order path (X_i, X_j) in a direct graph represented by a single arc, then X_i is a parent of X_j and denoted by $\pi(X_i) = \{X_j\}$ or $Pa(X_i) = \{X_j\}$. Also, X_j is a child of X_i, and denoted by $child(X_j) = \{X_i\}$.

Definition 2.2.8. [**Neighbourhood or adjacent**, (Koller and Friedman (2009))]. Let $Z \in \mathbf{X}$ be a vertex of G, the neighbourhood of vertex Z in graph G is defined by the set of vertices that are directly connected to vertex Z and denoted as follows:

$$nebr(Z) = \{W \in \mathbf{X} | (Z, W) \in \mathbf{E}\}. \tag{2.2}$$

Consider the node X_3 in the directed graph, figure 2.1a, X_1 is an ancestor of X_3 and X_2 is the parent of X_3. The neighbourhood of X_3 is $\{X_2\}$. But the neighbourhood of X_2 in same graph is $\{X_1, X_3, X_4\}$. Also, in figure 2.2, $\pi(X_1) = \{X_2, X_4, X_5\}$ and $\pi(X_4) = \phi$.

Remark 2.2.2. In a directed graphical model, the children are also the descendants, and the parents are also the ancestors.

Definition 2.2.9. [**Acyclic structure**]. The structure of a directed graph is acyclic if it does not have any cycle or loop paths.

Definition 2.2.10. [**Roots and leafs**, (Koller and Friedman (2009))]. The node X is a root in graph G if it has at least one outgoing arc and no incoming arcs (without any parent), and it is a leaf node if it has at least one incoming arc and no outgoing arcs.

A graph can be uniquely defined by the adjacency matrix or concentration matrix; two vertices are said to be adjacent if they are directly connected by an edge.

Definition 2.2.11. [**Adjacency matrix**]. An adjacency matrix is a square matrix defining the finite graph in such a way that the elements of the matrix indicate whether the pairs of vertices are adjacent (connected) in the graph G (denoted by 1) or not adjacent (unconnected) in G (denoted by 0).

That is, if $X_1, ..., X_p$ belong to the vertices of graph $G =< \mathbf{X}, \mathbf{E} >$, then the adjacent matrix and denoted by Adj is defined as follows

$$Adj(i, j) = \begin{cases} 1, & \text{if } (X_i, X_j) \in \mathbf{E}, i, j = 1, ..., p \\ 0, & \text{if } (X_i, X_j) \notin \mathbf{E}, i, j = 1, ..., p \end{cases}. \tag{2.3}$$

Note that $Adj(i,j) = 1$ means that there is an arc from node j (column index) towards node i (row index) (see figure 2.2).

$$Adj(i,j) = \begin{array}{c} \begin{matrix} X_1 & X_2 & X_3 & X_4 & X_5 & X_6 \end{matrix} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{matrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \end{matrix}$$

Adjacent matrix of figure 2.2.

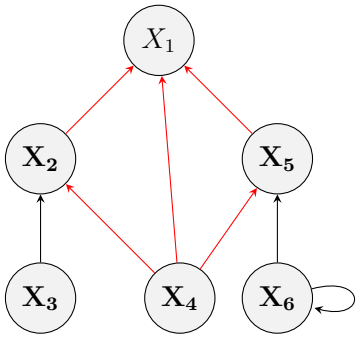


Figure 2.2. – Directed graph.

On the other hand, in a undirected graph, we cannot determine the tail and the head of the arc; i.e, $Adj(i,j) = 1$ means that there is an edge between node j and node i and vice versa. In this case, the adjacent matrix is symmetric, as shown in figure 2.3.

$$Adj(i,j) = \begin{array}{c} \begin{matrix} X_1 & X_2 & X_3 & X_4 & X_5 & X_6 \end{matrix} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \end{matrix}$$

Adjacent matrix of figure 2.3.

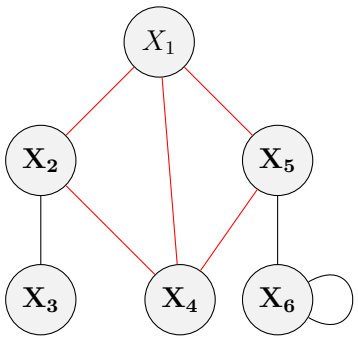


Figure 2.3. – Undirected graph.

Definition 2.2.12. [Complete DAG]. A Complete DAG is a graph that has edges between each pair of vertices.

2.3. Bayesian Networks

A Bayesian network (Pearl (1988); Jordan (1998)), as mentioned above, is a probabilistic acyclic graphical model for representing the probabilistic relationships between several random variables.

Definition 2.3.1. [Bayesian networks (BN)]. A Bayesian network (BN) is a directed acyclic graph (DAG) that encodes a joint probability distribution over a set of random variables $\mathbf{X} = (X_1, ..., X_p)$ and represents the probabilistic dependencies

between a given set of random variables denoted by $BN = < G, \Theta >$, where G is the graph structure of the set of nodes $\mathbf{X}$ and represents the independencies relations between the variables, and $\mathbf{\Theta}$ is the set of parameters that represents the amounts of the conditional independencies.

The keys behind Bayesian networks are factorisation, notational, and visualisation. That is, respectively, the factorisation of the probability distribution and using the graphical representation to visualise the conditional independence properties of the probability distribution.

Let us think graphically. If X, Y, and Z are random variables, consider their joint distribution $P(x, y, z)$. This joint distribution can be factorised as follows:

$$P(x, y, z) = P(z|x, y)P(x, y) = P(z|x, y)P(y|x)P(x). \tag{2.4}$$

So if the $P(x, y) = 0$, then the joint probability distribution $P(x, y, z) = 0$ even when we do not know the $P(z|x, y)$. Equation 2.4 can be represented graphically by figure 2.4c by starting from an empty graph represented by nodes without links as in figure 2.4a, then $y|x$ means that there is an arc from node Y to node X, see figure 2.4b, i.e., Y is a parent of X, also $z|x, y$ means that there are two arcs from node Z to nodes X and Y, see figure 2.4c, i.e., Z has two parents, X and Y. Additionally, X here is a root.

But let us assume that $Y \perp\!\!\!\perp Z|X$, then

$$P(Z|X, Y) = P(Z|X). \tag{2.5}$$

In this case, the red arc from Y to Z in figure 2.4c will be removed.

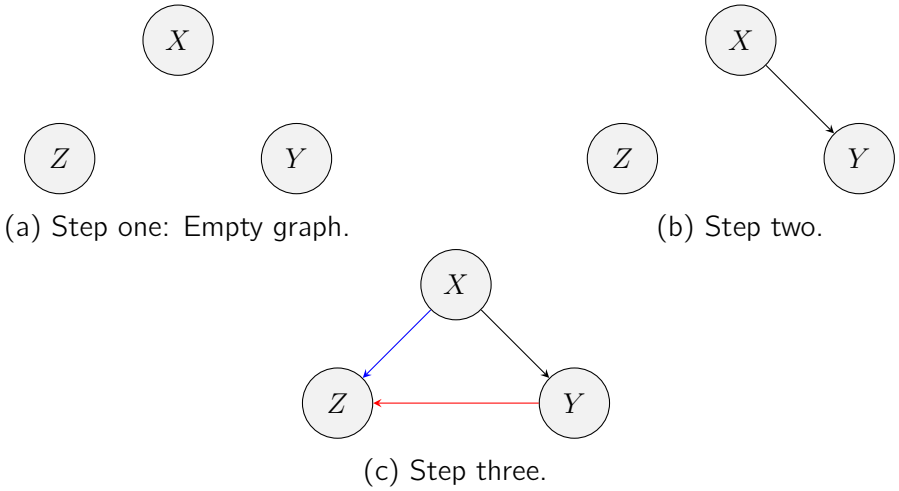
Definition 2.3.2. If $\mathbf{X} = (X_1, ..., X_p)$ is a vector of random variables and has a probability density function $P(\mathbf{X})$ and G is a DAG on p vertices, then we say $\mathbf{X}$ respects G (or $P(\mathbf{X})$ respects G) if

$$P(X_1, ..., X_p) = \prod_{j=1}^{p} P(X_j|\pi(X_j)). \tag{2.6}$$

This does not imply that all random variables are conditionally dependent, but some variables are conditionally independent. That is, if we have a set $\mathbf{X}$ of three mutual independent variables $\mathbf{X} = \{X_1, X_2, X_3\}$, then the joint probability distribution P over $\mathbf{X}$ is equal to

$$P(X_1, X_2, X_3) = P(X_1)P(X_2)P(X_3). \tag{2.7}$$

In this case, we say that the joint probability distribution P respects the DAG G, as given in figure 2.4a.



(a) Step one: Empty graph.

(b) Step two.

(c) Step three.

Figure 2.4. – Graphical models.

2.3.1. Generative process specification for probability distribution

This is a convention for specifying the probability distribution in a compact way, as certain conditional independence properties are omitted from the specification when we use this convention, and those conditional properties are implicitly implied (Bielza and Larrañaga (2014); Mitchell (1990)). For clarity, let X_1, X_2, X_3, X_4 and X_5 be random variables with the following particular generative distribution:

$$X_1, X_2 \sim Ber(\frac{1}{2}),\ X_1 \text{ and } X_2 \text{ are independent} \tag{2.8}$$

$$X_3 \sim \mathcal{N}(X_1 + X_2, \sigma^2) \tag{2.9}$$

$$X_4 \sim \mathcal{N}(aX_2 + b, 1) \tag{2.10}$$

$$X_5 = \begin{cases} 1, & \text{if } X_4 \geq 0 \\ 0, & \text{otherwise} \end{cases}, \tag{2.11}$$

where the symbol $\mathcal{N}$ denotes the normal distribution. The convention is when we say that X_1, X_2, X_3, X_4 and X_5 have the probability distribution mentioned above, and the joint distribution of $\mathbf{X} = (X_1, X_2, X_3, X_4, X_5)$ respects the graph shown in figure 2.5, that is

$$P(X_1, X_2, X_3, X_4, X_5) = P(X_1)P(X_2)P(X_3|X_1, X_2)P(X_4|X_2)P(X5|X_4) \tag{2.12}$$

Note that X_1 and X_2 are independent from equation 2.8, so there are no arrows between them and X_3 depends uniquely on X_1 and X_2 from equation 2.9; also, X_4 depends on X_2 from equation 2.10 and X_5 depends on X_4 from equa-

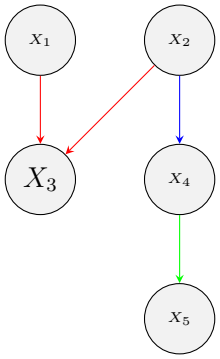


Figure 2.5. – The joint distribution of $\mathbf{X} = (X_1, X_2, X_3, X_4, X_5)$ represented by equations 2.8 - 2.11.

tion 2.11. Since $\mathbf{X} = (X_1, X_2, X_3, X_4, X_5)$ respects the graph 2.5, this means that graph 2.5 has certain conditional independent properties and these conditional independencies are not necessarily mentioned in equations 2.8 - 2.11. For example, in this graph, X_3 and X_4 are conditionally independent given X_2, which is not mentioned in equations 2.8 - 2.11. This is, what visualisation means is to visualise the joint probability distribution to discover the conditional independencies properties using a graphical model. The concept of generative process is useful for sampling from the joint probability distribution taking into consideration the properties of conditional independencies.

2.4. Conditional independence in a directed graphical model

The unique reason for using a graphical model is to express the conditional dependencies properties of the probability distribution. In section 2.4.1, we will talk about the ***d-separation*** criterion that obtains these conditional properties from the graphical model. Accordingly, we will show the relationship properties between variables through simple graphs.

1. **Tail to tail** relationships: A tail to tail relationship means that the arrows radiate from the node we are conditioning on, see figure 2.6a, where we condition on the node Z. Let X, Y, and Z be random variables, and the distribution of these three variables respect the graph in figure 2.6a; this means that the joint probability of X, Y, and Z factorises as

$$P(X, Y, Z) = P(Z)P(X|Z)P(Y|Z). \tag{2.13}$$

Now let us condition on Z,, i.e., what is the probability of X and Y, given

(known) Z? by Bayes rule

$$\begin{aligned} P(X,Y|Z) &= \frac{P(X,Y,Z)}{P(Z)},\ P(Z)>0 \\ &= \frac{P(Z)P(X|Z)P(Y|Z)}{P(Z)} \text{ by equation 2.13} \\ &= P(X|Z)P(Y|Z). \end{aligned} \tag{2.14}$$

That is, if Z is known then, $X \perp\!\!\!\perp Y|Z$. Other wise, X and Y are dependent.

2. **Head to tail**: A head to tail relationship means that one arrow is going out from the node we are conditioning on, and the other is coming towards it. As above, let X, Y, and Z be random variables, and the distribution of these three variables respect the graph 2.6b; this means that the joint probability of X, Y, and Z factorises as

$$P(X,Y,Z) = P(X)P(Z|X)P(Y|Z). \tag{2.15}$$

Conditioning on Z, the probability of X and Y given Z, by Bayes rule

$$\begin{aligned} P(X,Y|Z) &= \frac{P(X,Y,Z)}{P(Z)},\ P(Z)>0 \\ &= \frac{P(X)P(Z|X)P(Y|Z)}{P(Z)}, \text{ by equation 2.15} \\ &= \frac{P(X,Z)P(Y|Z)}{P(Z)}, \text{ since } P(X)P(Z|X) = P(X,Z) \\ &= \frac{P(X|Z)P(Z)P(Y|Z)}{P(Z)}, \text{ since } P(X,Z) = P(X|Z)P(Z), \text{ by Bayes rule} \\ &= P(X|Z)P(Y|Z). \end{aligned} \tag{2.16}$$

Similarly, if Z is known then, $X \perp\!\!\!\perp Y|Z$. Otherwise, X and Y are dependent.

3. **Head to head** relationship: A head to head relationship mean that the two arrows are coming towards the node we are conditioning on. As above, let X, Y and Z be random variables and the distribution of these three variables respect the graph in figure 2.6c, thus the joint probability of X, Y, and Z factorises as

$$P(X,Y,Z) = P(X)P(Y)P(Z|X,Y). \tag{2.17}$$

Also, in the same way, conditioning on Z, the probability of X and Y are

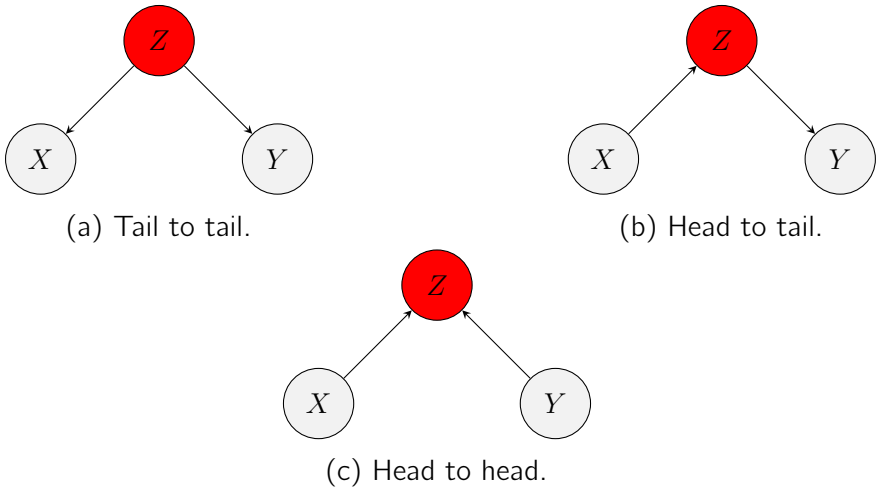
given Z by Bayes rule

$$P(X,Y|Z) = \frac{P(X,Y,Z)}{P(Z)}, \; P(Z) > 0 \tag{2.18}$$

$$= \frac{P(X)P(Y)P(Z|X,Y)}{P(Z)}, \quad \text{by equation 2.17}$$

$$\neq P(X|Z)P(Y|Z). \tag{2.19}$$

Equation 2.19 does not hold in general. In this case, if X and Y are independent, then X is marginally independent (not conditionally) of Y and denoted by $X \perp\!\!\!\perp Y$.



(a) Tail to tail.

(b) Head to tail.

(c) Head to head.

Figure 2.6. – Conditional independencies relationships based on node X.

Type 1 relationships (tail to tail), figure 2.6a, and type 2 relationships (head to tail), figure 2.6b, represent the same dependencies; i.e, X and Y are independent given Z. Type 3 relationships (head to head), figure 2.6c, can be uniquely identified, as X and Y are marginally independent, and in this case the structure, it is known as a ***v-structure***.

Remark 2.4.1. [**Equivalence class**]. The factorisation equation of the joint probability distributions for graphs 2.6a and 2.6b and given by equations 2.13 and 2.15 can be formulated to equivalent equations using Bayes theorem, such cases are known as ***Markov equivalent structures***, and each set of equivalent structures forms an ***equivalence class.***

2.4.1. *d-separation criterion*

d-separation criterion (Pearl (1988)) is an abbreviation for a *directed separation*, and as mentioned above in section 2.4, the ***d-separation*** criterion allows us

to read the conditional properties from the graphical models for a probability distribution. Thus, using the ***d-separation*** criterion, we can determine whether two sets of random variables are conditionally independent or not given a third set. The *d-separation* properties are a wide generalisation of the relationships mentioned in section 2.4.

The concept of *d-separation* is related to an important definition: the **path** and the **blocked path**. We already defined the concept of path in definition 2.2.4, thus the definition of **blocked path** remains.

Definition 2.4.1. [Blocked path].
A path between two vertices in graph $G =< \mathbf{X}, \mathbf{E} >$ is blocked with respect to a subset $\mathbf{Z} \subseteq \mathbf{X}$ if it passes through vertex $W \in \mathbf{X}$ such that either one of two properties hold the following:

1. The arrow is *tail to tail* or *head to tail* and $W \in \mathbf{Z}$.
2. The arrow is *head to head* and $W \notin \mathbf{Z}$, and none of the descendants of W belong to Z.

After illustrating these concepts, we can define the **d-separation** concept as follows:

Definition 2.4.2. [d-separation]. If $\mathbf{X}, \mathbf{Y}$, and $\mathbf{Z}$ are three disjoint subsets of vertices in a DAG G, then $\mathbf{X}$ and $\mathbf{Y}$ are *d-separated* by $\mathbf{Z}$, if all the paths from a vertex in $\mathbf{X}$ to a vertex in $\mathbf{Y}$ are blocked with respect to $\mathbf{Z}$.

Based on the **d-separation** definition, we can obtain the concept of conditional independencies relations from the probabilistic graphical models (PGMs) in the following theorem:

Theorem 2.4.1. ***[Conditional independence in PGMs].***
Let G be a PGM and $\mathbf{X}, \mathbf{Y}$, and $\mathbf{Z}$ be three disjoint subsets of vertices in a DAG G, if $\mathbf{X}$ and $\mathbf{Y}$ are d-separated by $\mathbf{Z}$, then $\mathbf{X}$ is conditionally independent of $\mathbf{Y}$ given $\mathbf{Z}$ and denoted by $\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z}$.

2.5. Markov Property

Recall that for any sequence of random variables $X_1, X_2, ..., X_p, ...$, it is a ***Markovian chain*** if it holds for any $p \in \mathbb{N}$

$$P(X_{p+1}|X_1, ..., X_p) = P(X_{p+1}|X_p), \tag{2.20}$$

That is, X_{p+1} is conditionally independent of $X_1, X_2, ..., X_{p-1}$, given X_p and written symbolically

$$X_{p+1} \perp\!\!\!\perp (X_1, ..., X_{p-1}) | X_p, \tag{2.21}$$

or

$$X_{p+1} \perp\!\!\!\perp_P (X_1, ..., X_{p-1}) | X_p \tag{2.22}$$

if the relation related to a given probability distribution P (figure 2.7).

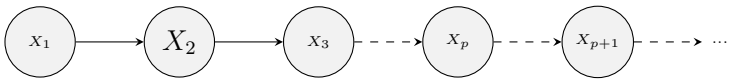


Figure 2.7. – Markovian chain representation.

The ***Markovian property*** (Korb and Nicholson (2010)) allows us to factorise the joint probability distribution over the set of variables $\mathbf{X} = (X_1, ..., X_p)$ as follows:

$$\begin{aligned} P(X_1, ..., X_p) &= P(X_p | X_1, ..., X_{p-1}) P(X_1, ..., X_{p-1}) \quad \text{by Bayes rule} && (2.23) \\ &= P(X_p | X_{p-1}) P(X_{p-1} | X_1, ..., X_{p-2}) P(X_1, ..., X_{p-2}) \quad \text{by Eq. 2.20} \\ &= P(X_p | X_{p-1}) P(X_{p-1} | X_{p-2}) P(X_1, ..., X_{p-2}). && (2.24) \end{aligned}$$

If we continue the procedure for another $p - 1$ iteration in equation 2.24, we obtain

$$P(X_1, ..., X_p) = P(X_1) P(X_2 | X_1) P(X_3 | X_2) ... P(X_{p-1} | X_{p-2}) P(X_p | X_{p-1}). \tag{2.25}$$

The idea behind ***Markovian property*** is that ***the future is independent of the past, given the present***. In static Bayesian networks, the data is not temporal (related to time); so the **present** here refers to the parents nodes; the **future** refers to the descendants nodes, and the **past** refers to the nondescendant nodes, i.e, ***each variable*** $\mathbf{X_i}$ ***is independent of its nondescendants given its parents in*** G. In chapter four and five, we will talk about Bayesian networks for temporal data known as **dynamic Bayesian networks, (DBN)**, (Friedman et al. (1998)).

The Markovian properties on the Bayesian networks follows directly from the concept of *d-separation* and allows us to represent the joint probability distribution of the set of variables $\mathbf{X} = (X_1, ..., X_p)$ as the product of local conditional probability distributions. The Bayesian networks $BN =< G, \mathbf{\Theta} >$ defining a unique joint probability distribution over $\mathbf{X}$ is given by

$$\begin{cases} P_{BN}(X_1, ..., X_p) & = \prod_{j=1}^{p} P(X_j|\pi(X_j)) = \prod_{j=1}^{p} \theta_{X_j|\pi(X_j)}, \quad \text{for discrete data.} \\ f_{BN}(X_1, ..., X_p) & = \prod_{j=1}^{p} f(X_j|\pi(X_j)) = \prod_{j=1}^{p} \theta_{X_j|\pi(X_j)}, \quad \text{for continuous data.} \end{cases} \tag{2.26}$$

where $\pi(X_j)$ is the parents set of X_j, and $\boldsymbol{\Theta} = \{\theta_{X_j|\pi(X_j)}\}$ is the set of parameters qualifying the network.

Definition 2.5.1. [**Markov blanket, (Pearl (1988))**]. The Markov blanket of a node X in a Bayesian network is the set of nodes composed of X's parents, its children, and the children's other parents.

2.6. Learning Bayesian networks

Learning Bayesian networks model (Heckerman et al. (1995); Lam and Bacchus (1994); Geiger and Heckerman (1996); Koller and Friedman (2009); Korb and Nicholson (2010); Cooper and Herskovits (1992)) goes through two procedures: ***learning structure*** and ***learning parameter***. In *learning structure*, we are looking for the structure that gives the most accurate picture of the joint probability distribution over the set of random variables $\mathbf{X}$. Several methods are proposed for searching the best structure if it is unknown. *Learning parameters* refer to computing the conditional probabilities based on the structure given by the first step. More obviously, as we have discussed, a graphical model can be used to answer probabilistic inference queries by finding the closest underlying joint probability distribution for the training sample data, $\mathcal{D} = \{(y_i, x_{i1}), (y_i, x_{i2}), ..., (y_i, x_{ip})\}$, $i = 1, ..., n$. Additionally, we may want to solve a simple classification task and answer queries in the form $P(Y|\mathbf{X})$ for any new instance $\mathbf{x}$ using maximum a posterior probability (**MAP**) to perform the predictions, where Y is the discrete or factor variable.

2.6.1. Learning parameters

In this section, we will assume that the structure is known, and the data is complete; that is, there are no missing values or latent variables. There are two main approaches for learning parameters in Bayesian networks: learning parameters using maximum likelihood estimation (**MLE**) and learning parameters using Bayesian estimation.

2.6.1.1. Maximum Likelihood Estimation (MLE)

Estimating parameters in Bayesian networks is to find the conditional probabilities $\theta_j = P(X_j|\pi(X_j))$ that qualifies these networks for each possible values of X_j and $\pi(X_j)$. In Bayesian networks, estimating parameters using maximum likelihood estimation MLE is finding the parameter θ such that

$$L(\hat{\theta}; \mathbf{X}) = \underset{\theta \in \mathbf{\Theta}}{max}\ L(\theta; \mathbf{X}). \tag{2.27}$$

In next sections, we will study estimating the parameters in Bayesian networks for discrete and continuous data.

2.6.1.2. MLE for discrete Bayesian networks

Let $\mathbf{X} = (X_1, ..., X_p)$ be a discrete random vector; we specify the joint probability distribution over these variables to be a multinomial distribution, assigning a probability to each combination of levels of the variables. In the context of Bayesian networks, this joint distribution is called the *global distribution*. Using the global distribution directly is difficult because the number of its parameters is very high. But, we can use the information encoded in the DAG to divide the global distribution into a set of smaller local distributions, one for each variable. Recall that arcs represent direct dependencies; variables that are not linked by an arc are conditionally independent. Each variable depends only on its parents; its distribution is univariate and has a small number of parameters. Even the set of all the local distributions has, overall, fewer parameters than the global distribution (Scutari and Denis (2014); Koller and Friedman (2009); Neapolitan (2003)).

If $\mathbf{X}$ has $(\sim)$, the multinomial distribution is as follows:

$$\mathbf{X} \sim multinomial(n, \theta_j), j = 1, ..., p. \tag{2.28}$$

Then the maximum likelihood estimation aims to maximise the likelihood (or log-likelihood) function over $\mathbf{X}$, which is

$$f(\mathbf{X}) = P(\mathbf{X} = \mathbf{x}, \mathbf{\Theta}) = \frac{n!}{x_1!x_2!...x_p!}\theta_1^{x_1}\theta_2^{x_2}...\theta_p^{x_p}, \tag{2.29}$$

where $x_j = \{0, 1, 2, ..., n\}$ is the number of successes of the p^{th} outcome in n trails; $\sum_{j=1}^{p} x_j = n$, θ_j is the probability of success of the p^{th} outcome, $\sum_{j=1}^{p} \theta_j = 1$, and $\theta_j \in [0, 1]$.

Consider figure 2.8, the joint probability distribution of the graph is given by

$$P(X_1, X_2, X_3, X_4, X_5) = \underbrace{P(X_1)}_{\theta_1}\underbrace{P(X_2|X_1)}_{\theta_2}\underbrace{P(X_3)}_{\theta_3}\underbrace{P(X_4|X_1, X_3)}_{\theta_4}\underbrace{P(X_5)}_{\theta_5} \tag{2.30}$$

To find the parameters estimation $\theta_j = P(X_j|\pi(X_j))$ using MLE between two nodes in Bayesian networks, there are two cases: when $\pi(X_j) = \phi$, that is when there are no incoming arcs coming towards X_j, as you can see in the above equation. Finding the MLE for $\theta_1 = P(X_1)$, $\theta_3 = P(X_3)$, and $\theta_5 = P(X_5)$ is carried out using MLE for their joint probability distribution, which is multinomial for each one (**Case one**). In **case two**, we will find the MLE for the variables when they have parents; for example $\theta_2 = P(X_2|X_1)$ and $\theta_4 = P(X_4|X_1, X_3)$.

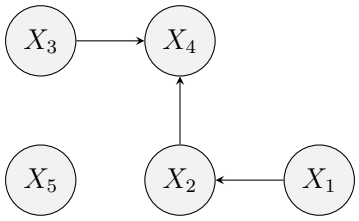


Figure 2.8. – Bayesian network example.

Case one: (MLE for $P(X_j|\pi(X_j))$, when $\pi(X_j) = \phi$).
In this case, we will find the MLE of each variable X_j separately. As X_j is a discrete variable, and it may take values from a finite set $X_j = \{1, ..., J\}$ and $J \in \mathbb{N}$, we assume $\mathbf{X} = (X_1, ..., X_p)$ has multinomial distribution

$$f(x_1, ..., x_p, \theta_1, ..., \theta_k) = \frac{n!}{x_1!, ...x_p!}\theta_1^{x_1}...\theta_p^{x_p} \tag{2.31}$$

$$= C(x)\prod_{j=1}^{p}\theta_j^{x_j}, \text{ where } C(x) = \frac{n!}{x_1!, ...x_p!}. \tag{2.32}$$

Then, the multinomial likelihood function for n samples is given by

$$L(\theta_1, ..., \theta_p) = C(x)\prod_{j=1}^{p}\theta_j^{x_j} \tag{2.33}$$

and the log-likelihood function is given by

$$\ell(\theta_1, ..., \theta_p) = \log L(\theta_1, ..., \theta_p),$$

$$= \log C(x) + \sum_{j=1}^{p} x_j \log\ \theta_j \tag{2.34}$$

Now, since $\sum_{j=1}^{p} x_j = n$ and $\sum_{j=1}^{p} \theta_j = 1$, therefore $x_p = n - \sum_{j=1}^{p-1} x_j$ and $\theta_p = 1 - \sum_{j=1}^{p-1} \theta_j$. So that, we have

$$\begin{aligned}\ell(\theta_1, ..., \theta_{p-1}) &= \log L(\theta_1, ..., \theta_{p-1}) \\ &= x_1 \log \theta_1 + ... + x_j \log \theta_j + ... + x_{p-1} \log \theta_{p-1} + x_p \log(1 - \sum_{j=1}^{p-1} \theta_j).\end{aligned} \tag{2.35}$$

The differentiation of multinomial log-likelihood equations are

$$\frac{\partial \ell(\theta_1, ..., \theta_{p-1})}{\partial \theta_j} = \frac{x_j}{\theta_j} - \frac{x_p}{\theta_p} = 0, \qquad j = 1, ..., p-1.$$

Hence, the MLE of $\theta_1, ..., \theta_{p-1}$ satisfies the equation

$$\frac{x_1}{\hat{\theta}_1} = \frac{x_p}{\hat{\theta}_p}, \ \frac{x_2}{\hat{\theta}_2} = \frac{x_p}{\hat{\theta}_p}, \ ..., \ \frac{x_{p-1}}{\hat{\theta}_{p-1}} = \frac{x_p}{\hat{\theta}_p}.$$

This is equivalent to

$$\frac{x_1}{x_p} = \frac{\hat{\theta}_1}{\hat{\theta}_p}, \ \frac{x_2}{x_p} = \frac{\hat{\theta}_2}{\hat{\theta}_p}, \ ..., \ \frac{x_{p-1}}{x_p} = \frac{\hat{\theta}_{p-1}}{\hat{\theta}_p}. \tag{2.36}$$

Using

$$\sum_{j=1}^{p} \hat{\theta}_j = 1 = \sum_{j=1}^{p} \frac{x_j}{x_p} \hat{\theta}_p = \frac{\hat{\theta}_p}{x_p} \sum_{j=1}^{p} x_j = \frac{n\hat{\theta}_p}{x_p}. \tag{2.37}$$

Hence, the MLE of $\theta_1, ..., \theta_p$ is given by

$$\hat{\theta}_1 = \frac{x_1}{n}, ..., \hat{\theta}_p = \frac{x_p}{n}. \tag{2.38}$$

For clarity, if the random variable $X_1 = (x_1, ..., x_n) \sim multinomial(n, \theta_1)$, then the MLE for θ_1 is

$$\hat{\theta}_1 = \frac{x_1}{n} = \frac{\text{Count}(X_1 = x_1)}{\text{sample size}}. \tag{2.39}$$

Case Two: (MLE for $P(X_j|\pi(X_j))$, where $\pi(X_j) \neq \phi$).
In this case, we want to find the estimation of $P(X_j|\pi(X_j))$. The log-

likelihood function for joint probability distribution of the graph G is

$$\ell(\Theta) = \log \prod_{i=1}^{n} \prod_{j=1}^{p} P(X_j = x_{ij} | \pi(X_j) = \pi_{ij}) \tag{2.40}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{p} \log P(X_j = x_{ij} | \pi(X_j) = \pi_{ij}). \tag{2.41}$$

We can separate this problem into n sub-tasks to maximise

$$L_i = \sum_{j=1}^{p} P(X_j = x_{ij} | \pi(X_j) = \pi_{ij}), \quad i = 1, ..., n. \tag{2.42}$$

Now, group the terms that have the same outcomes of X_j, and let $\mathbf{x}$ be all the possible values of X_j, and $\boldsymbol{\pi}$ be the set of all possible values of parents of X_j; also, let $Count(\mathbf{x}, \boldsymbol{\pi})$ be the number of examples $x_{ij} \in \mathbf{x}$ and $\pi_{ij} \in \boldsymbol{\pi}$. Note that

$$n = \sum_{\mathbf{x}} \sum_{\boldsymbol{\pi}} Count(x_{ij}, \pi_{\mathbf{ij}}) \tag{2.43}$$

and we can write

$$M_i = \sum_{\mathbf{x}} \sum_{\boldsymbol{\pi}} \log P(X_j = x_{ij} | \pi(X_j) = \pi_{ij}), \quad x_{ij} \in \mathbf{x}, \ \pi_{ij} \in \boldsymbol{\pi}. \tag{2.44}$$

Note that $\sum_{\mathbf{x}} P(X_j = x_{ij} | \pi(X_j)) = 1$, so our problem becomes maximised

$$\sum_{\mathbf{x}} C_{\mathbf{x}} \log w_{\mathbf{x}} \tag{2.45}$$

subject to

$$w_{\mathbf{x}} \geq 0 \text{ and } \sum_{\mathbf{x}} w_{\mathbf{x}} = 1, \tag{2.46}$$

where $C_{\mathbf{x}} = Count(\mathbf{x}, \boldsymbol{\pi})$ and $w_x = P(X_j = x_{ij} | \pi(X_j) = \pi_{ij})$.
This problem can be solved using Lagrange multipliers by differentiating this equation with respect to $w_{\mathbf{x}}$

$$M_i = \sum_{\mathbf{x}} C_{\mathbf{x}} \log w_{\mathbf{x}} + \lambda(1 - \sum_{\mathbf{x}} w_{\mathbf{x}}), \quad \lambda \in [0, 1]. \tag{2.47}$$

The solution will be

$$w_{\mathbf{x}} = \frac{C_{\mathbf{x}}}{\sum_{\mathbf{x}} C_{\mathbf{x}}}, \tag{2.48}$$

i.e.

$$P(X_j = x_{ij}|\pi(X_j) = \pi_{ij}) = \frac{Count(X_j = x_{ij}, \pi(X_j) = \pi_{ij})}{\sum_{\mathbf{x}} Count(X_j = x_{ij}, \pi(X_j) = \pi_{ij})} \tag{2.49}$$

$$= \frac{Count(X_j = x_{ij}, \pi(X_j) = \pi_{ij})}{Count(\pi(X_j) = \pi_{ij})}. \tag{2.50}$$

For more details regarding cases one and two in a discrete case, see (Koller and Friedman (2009); Johnson and Wichern (2007); Santafé (2010)).

2.6.1.3. Learning parameters of Gaussian Bayesian networks

To specify completely the joint probability distribution of the continuous variables in the Bayesian network, the multivariate normal distributions are considered in this dissertation. If we are modelling p variables, we must specify p means (μ), p variances (σ^2) and $\frac{1}{2}p(p-1)$ correlation coefficients. Furthermore, the correlation coefficients must be such that the resulting correlation matrix is non-negative definite. But, in the context of BNs, we only need to specify the local distribution of each node conditional on the values of its parents, without worrying about the positive definiteness of the correlation matrix of the global distribution (Scutari and Denis (2014)).

Let us consider that a single random variable X has univariate normal distribution such that

$$X = (x_1, ..., x_n) \sim \mathcal{N}(\mu, \sigma^2), \quad x_i \in \mathcal{R}, -\infty < \mu < \infty, 0 < \sigma^2 < \infty. \tag{2.51}$$

The variable X in this case has two parameters that must be estimated: μ and σ^2.

The likelihood function over X is given by

$$L(\mu, \sigma^2) = f((x_1, ..., x_n); \mu, \sigma^2) = (2\pi)^{-\frac{n}{2}} \sigma^{-2n} \exp - \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{\sigma^2}$$

, and the log-likelihood function

$$\ell(\mu, \sigma^2) = \log L(\mu, \sigma^2)$$

$$= -\frac{n}{2} \log 2\pi - n \log \sigma^2 - \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{\sigma^2} \tag{2.52}$$

Now, differentiate equation 2.52 with respect to μ and then with respect to σ^2

and by equating the derivatives to zero, we get

$$\hat{\mu} = \frac{\sum_{i=1}^{n} x_i}{n} = \bar{x} \quad \text{and} \quad \hat{\sigma}^2 = \frac{\sum_{i=1}^{n} (x_i - \hat{\mu})^2}{n}.$$

In our example regarding figure 2.8, if the distribution of G is

$$f(X_1, X_2, X_3, X_4, X_5) = \underbrace{f(X_1)}_{\theta_1} \underbrace{f(X_2|X_1)}_{\theta_2} \underbrace{f(X_3)}_{\theta_3} \underbrace{f(X_4|X_1, X_3)}_{\theta_4} \underbrace{f(X_5)}_{\theta_5}, \tag{2.53}$$

the parameters estimation for the first, third, and fifth terms are as in the previous example for a single random variable. But for the second and fourth terms, we need to know the distribution of $X_2|X_1$ and $X_4|(X_1, X_3)$.

For example, consider finding the parameters estimation for the local distribution of X_1 and X_2 in figure 2.8 with a probability density distribution $f(X_2|X_1)$. The joint probability function for X_1 and X_2 is bivariate normal, i.e,

$$X_1 \sim \mathcal{N}(\mu_1, \sigma_{11}^2), \tag{2.54}$$
$$X_2 \sim \mathcal{N}(\mu_2, \sigma_{22}^2). \tag{2.55}$$

Thus,

$$\begin{aligned} f(x_1, x_2) = & \frac{1}{2\pi\sqrt{\sigma_{11}^2 \sigma_{22}^2 (1 - \rho_{12}^2)}} \\ & \times \exp\{-\frac{1}{2(1 - \rho_{12}^2)} \left(\frac{x_1 - \mu_1}{\sigma_{11}^2}\right)^2 + \left(\frac{x_2 - \mu_2}{\sigma_{22}^2}\right)^2 \\ & - 2\rho_{12} \left(\frac{x_1 - \mu_1}{\sigma_{11}^2}\right)^2 \left(\frac{x_2 - \mu_2}{\sigma_{22}^2}\right)^2\}, \end{aligned} \tag{2.56}$$

or in another form

$$f(x_1, x_2) = \frac{1}{2\pi|\Sigma|^{1/2}} e^{-(\mathbf{x}-\mu)\mathbf{\Sigma^{-1}}(\mathbf{x}-\mu)/\mathbf{2}} \tag{2.57}$$

,
where

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 \end{bmatrix}, \ \mathbf{\Sigma^{-1}} = \frac{1}{\sigma_{11}^2 \sigma_{22}^2 - \sigma_{12}^4} \begin{bmatrix} \sigma_{22}^2 & -\sigma_{12}^2 \\ -\sigma_{12}^2 & \sigma_{11}^2 \end{bmatrix}, \ \mu = \begin{bmatrix} \mu_1 = E(X_1) \\ \mu_2 = E(X_2) \end{bmatrix}, \tag{2.58}$$

such that the correlation between the two variables is $Corr(X_1, X_2) = \rho_{12} = \sigma_{12}/(\sigma_{11}\sigma_{22})$; the means are $\mu_1 = E(X_1)$, $\mu_2 = E(X_2)$; the variances are $\sigma_{11}^2 =$

$var(X_1)$, $\sigma_{22}^2 = var(X_2)$, and the covariance is $Cov(X_1, X_2) = \sigma_{12}$. But the conditional distribution of $X_1|X_2$ is

$$f(x_2|x_1) = \frac{f(x_1, x_2)}{f(x_1)}, \tag{2.59}$$

where $f(x_1, x_2)$ is the bivariate normal density, and $f(x_1)$ is the marginal distribution of X_1, which is a univariate normal distribution. Dividing the joint density of X_1 and X_2, as shown in equation 2.56, by the marginal density of X_1 and eliminating the terms give the conditional density

$$\begin{aligned} f(x_2|x_1) &= \frac{f(x_1, x_2)}{f(x_1)} \\ &= \frac{1}{2\pi\sigma_{22}^2(1-\rho_{12})} e^{-[x_2-\mu_2-(\sigma_{12}/\sigma_{11}^2)(x_1-\mu_1)]^2/2\sigma_{22}^2(1-\rho_{12}^2)}. \end{aligned} \tag{2.60}$$

$$-\infty < x_2 < \infty$$

Thus, the distribution of the local graph $X_1 \rightarrow X_2$ is

$$X_2|X_1 \sim \mathcal{N}(\mu_2 + \frac{\sigma_{12}}{\sigma_{11}^2}(X_1 - \mu_1), \sigma_{22}^2(1-\rho_{12}^2)). \tag{2.61}$$

The estimation of the population mean is the sample mean $\bar{X}$, and the estimation of the population variance is the sample variance S; so learning parameters of the local graph $X_2 \rightarrow X_1$ is given by

$$\hat{\mu}_{X_2|X_1} = \bar{X}_2 + \frac{S_{12}}{S_{11}^2}(X_1 - \bar{X}_1) \text{ and } \hat{\sigma}_{X_2|X_1}^2 = S_{22}^2(1-\hat{\rho}_{12}^2), \text{ where, } \rho_{12} = \frac{\sigma_{12}}{\sigma_{11}\sigma_{22}}. \tag{2.62}$$

Note that $\Sigma_{22} - \Sigma_{12}\Sigma_{11}^{-1}\Sigma_{12} = \sigma_{22}^2 - \sigma_{12}/\sigma_{11}^2$ and $\Sigma_{12}\Sigma_{11}^{-1} = \sigma_{12}/\sigma_{11}^2$, thus you can rewrite equation 2.62 as

$$X_2|X_1 \sim \mathcal{N}(\mu_2 + \Sigma_{12}\Sigma_{11}^{-1}(x_1 - \mu_1), \Sigma_{22}^2(1-\rho_{12}^2)) \tag{2.63}$$

. Moreover, note from equation 2.62

$$\begin{aligned} \hat{\mu}_{X_2|X_1} &= \bar{X}_2 + \frac{S_{12}}{S_{11}^2}(X_1 - \bar{X}_1) \\ &= \underbrace{\bar{X}_2 - \frac{S_{12}}{S_{11}^2}\bar{X}_1}_{a_0} + \underbrace{\frac{S_{12}}{S_{11}^2}}_{a_1} X_1 \\ &= a_0 + a_1 X_1, \end{aligned} \tag{2.64}$$

we get the linear regression equation of fitting X_2 over X_1, where a_0, a_1 are the

regression coefficients.

In general, the ***p-dimensional normal density*** $\mathcal{N}_p(\mu, \mathbf{\Sigma})$ has the form

$$f(x_1, ..., x_p) = \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}|^{1/2}} e^{-(\mathbf{x}-\mu)\mathbf{\Sigma^{-1}}(\mathbf{x}-\mu)/\mathbf{2}} \tag{2.65}$$

, where $-\infty < x_j < \infty$, $j = 1, ..., p$, and this is also called ***multivariate normal density***. In this case, **all the conditional distributions have multivariate normal distributions.** Thus, for a random vector $\mathbf{X} = (X_1, ..., X_p)$ and subset $\mathbf{Z}$ of $\mathbf{X}$, the conditional distribution of $X_j|\mathbf{Z} = (Z_1, ..., Z_k)\backslash\{X_j\}$ is a Gaussian distribution given by

$$X_i|\mathbf{Z} = \mathbf{z} \sim \mathcal{N}(\mu_X + (z - \mu_{\mathbf{Z}})A_j, \sigma_{XX} - \Sigma_{\mathbf{Z}X}^T \Sigma_{\mathbf{ZZ}}^{-1} \Sigma_{\mathbf{Z}X}), \tag{2.66}$$

with a conditional mean

$$\mu_X + (z - \mu_{\mathbf{Z}})A_j, \quad \text{where} \quad A_j = \Sigma_{\mathbf{ZZ}}^{-1} \Sigma_{\mathbf{Z}X}, \tag{2.67}$$

and a conditional covariance

$$\mathbf{\Sigma} = \sigma_{XX} - \Sigma_{\mathbf{Z}X}^T \Sigma_{\mathbf{ZZ}}^{-1} \Sigma_{\mathbf{Z}X}, \tag{2.68}$$

which does not depend upon the value(s) of the conditioning variable(s), i.e, if X is a variable and has k parents $\mathbf{Z} = \{Z_1, ..., Z_k\}$ as in figure 2.9.

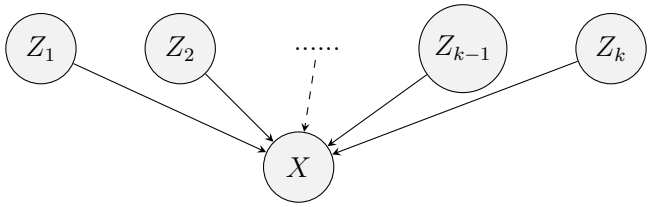


Figure 2.9. – k parents for node X.

The joint distribution that represents the local graph 2.9 is given by

$$f(X, Z_1, .., Z_k) = f(X|Z_1, .., Z_{k-1}, Z_k), \tag{2.69}$$

where

$$X|Z_1, .., Z_{k-1}, Z_k \sim \mathcal{N}(a_0 + a_1 Z_1 + ... + a_k Z_k,\ \sigma_{XX} - \Sigma_{\mathbf{Z}X}^T \Sigma_{\mathbf{ZZ}}^{-1} \Sigma_{\mathbf{Z}X}) \tag{2.70}$$

i.e. the parameters estimation are

$$\hat{\mu} = \hat{a}_0 + \hat{a}_1 Z_1 + ... + \hat{a}_k Z_k, \tag{2.71}$$

$$\hat{\Sigma} = \sigma_{XX} - \Sigma_{\mathbf{Z}X}^T \Sigma_{\mathbf{ZZ}}^{-1} \Sigma_{\mathbf{Z}X}, \tag{2.72}$$

where the a's can be computed by fitting the linear regression of X over $\mathbf{Z} = (Z_1, ..., Z_k)$ and given by this equation

$$\hat{A} = (\mathbf{Z}^T \mathbf{Z})^{-1} \mathbf{Z}^T X \tag{2.73}$$

For this section, see (Johnson and Wichern (2007)) for more details.

2.6.1.4. Problems related to the MLE

In the previous sections, we discussed the point estimation method, namely MLE, that characterises the population under consideration as fixed unknown constants. This approach is called **classical approach** to the problem of statistical inference or **frequentist inference**. The likelihood equations need to be calculated for a given distribution and estimation problem. The mathematics is often non-trivial. Also, assume that a coin is tossed 10 times and comes out tails 10 times. The estimated probability of heads equals zero, although based on our prior knowledge, the probability of head is equal to $\frac{1}{2}$. That is why Bayesian estimation is introduced here. (Johnson and Wichern (2007); Friedman et al. (1997); Korb and Nicholson (2010)).

2.6.1.5. Bayesian estimation

In this approach, we encode our prior knowledge about θ with a probability distribution called *prior distribution* and is denoted by $P(\theta)$; this distribution represents how we are a priori likely to believe the different choices of parameters. Once we quantify our knowledge (or lack thereof) about the possible values of θ, we can create a joint distribution over the parameters θ and the data cases that we are about to observe. This joint distribution captures our assumptions about the experiment (Koller and Friedman (2009)).

The network structure implies that the joint distribution of a particular data set and θ factorise as

$$P(\mathbf{X}; \theta) = P(X_1, ..., X_p|\theta) P(\theta) \tag{2.74}$$

Note that the term $P(X_1, ..., X_p|\theta)$ is the likelihood function $L(\theta; \mathbf{X})$. The network specifies a joint probability model over parameters and data. There are several ways in which we can use this network. Most obviously, we can take an observed

data set and use it to instantiate the values of $x_1, ..., x_n$; we can then compute the **posterior distribution** over θ

$$P(\theta|x_1, ..., x_n) = \frac{P(x_1, ..., x_n|\theta)P(\theta)}{P(x_1, ..., x_n)}. \tag{2.75}$$

In this posterior, the first term in the numerator is **the likelihood**; the second is **the prior** over parameters, and the denominator is a **normalizing factor**. We see that the posterior is (proportional to) a product of the likelihood and the prior

$$P(\theta|X) \propto P(X|\theta)P(\theta) = C^* P(X|\theta)P(\theta), \tag{2.76}$$

where C^* is a normalised factor so that it will be a proper density function.

For example, let $X|\mathbf{\Theta} = \theta \sim Ber(\theta)$ and $\theta \sim Beta(\alpha, \beta)$, then the probability density function of $X|\mathbf{\Theta} = \theta$ is given by

$$P(X = x|\theta) = \theta^x(1-\theta)^{1-x} \tag{2.77}$$

where $\theta \in (0, 1)$. Note that $E(X|\theta) = \theta$, $x \in \{1, ..., J\}$, and $var(X|\theta) = \theta(1-\theta)$. The probability density function $\mathbf{\Theta} = \theta$ is given by

$$P(\theta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{Beta(\alpha, \beta)}, \text{ where } Beta(\alpha, \beta) = \frac{\Gamma\alpha\Gamma\beta}{\Gamma(\alpha+\beta)} \tag{2.78}$$

$\theta \in (0, 1)$ and α, $\beta > 0$ with $E(\theta) = \frac{\alpha}{\alpha+\beta}$ and $var(\theta) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$. Then the joint probability density function of X and $\Theta = \theta$ is

$$\begin{aligned} P(x, \theta) = P(\theta)P(x|\theta) &= \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{Beta(\alpha, \beta)} \times \theta^x(1-\theta)^{1-x} \\ &= \frac{\theta^{\alpha+x-1}(1-\theta)^{\beta+(1-x)-1}}{Beta(\alpha, \beta)}. \end{aligned} \tag{2.79}$$

Since Θ is a continuous random variable, the marginal or predictive probability density of X is

$$\begin{aligned} P(X) &= \int_\Theta P(x, \theta)d\theta = \int_\Theta P(\theta)f(x|\theta)d\theta \\ &= \int_0^1 \frac{\theta^{\alpha+x-1}(1-\theta)^{\beta+(1-x)-1}}{Beta(\alpha, \beta)}d\theta \end{aligned} \tag{2.80}$$

by multiplying and dividing by $\frac{\Gamma\alpha^*\Gamma\beta^*}{\Gamma(\alpha^*+\beta^*)}$, we get

$$P(X) = \frac{\Gamma\alpha^*\Gamma\beta^*\Gamma(\alpha+\beta)}{\Gamma(\alpha^*+\beta^*)\Gamma\alpha\Gamma\beta} = \frac{Beta(\alpha^*,\beta^*)}{Beta(\alpha,\beta)} \tag{2.81}$$

,

where $Beta(\alpha^*,\beta^*) = \frac{\Gamma\alpha^*\Gamma\beta^*}{\Gamma(\alpha^*+\beta^*)}$.

Thus, the posterior probability distribution is given by

$$\begin{aligned} P(\theta|X) &= \frac{P(X,\theta)}{P(X)} = \frac{P(\theta)f(X|\theta)}{\int\limits_{\Theta} f(x|\theta)d\theta} \\ &= Beta(\alpha^* = \alpha + x, \beta^* = \beta + (1-x)). \end{aligned} \tag{2.82}$$

This is a beta distribution with **hyper parameters** $\alpha^* = \alpha+x$ and $\beta^* = \beta+(1-x)$.

This result presents a property of the Beta distribution: If the prior is a Beta distribution, then the posterior distribution (the prior conditioned on the evidence) is also a Beta distribution. In this case, we say that the Beta **conjugate prior** distribution (definition 2.6.1) is conjugate to the Bernoulli likelihood function.

Definition 2.6.1. [Conjugate prior].
A family of priors $P(\theta : \alpha)$ is conjugate to a particular model $P(\zeta|\theta)$ if for any possible data set $\mathcal{D}$ of Independent and Identically Distributed (IID) samples from $P(\zeta|\theta)$, and any choice of legal hyper-parameters α for the prior over θ, there are hyper-parameters α^* that describe the posterior. That is,

$$P(\theta : \alpha^*) \propto P(\mathcal{D}|\theta)P(\theta : \alpha). \tag{2.83}$$

The main philosophical difference between the Bayesian approach and MLE approach is in the use of the posterior. Instead of selecting from the posterior a single value for the parameter θ, we use it, in its entirety, for predicting the probability over a new instance. An immediate consequence is that we can compute the probabilities over the new instance x_{n+1} from the above example by

$$\begin{aligned} P(x_{n+1}|x_1,...,x_n) &= \int P(x_{n+1}|x_1,...,x_n,\theta)P(\theta|x_1,...,x_n)d\theta \\ &= \int P(x_{n+1}|\theta)P(\theta|x_1,...,x_n)d\theta \\ &= E_{P(\theta|\mathbf{X})}(x_{n+1}|\theta) \\ &= \frac{\alpha^*}{\alpha^*+\beta^*}. \end{aligned} \tag{2.84}$$

where, in the second step, we use the fact that the instances are independent given θ, $\alpha^* = \alpha + x_{n+1}$ and $\beta^* = \beta + (1 - x_{n+1})$. This prediction in equation 2.84, called the ***Bayesian estimator***, is quite similar to the MLE prediction except that it adds one "imaginary" sample to each count. Clearly, as the number of samples

grows, the Bayesian estimator and the MLE estimator converge to the same value (Koller and Friedman (2009)).

Definition 2.6.2. [**Bayes estimate**] Suppose we have observations over previous instances from $\mathcal{D}$, and suppose that we are about to sample a new instance x_{new}, then the Bayesian estimator is the posterior distribution over a new example:

$$\begin{aligned} P(x_{new}|\mathcal{D}) &= \int P(x_{new}|\mathcal{D},\theta)P(\theta|\mathcal{D})d\theta \\ &= \int P(x_{new}|\theta)P(\theta|\mathcal{D})d\theta \\ &= E_{P(\theta|\mathbf{X})}(x_{new}|\theta). \end{aligned} \tag{2.85}$$

Thus, our prediction is the average overall parameters according to the posterior.

In discrete Bayesian networks, let training data $\mathcal{D} = \{X_1, ..., X_p\}$ come from multinomial distribution

$$X_1, ..., X_p|\theta \sim multinomial(n; \theta_1, ..., \theta_p) \tag{2.86}$$

then

$$P(X_1 = x_1, ..., X_p = x_p|\theta_1, ..., \theta_p) = \frac{\Gamma(n+1)}{\prod_{j=1}^{p}\Gamma(x_j+1)}\prod_{j=1}^{p}\theta_j^{x_j} \tag{2.87}$$

, where $\sum_{j=1}^{p}\theta_j = 1$ and $\sum_{j=1}^{p}x_j = n$. Based on definition 2.6.1, since the posterior is proportional to the prior multiplied by the likelihood, it is suitable to choose the prior from the same family of likelihood. ***Dirichlet distribution*** generalises the beta distribution. A Dirichlet distribution with parameters $\alpha_1, ..., \alpha_p$ and denoted by $Drich(\alpha_1, ..., \alpha_p)$ is given by

$$P(\theta) = f(\theta_1, ..., \theta_p; \alpha_1, ..., \alpha_p) = \frac{\Gamma\left(\sum_{j=1}^{p}\alpha_j\right)}{\prod_{j=1}^{p}\Gamma(\alpha_j)}\prod_{j=1}^{p}\theta_j^{\alpha_j-1} = \frac{1}{Beta(\alpha)}\prod_{j=1}^{p}\theta_j^{\alpha_j-1} \tag{2.88}$$

, where $\sum_{j=1}^{p}\theta_j = 1$, $Beta(\alpha) = \frac{\prod_{j=1}^{p}\Gamma(\alpha_j)}{\Gamma\left(\sum_{j=1}^{p}\alpha_j\right)}$, such that, $\alpha = (\alpha_1, ..., \alpha_p)$, and $E(\boldsymbol{\Theta} = \theta_j) = \frac{\alpha_j}{\sum_{j=1}^{p}\alpha_j}$.

The posterior distribution then is

$$
\begin{aligned}
f(\theta|\mathcal{D}) &\propto f(\theta, \mathcal{D}) \\
&= f(\theta_1, ..., \theta_p) f(x_1, ..., x_p|\theta_1, ..., \theta_p) \\
&\propto \prod_{j=1}^{p} \theta_j^{\alpha_j - 1} \prod_{x_j \in \mathcal{D}} \prod_{j=1}^{p} \theta_j^{x_j} \\
&= \prod_{j=1}^{p} \theta_j^{\alpha_j + \sum\limits_{x_j \in \mathcal{D}} x_j - 1}
\end{aligned} \tag{2.89}
$$

which is the density Dirichlet distribution with hyper-parameters

$$
\alpha_j^* = \alpha_j + \sum_{x_j \in \mathcal{D}} x_j.
$$

That is, $f(\theta|\mathcal{D}) = Drich(\alpha_1^*, ..., \alpha_p^*)$.

Thus, the learning parameters in Bayesian networks, denoted by $P(X_j = x_j|\pi(X_j))$ are given by

$$
\begin{aligned}
P(X_j = x_j|\pi(X_j)) &= \int_0^p P(X_j = x_j|\theta_j) P(\theta_j|\pi(X_j)) d\theta_j \\
&= \int_0^p \theta_j P(\theta_j|\pi(X_j)) d\theta_j \\
&= E(\theta_j|\pi(X_j)) \\
&= \frac{\alpha_j + x_j}{\sum\limits_{j=1}^{p} (\alpha_j + x_j)} \\
&= \frac{\alpha_j + x_j}{\sum\limits_{j=1}^{p} (\alpha_j) + n}.
\end{aligned} \tag{2.90}
$$

Remark 2.6.1. Maximum a posteriori estimation (MAP) selects the parameter configuration for a Bayesian network model, $\hat{\mathbf{\Theta}}$, that maximises the posterior probability of the parameters

$$
\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{argmax}\ P(\mathbf{\Theta}|\mathcal{D}) \tag{2.91}
$$

For more details about Bayesian estimation, see (Hogg (1978); Koller and Friedman (2009); Neapolitan (2003)).

2.6.2. Learning Structure

The main goal of a learning network structure is to perform the density estimation that is a queired from the instances that were not in our training data. An additional reason for a learning network structure is to discover the dependencies in the learned network. The space of a DAG increases exponentially as the number of variables increases (2^p). Also, in space $\mathcal{R}^p$ where $p > 1$ or finite space learning structures become more difficult. Many approaches are proposed to build the pattern of Bayesian networks. The networks will represent the causal relations between the variables, and we can then study the conditional independencies relations based on that. Learning structure can be divided into two approaches: **conditional independencies-based approach** to detect the conditional independencies to search for the Markov blanket for each node and **networks score-based approach** to search for the network that has the maximum score depending on a special scores measure. The Bayesian network score gives each structure a score and searches for the best structure from the range of all possible structures for the Bayesian network. Identifying the highest score of the Bayesian networks requires optimisation methods, such as greedy search, iterated hill climbing, and simulated annealing.

2.6.2.1. Conditional independencies-based approaches

The principle of conditional independencies-based approaches is to search for the Markov blanket for each node X_j, $j = 1, ..., p$; i.e., it selects the relative variables with node X_j to identify the causal relations between the random variables, as the arcs represent probabilistic dependencies. The tests are usually done using statistical methods or information theoretic measures.
Let X and Y be two variables, and $\mathbf{Z}$ be the set of all parents of the conditioning variables, then we can test the following hypothesis:

$$H_0 : X \perp\!\!\!\perp Y|\mathbf{Z} \quad vs \quad H_1 : X \not\perp\!\!\!\perp Y|\mathbf{Z} \tag{2.92}$$

by several statistic tests, such as the *log likelihood ratio* test $G^2(X, Y|\mathbf{Z})$ or Pearson's $\mathcal{X}^2$ test $\mathcal{X}^2(X, Y|\mathbf{Z})$ (Scutari and Denis (2014); Neapolitan (2003)), for discrete data. The *log likelihood ratio* measure is given by

$$G^2(X, Y|\mathbf{Z}) = \sum_{i\in I}\sum_{j\in J}\sum_{k\in K}\frac{n_{ijk}}{n}\log\frac{n_{ijk}\times Count(Z_k = z_k)}{Count(X_i = x_i, Z_k = z_k)\times Count(Y_j = y_j, Z_k = z_k)} \tag{2.93}$$

, where I, J, and K are the levels or categories of X, Y, and $\mathbf{Z}$ respectively, and Person's $\mathcal{X}^2$ is given by

$$\mathcal{X}^2(X, Y|\mathbf{Z}) = \sum_{i\in I}\sum_{j\in J}\sum_{k\in K}\frac{(n_{ijk} - r_{ijk})^2}{r_{ijk}} \tag{2.94}$$

, where

$$r_{ijk} = \frac{Count(X_i = x_i, Z_k = z_k) \times Count(Y_j = y_j, Z_k = z_k)}{Count(Z_k = z_k)},$$

and $n_{ijk} = Count(X_i = x_i, Y_j = y_j, Z_k = z_k)$ is the number of observations when $X_i = x_i$, $Y_j = y_j$ and $Z_k = z_k$. In both cases, the null hypothesis can be tested with the degree of freedom equal to $(I-1)(J-1)K$.

On the other hand, for the continuous training data set, we can test the null hypothesis using various measures such as the t student's test $t(X, Y|\mathbf{Z})$ or Fisher's $\mathcal{Z}$ test $\mathcal{Z}(X, Y|\mathbf{Z})$ (Scutari and Denis (2014); Neapolitan (2003)). The student's t test is given by

$$t(X, Y|\mathbf{Z}) = \rho_{X,Y|\mathbf{Z}} \sqrt{\frac{n-2}{1-\rho^2_{X,Y|\mathbf{Z}}}}, \tag{2.95}$$

with $n - |\mathbf{Z}| - 2$ degree of freedom, where $\rho_{X,Y|\mathbf{Z}}$ is the partial correlation coefficients of X and Y given $\mathbf{Z}$, and the Fisher's $\mathcal{Z}$ test is given by

$$\mathcal{Z}(X, Y|\mathbf{Z}) = \frac{\sqrt{n - |\mathbf{Z}| - 3}}{2} \log \frac{1 + \rho^2_{X,Y|\mathbf{Z}}}{1 - \rho^2_{X,Y|\mathbf{Z}}}. \tag{2.96}$$

The conditional independence approach is equivalent to minimising Kullback–Leibler divergence using the score-based approach. If the null hypothesis is rejected, the arcs $Y \to X$ will exist in the DAG, see figure 2.6b, that represent the null hypothesis.

Many algorithms dependent on the conditional independencies-based approach were introduced to find the structures of DAGs; for example, the inductive causation algorithm (Verma and Pearl (1991); Nagarajan et al. (2014)), PC algorithm (Spirtes et al. (2001)), and grow shrink algorithm (Edera et al. (2014)).

The biggest disadvantage of conditional independent approaches is that the wrong decisions in one test will lead to wrong estimation for new cases and the exponential number of conditional independence tests.

2.6.2.2. Networks score-based approaches

Conditional independence tests with large condition sets may be unreliable unless the size of the data set is large. The score solves the structure as an optimisation problem. We will find the set of possible network structures and use a scoring measure to find the best model fit for the training data, so we will have an exponential number of possible networks to find its score and choose the structure with the highest score. The score is based on penalised log-likelihood, such as the AIC score (Akaike information criterion, (Akaike (1973))), BIC score

(Bayesian information criterion, (Schwarz (1978))), and MDL score (Minimum description length, (Lam and Bacchus (1994); Rissanen (2007); Suzuki (1993))). The Bayesian score is equivalent to the marginal likelihood of the model given the data such as BDeu score (Bayesian Dirichlet equivalent uniform (Heckerman et al. (1995))). The score needs a strategy to choose the best structure from the search space. Many heuristic search methods were proposed for the learning structures of Bayesian networks, such as the greedy search algorithm, simulate annealing, tabu search, or genetic algorithm. The most common search algorithm is the greedy search, such as the hill climbing algorithm, algorithm 1, (Scutari and Denis (2014); Korb and Nicholson (2010)). This algorithm usually starts from an empty structure and finds the structure from the space through searching the maximum score by adding, deleting, or reversing the arc at the same time. The main advantage of this approach is that it uses all the variables at once and builds the whole structure and individual mistakes are less frequent. This dissertation is interested in illustrating the *penalised log-likelihood* score.

The likelihood score is the probability of the training data given a Bayesian network model. The likelihood score is given by

$$\log P(\mathcal{D}|G,\boldsymbol{\Theta}) = \log \prod_{j=1}^{p}\prod_{i=1}^{n} P(X_j = x_{ij}|\pi(X_j)). \tag{2.97}$$

Since the network complexity increases as the number of parents for each node increases, there is a frequency of possible error in the estimation; so a penalty term is added as follows:

$$\log P(\mathcal{D}|G,\boldsymbol{\Theta}) = \log \prod_{j=1}^{p}\prod_{i=1}^{n} P(X_j = x_{ij}|\pi(X_j)) - \psi(n)dim(G) \tag{2.98}$$

, where $dim(G)$ is the dimension of the Bayesian network represented by G, and $\psi(n)$ is function of instances n. The most common score is the BIC score which is given by

$$\begin{cases} BIC = \sum\limits_{j=1}^{p} \log P(X_j|\pi(X_j)) - \frac{l}{2}\log n & \text{for discrete data,} \\ BIC = \sum\limits_{j=1}^{p} \log f(X_j|\pi(X_j)) - \frac{l}{2}\log n & \text{for continuous data,} \end{cases} \tag{2.99}$$

where l is the number of parameters in the network.

2.7. Inference (Queries)

The most important thing that can be done after learning the Bayesian model is to perform inference estimation by computing the posterior probability distri-

Algorithm 1 Hill Climbing Algorithm.

1: Start from complete graph or empty graph G
2: Compute the score of G, denoted by S_G
3: **do**
4: Add or delete or reverse arcs considering that the network structure G must be acyclic and compute the new score S_{Gnew}
5: if $S_{Gnew} > S_G$ then $S_{Gnew} = S_G$
6: **while** the score S_G increases
7: Return the DAG G.

bution over queried nodes given values for evidence nodes. To be as accurate as possible, we can perform inference by applying Bayes' theorem many times to solve the query.

For a simple connection in a Bayesian network structure, $X \rightarrow Y \rightarrow Z$, the posterior probability (belief) of Y, given the evidence about the parent X, is

$$P(X = x|Y = y) = \frac{P(Y = y|X = x)P(X = x)}{P(Y = y)}, \tag{2.100}$$

where $P(X = x)$ is the prior and $P(Y = y|X = x)$ is the likelihood. Also, using the independencies implied in the network,

$$P(Z|X = x) = \sum_{Y=y} P(Z|Y)P(Y|X = x). \tag{2.101}$$

Additionally, if we have evidence about the node, $Z = z$, then the posterior probability of X, given Z, is given by Bayes' theorem and the chain rule

$$\begin{aligned} P(X = x|Z = z) &= \frac{P(Z = z|X = x)P(X = x)}{p(Z = z)} \\ &= \sum_{Y=y} \frac{P(Z = z|Y = y)P(Y = y|X = x)P(X = x)}{P(Z = z)}, \end{aligned} \tag{2.102}$$

since $Z \perp\!\!\!\perp X|Y$.

Note that for long and multiple paths, the inference computation will be complex; so approximate inference algorithms are recommended. The basic idea behind simulation is to generate a large number of cases from the global distribution of G and then estimate the posterior probability using the simulated data. Additionally to generate large cases, let the value of posterior probability converge to the exact value by the Law of Large numbers from statistics. Various simulation methods were discussed for estimating the probability, such as *forward sampling* and *Likelihood weighting* algorithms. In the following, we will illustrate how to find the approximate posterior probability $P(\mathbf{Y} = \mathbf{y}|\mathbf{Z} = \mathbf{z})$,

where $\mathbf{Y}, \mathbf{Z} \subseteq \mathbf{X}$.

These approximate inference algorithms use Monte Carlo simulations to sample from the global distribution of $\mathbf{X}$ and thus estimate posterior probability. In particular, they generate a large number of samples from the Bayesian network and estimate the relevant conditional probabilities by weighing the samples that include both $\mathbf{Y} = \mathbf{y}$ and $\mathbf{Z} = \mathbf{z}$ against those that include only $\mathbf{Z} = \mathbf{z}$. For more details see (Scutari and Denis (2014); Korb and Nicholson (2010); Neapolitan (2003)).

One of the most important concepts related to a sampling algorithm is what is called the topological order.

Definition 2.7.1. [**Topological order, (Koller and Friedman (2009))**]. Let $X_1, ..., X_p$ be nodes on a Bayesian graph $G =< \mathbf{X}, \mathbf{E} >$, where $\mathbf{X}$ and $\mathbf{E}$ as mentioned before; then $X_1, ..., X_p$ is in topological order whenever we have $X_i \to X_j$, then $i < j$.

We can have many topological orders as in figure 2.5; X_1, X_2, X_3, X_4, X_5 or X_2, X_1, X_3, X_4, X_5 or X_1, X_2, X_4, X_3, X_5, among others. That is, the topological orders are not unique.

2.7.1. Forward Sampling simulation algorithm

Logic Sampling simulation (Scutari and Denis (2014); Korb and Nicholson (2010)) starts by ordering the nodes topologically, beginning from the top of the a DAG $G =< \mathbf{X}, \mathbf{E} >$. Let $X_1, ..., X_p$ be the topological order of the DAG, then for each variable $X_j \in \mathbf{X}$, $j = 1, ..., p$, generate a suitable large instance $\mathbf{x_j} = (x_1, ..., x_{n^*})$ from the local Bayesian probability distribution of $X_j|\pi(X_j)$, where n^* is a large sample size. Now we can perform any posterior probability estimation by weighing the samples that include both $\mathbf{Y} = \mathbf{y}$ and $\mathbf{Z} = \mathbf{z}$ against those that include only $\mathbf{Z} = \mathbf{z}$

$$P(\mathbf{Y} = \mathbf{y}|\mathbf{Z} = \mathbf{z}) = \frac{\sum_{i=1}^{n^*} Count_i(\mathbf{Y} = \mathbf{y}, \mathbf{Z} = \mathbf{Z})}{\sum_{i=1}^{n^*} Count_i(\mathbf{Z} = \mathbf{z})}, \tag{2.103}$$

where $\mathbf{Y}, \mathbf{Z} \subseteq \mathbf{X}$. But sometimes the probability of the evidence $P(\mathbf{Z} = \mathbf{z})$ is small which makes the algorithm inefficient because some instances in the generated sample will be discarded without contributing to the estimation of posterior probability.

2.7.2. Likelihood weighing simulation algorithm

We want to find the estimation of the posterior probability as an improvement to forward sampling algorithm. This algorithm (Scutari and Denis (2014);

Korb and Nicholson (2010)) also starts by ordering the nodes of $G =< \mathbf{X}, \mathbf{E} >$ topologically, then for each variable $X_j \in \mathbf{X}$, $j = 1, ..., p$, generate a suitable large instance $\mathbf{x_j} = (x_1, ..., x_{n^*})$ from the local Bayesian probability distribution of $X_j|\pi(X_j)$, where n^* is a large sample size.

In this approach, all the samples generated by likelihood weighing include the evidence by design. However, this means that we are not sampling from the original Bayesian network any more, but we are sampling from a second Bayesian network in which all the nodes in evidence are fixed.

To estimate the posterior probability $P(\mathbf{Y} = \mathbf{y}|\mathbf{Z} = \mathbf{z})$, where $\mathbf{Y}, \mathbf{Z} \subseteq \mathbf{X}$, collect the generated samples together as $\mathbf{x} = (\mathbf{x_{i1}}, .., \mathbf{x_{ip}})$, $i = 1, ..., n^*$; then use the values of evidence $\mathbf{z} = (z_1, ..., z_k)$ that is corresponding to the set of k parents (evidence nodes) $\mathbf{Z}$ to compute the weight w_i

$$w_i = \prod_{X_j \in \mathbf{Z}} P(X_j|\pi(X_j)). \tag{2.104}$$

The name indicates that the weights of different samples are derived from the likelihood of the evidence accumulated throughout the sampling process $(\mathbf{x_1}, w_1)$, ..., $(\mathbf{x_{n^*}}, w_{n^*})$. Then the posterior probability estimation is

$$P(\mathbf{Y} = \mathbf{y}|\mathbf{Z} = \mathbf{z}) = \frac{\sum_{i=1}^{n^*} w_i \mathbf{I}_{(\mathbf{y_i}=\mathbf{y})}}{\sum_{i=1}^{n^*} w_i}, \tag{2.105}$$

where $\mathbf{I}_{(\mathbf{y_i}=\mathbf{y})} = 1$, is the instance $\mathbf{y_i}$ in the generated sample that corresponds to the set of nodes $\mathbf{Y}$ and equal to $\mathbf{y}$. Otherwise, $\mathbf{I}_{(\mathbf{y_i}=\mathbf{y})} = 0$.

3. Classification using Bayesian networks

Supervised learning is an algorithm that uses input categorical training data to infer the mapping function (model) by mapping the training data $\mathcal{D} = (X_1, ..., X_p)$ to output one $\mathbf{Y} = \{1, ..., J\}$, where J is the number of labels.

Classification is to assign labels or categories or levels $Y \in \{1, ..., J\}$ to instances described by a set of predictor variables $\{(y_1, \mathbf{x_1}), ..., (y_i, \mathbf{x_i}), ..., (y_n, \mathbf{x_n})\}$, where $\mathbf{x_i} = (x_{i1}, ..., x_{ip})$ is the i^{th} instance.

Bayesian networks have been extended to answer the classification task where one wishes to predict the label of a class variable $Y \in \{1, ..., J\}$, having observed a set of explanatory variables $\mathbf{X} = (X_1, ..., X_p)$.

Many approaches use simple, general, and complex Bayesian networks for classification. The simplest Bayesian network classifier is the Naive Bayes approach (NB) (Maron and Kuhns (1960); Minsky (1961)), where the components of $\mathbf{X}$ are assumed to be independent given the class Y. Tree Augmented Naive Bayes (TAN) (Friedman et al. (1997)) is a direct extension of NB, where each variable X_j, $j = 1, ..., p$, may depend on at most one other variable than Y. *Unrestricted Bayesian Networks* (Koller and Friedman (2009); Friedman et al. (1997); Heckerman et al. (1995)) build Bayesian networks over the joint set (Y, $\mathbf{X}$) and classify any instance by estimating the posterior probability $P(Y|\mathbf{X})$ using the network. *Multinet Bayesian Networks* (Friedman et al. (1997); Geiger and Heckerman (1996)) build multiple Bayesian networks over the observations corresponding to each label of Y. This gives an estimation of $P(\mathbf{X}, Y)$, and using Bayes rule, one may compute $P(Y|\mathbf{X})$.

Feature selection approaches aim to reduce the dimension of the data, keeping only the important variables for the classification. Different feature selection approaches have been suggested in literature (Langley and Sage (1994); Pazzani and Billsus (1997)). Such approaches may be embedded in the process of classification or used as independent pre-processing. We use the later approaches based on random forests variable importance (Ishak (2007)).

Bayesian networks for continuous data are strongly based on the Gaussian assumption and very sensitive to it. When this assumption is not true it is common to use either Cox-Box transformations or to discretise the data (Friedman et al. (1997)).

In this part of the dissertation, we show that combining feature selection and discretisation through Bayesian network classifiers gives a powerful approach compared to other classical machine learning methods for classification, such as random forests (RF), support vector machines (SVM), and CART (classification and regression trees (Breiman et al. (1984)). We introduce these two new ap-

proaches we have developed together. The method giving the best result over a wide choice of experiments is the multinets approach (Friedman et al. (1997); Geiger and Heckerman (1996)).

A large choice of data sets from the UCI machine learning repository are used in our experiments, and an application to Epilepsy type prediction based on PET (Positron Emission Tomography) scan data confirms the efficiency of our approach.

In next section, we will introduce the most known Bayesian classifiers such as Naive Bayes (NB),(Maron and Kuhns (1960); Minsky (1961)), Tree Augmented Naive Bayes (TAN),(Friedman et al. (1997)), Multinets Bayesian classifier (MN), (Friedman et al. (1997); Geiger and Heckerman (1996)), and Unrestricted Bayesian Networks classifier (UBN) (Friedman et al. (1997); Heckerman et al. (1995); Koller and Friedman (2009)).

3.1. Some existing approaches

We will give here a brief description of some known existing approaches.

3.1.1. Naive Bayes (NB)

Naive Bayes (Maron and Kuhns (1960); Minsky (1961)) is the simplest Bayesian classifier. It is naive because of its naive assumption that all the variables are conditionally independent, given the class variable Y, as shown in figure 3.1, and its Bayes because it calculates the probability of each class, given the other variables, by Bayes rule as

$$P(y|x_1,..,x_p) = \frac{P(Y=y)\prod_{j=1}^{p} P(X_j|Y=y)}{P(X_1,...,X_p)} \tag{3.1}$$

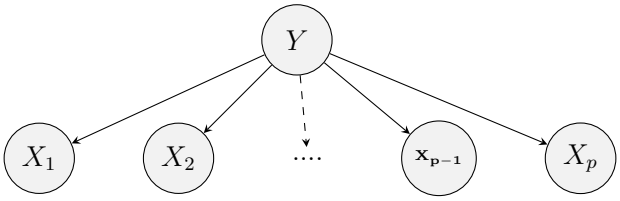


Figure 3.1. – Structure of naive Bayes classifier.

From figure 3.1, the factorisation of the joint probability distribution for naive Bayes classifier based on its assumption is given by

$$P(y, X_1, ..., X_p) = P(y)\prod_{j=1}^{p} P(X_j|y), \tag{3.2}$$

and the class prediction for Y may be computed using

$$\hat{y} = \underset{y}{argmax}\ P(y|X_1, ..., X_p) = \underset{y}{argmax}\ P(y)P(X_1, ..., X_p|y), \tag{3.3}$$

where $P(X_1, ..., X_p|y)$ is inferred from the network. For the learning parameters, we calculate $P(Y = y_i)$, $y_i \in \{1, ..., J\}$ by

$$P(Y = y_i) = \frac{n_i}{n} \tag{3.4}$$

, where n_j is the number of instances that have the class y_i and compute the probability $P(X_j = x_k|Y = y_j)$ as follows:

$$P(X_j = x_k|Y = y_i) = \frac{n_{ijk}}{n_j}, \tag{3.5}$$

for each value x_k of the attribute X_i in the discrete case, where n_{ijk} is the number of cases have the class y_i and value x_k for the attribute X_j. When $X_j's$ are continuous, then each X_j belongs to normal distribution with the mean μ and variance equal σ^2; so

$$P(X_j = x_k|Y = y_i) = g(x_k; \mu_{ij}, \sigma^2_{ij}), \tag{3.6}$$

where g is the normal density function with a mean of X_i equal μ_{ij} and variance σ^2_{ij}. For more details, see (Mitchell (1990); Koller and Friedman (2009)).

The naive Bayes rates well with comparison to other classifiers, especially for medical application (Jebreen and Ghattas (2016)), as we will see in the next sections.

3.1.2. Tree Augmented Naive Bayes (TAN)

TAN (Friedman et al. (1997)) is another type of restricted Bayesian network classifier that takes into account the correlation between the predictor variables. It allows each variable in the network to have at most one other parent than Y, see figure 3.2. The factorisation for joint probability distribution under TAN assumption is given by

$$P(X_1, .., X_p, y) = P(y)P(X_q|y) \prod_{j=1, j\neq q}^{p} P(X_i|y, \pi(X_i)), \tag{3.7}$$

where X_q denotes the root node, which is X_3 in figure 3.2

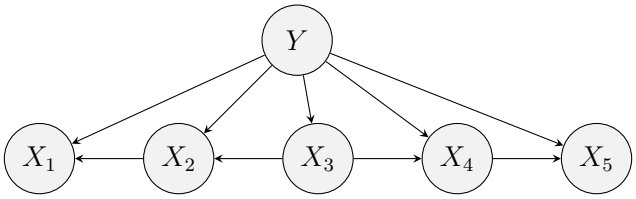


Figure 3.2. – Structure of TAN classifier.

The Chow and Liu algorithm (Chow and Liu (1968)) is updated by (Friedman et al. (1997)) to learn the structure of TAN taking into account the class variable by computing the conditional mutual information between each pair of predictor variables given the class as follows:

$$I(X_i, X_j|Y) = \sum_{x_i} \sum_{x_j} \sum_{y} P(x_i, x_j, y) \log \frac{P(x_i, x_j|y)}{P(x_i|y)P(x_j|y)}, \tag{3.8}$$

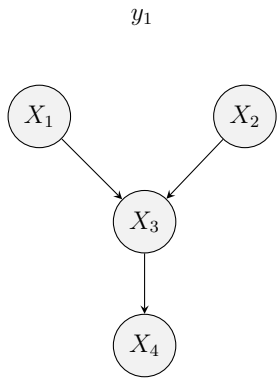
for $i, j = 1, .., p$ and $i \neq j$. Learning the TAN structure starts from a complete undirected graph with edges having a weight equal to $I(X_i, X_j|Y)$. The Kruskal algorithm (Kruskal (1956)) is used to obtain the maximum spanning tree from the structure. Finally, the class variable is set to be the parent of all predictor variables, and the second parent is chosen randomly among the predictor variables to determine the direction of the edges which is agreed with TAN structure restriction.

3.1.3. Multinet Bayesian networks (MN)

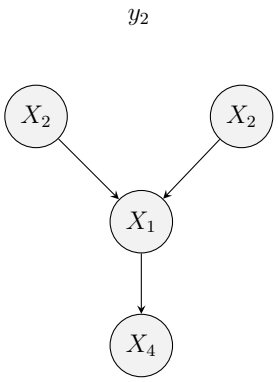
MNs (Friedman et al. (1997); Geiger and Heckerman (1996)) consist of estimating J Bayesian networks separately for each label of Y, figure 3.3. Each class is used to estimate the conditional probabilities $P(\mathbf{X}|Y)$. Bayes rule is then used to estimate the posterior probabilities

$$P(Y|\mathbf{X}) = \frac{P(\mathbf{X}|Y)P(Y)}{\sum\limits_{x,y} P(\mathbf{X}, Y)P(Y)}. \tag{3.9}$$

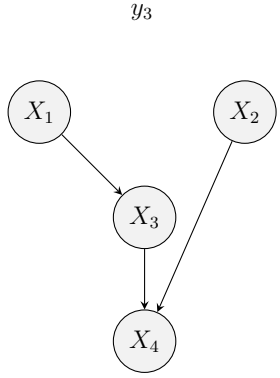
The idea behind this approach is that the interactions between the input variables $\mathbf{X}$ may be different according to the value of Y.



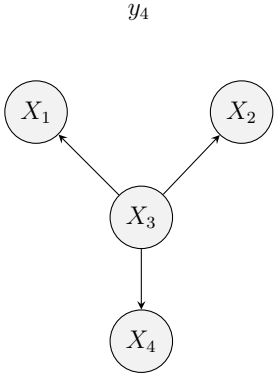
(a) Bayesian networks for the data has label y_1.



(b) Bayesian networks for the the data has label y_2.



(c) Bayesian networks for the data has label y_3.



(d) Bayesian networks for the data has label y_4.

Figure 3.3. – Example of multinets Bayesian networks for the data with four labels.

3.1.4. Unrestricted Bayesian Network Classifier (UBN)

A Bayesian network denoted by $BN = (G, \mathbf{\Theta})$ is a directed acyclic graph G with parameter Θ. Each node of G corresponds to set of variables $\mathbf{X} = \{Y, X_1, ..., X_p\}$, where Y is the class of variables. The structure of graph G represents the dependencies between the variables, and $\Theta = (\theta_1, ..., \theta_p)$ are the conditional probabilities of each variable X_j over $\pi(X_j)$, the set of its parents in the graph. A Bayesian network assumes that the joint probability of $\mathbf{X} = (X_1, ..., X_p)$ is factorised as follows:

$$P(X_1, ..., X_p) = \prod_{j=1}^{p} P(X_j|\pi(X_j)). \tag{3.10}$$

A graph G is unrestricted because there is no restriction between the variables as in NB or TAN. So consider the case where the class of variables Y has no parents. Thus, the factorisation of joint probability distribution over $\mathbf{X}$ is given

by

$$P(Y, X_1, ..., X_p) = P(Y) \prod_{j=1}^{p} P(X_j|\pi(X_j)). \tag{3.11}$$

Learning a Bayesian network (Friedman et al. (1997); Heckerman et al. (1995); Koller and Friedman (2009)) is carried out through the estimation of the structure of the graph G and the set of parameters $\mathbf{\Theta}$ from the data set.

Structure learning for Bayesian networks as we studied before may be done using conditional independency statistical tests or algorithms aimed to maximise a score over the structure (e.g. BIC score and MDL score) using heuristic search algorithms, such as hill clamming algorithm, algorithm 1. In our experiments, we used BIC score to learn the structure of Bayesian networks, which is the most common.

The parameters of Bayesian networks may be learned either by maximum likelihood estimation (MLE) or by Bayesian estimation.

In our work, we used MLE to estimate the parameters for the continuous attributes as studied in the previous sections. For a discrete node, the corresponding conditional distribution is assumed to be multinomial with parameter θ as in the previous sections. The Bayesian Maximum, a posteriori approach, is used to estimate the parameters; i.e, the estimation of θ is the one which maximises

$$P(\Theta|\mathbf{X}) \propto P(\mathbf{X}|\Theta)P(\Theta), \tag{3.12}$$

where $P(\Theta|\mathbf{X})$ is the posterior distribution; $P(\mathbf{X}|\Theta)$ is the multinomial likelihood function, and $P(\Theta)$ is the prior distribution taken to be a Dirichlet distribution.

The posterior probabilities $P(Y|\mathbf{X})$ may be inferred by using different approaches such as using Bayes rule as observed with learning parameters in Naive Bayes or using the approximating inference algorithm in section 2.7.

3.2. Pre-processing

We introduce here two pre-processing approaches often used in supervised learning: discretisation and feature selection.

3.2.1. Discretisation

Bayesian networks for continuous data are strongly based on the Gaussian assumption. In real data sets, this assumption rarely holds, and the variables may be both continuous or discrete. That is why we choose to discretise the data. The discretisation can be carried out with or without the existing class of variables, and keeping the class variable inside computation will keep the information on the data and maximise the dependence between the variable and the

class (Martínez (2010)). Also, the discretisation can be performed on the variables individually or simultaneously. There are many approaches to discretise data, discretisation by interval (Hartemink (2001)), entropy measure (Fayyad and Irani (1993)), ReliefF (Kononenko (1994); Sikonja and Kononenko (1995)), amo0ng others. In the following sections, we will give a brief description of discretisation by entropy and ReliefF measure.

3.2.1.1. Discretisation using entropy measure

The Entropy discretisation method (Fayyad and Irani (1993)) discretises the data to have maximal information gain. This is done by discretising the variable individually to J^* breaks (bins or splits). Let us consider we have data set $\mathcal{D} = \{(x_1, y_1), ..., (x_n, y_n)\}$, with J classes and $y_j \in \{1, .., J\}$ and a partition boundary b^*.

First, it calculates the entropy for each level in the class variable Y by

$$H(Y) = -\sum_{i \in J} p_i \log p_i, \tag{3.13}$$

where p_i is the probability of each label in the class variable. Then for a variable X_j, it randomly chooses a bin b^* that divides the variable instances into two parts: $\mathcal{D}_{1,X_j>b^*}$, which means that all the instances are greater than the bin b^*, and $\mathcal{D}_{2,X_j \leq b^*}$, which means that all the instances are lower than the bin b^* with size n_1 and n_2 respectively and then calculates the entropy for each partition

$$H(Y|X_j, b^*) = \frac{n_1}{n} H(Y|X_j, \mathcal{D}_{1,X_j>b^*}) + \frac{n_2}{n} H(Y|X_j, \mathcal{D}_{2,X_j \leq b^*}). \tag{3.14}$$

Now, computing the information gain for each bins b^* by

$$\text{Information-Gain } (Y|X_j, b^*) = H^*(Y|X_j, b^*) = H(Y) - H(Y|X_j, b^*). \tag{3.15}$$

Repeating the above procedure for different bins b^*'s to select the optimal split point, b^*_{opt}, that corresponds to the highest information gain, you will obtain a binary discretisation by encoding the variable instances with two discrete labels. This approach can then be applied recursively to both of the partitions induced by b^*_{opt} until some condition is satisfied to obtain multiple intervals on the feature X. For that, the entropy discretisation uses the Minimal Description Length Principle to determine the optimal information gained and stops searching for the optimal split point b^*_{opt}, if

$$H^*(Y|X_j, b^*) < \log \frac{n-1}{n} + \frac{(3^{|J|} - 2) + [|J|H(Y) - |J_1|H(Y|X_{j,\geq b^*}, b^*) - |J_2|H(Y|X_{j,<b^*})]}{n}, \tag{3.16}$$

where $|J_i|$ is the number of classes in each partition (Martínez (2010)).

3.2.1.2. Discretisation using ReliefF measure

We used the ReliefF measure (Kononenko (1994); Sikonja and Kononenko (1995)), algorithm 2, for variables' quality estimation to discretise the continuous variables. The ReliefF algorithm selects an instance R_i randomly and then searches for the k nearest instances having the same class as R_i, which is called nearest hits $H_i(Y)$, and searches for the k nearest neighbours having a different class than R_i, which called nearest misses $M_i(Y)$. This process is repeated m times to update the quality estimation for the variables, where m is a user defined parameter. The quality estimation for the variable X_j at every iteration is

$$\begin{aligned} W[X_j] = W[X_j] - & \frac{\sum_{i=1}^{k} \text{diff}(X_j, R_i, H_i)}{m.k} \\ & + \frac{\sum_{Y \neq class(R_i)} \left[\frac{P(Y)}{1-P(class(R_i))} \sum_{i=1}^{k} \text{diff}(X_j, R_i, M_i(Y)) \right]}{m.k}, \end{aligned} \tag{3.17}$$

where $W[X_j]$ is initialised to zero, $\forall\ j = 1, .., p$, and $\text{diff}(X_j, t_1, t_2)$ computes the difference between the values of the attribute X_j for instances t_1 and t_2.

We use the measures

$$\text{diff}(X_j, t_1, t_2) = \left\{ \begin{array}{ll} 0, & value(X_j, t_1) = value(X_j, t_2) \\ 1, & value(X_j, t_1) \neq value(X_j, t_2), \end{array} \right\} \tag{3.18}$$

if the variables are nominal and

$$\text{diff}(X_j, t_1, t_2) = \frac{|value(X_j, t_1) - value(X_j, t_2)|}{max(X_j) - min(X_j)}, \tag{3.19}$$

if the variables are continuous.

To discretise the attribute X_j, the ReliefF measure is used with a greedy search algorithm, algorithm 3, to find the split point, which maximises the heuristic measure $W[X_j]$. At each iteration the algorithm searches for the new split point maximises the heuristic estimate of the discretised variable.

3.2.2. Feature selection using random forest

Here we will give a brief description of the Random Forest (RF) algorithm and the approach of ranking an important variable using RF. We will also illustrate our approach of feature selection based on RF.

Algorithm 2 Algorithm of ReliefF Measure.

1: Set the weights $W[X_j] = 0$ for all $j = 1, ..., p$
2: **for** i=1:m, m is user define **do**
3: Select an instance Ri randomly which has class y ;
4: Select randomly the k nearest neighbours instances, H_i having the same class than R_i;
5: Select randomly the k nearest neighbours instances, M_i having different class than R_i;
6: **for** X_j, j=1:p **do**
7:

$$W[X_j] = W[X_j] - \frac{\sum_{i=1}^{k} \text{diff}(X_j, R_i, H_j)}{m.k} + \frac{\sum_{Y \neq class(R_i)} \left[\frac{P(Y)}{1-P(class(R_i))} \sum_{i=1}^{k} \text{diff}(X_j, R_i, M_i(Y)) \right]}{m.k},$$

8: **end for**
9: **end for**

3.2.2.1. RF Classifier

RF (Breiman (2001)) is among the most known and powerful classification models. They combine (ensemble) a large number of trees (Breiman et al. (1984)) trained over the bootstrap samples of the original data set. The RF algorithm has two main parameters: the number of trees T in an ensemble to grow and the number of features p to select randomly at each split.

3.2.2.2. Feature selection using RF

RFs (Breiman (2001)) are particularly attractive because they offer a very original variable importance measure (Strobl et al. (2008)) widely analysed in litera-

Algorithm 3 Greedy Algorithm searching for the split points that maximise the ReliefF measure.

1: Best Discretisation = {}
2: Set of split point = {}
3: **repeat** m times
4: **if** Set of split points is best so far **then**
5: Best discretisation =Set of split points
6: **end if**
7: **until** the heuristic search is worse than the previous step.

ture and proved to be very efficient in a large number of situations (Díaz-Uriarte and Alvarez de Andrés (2006)).

We use the variable importance assessed by RFs in order to select the best subset of variables for the classification task. This is done following the idea given by (Ishak (2007)) and summarised in algorithm 4. Using the feature selection approach (Langley and Sage (1994); Pazzani and Billsus (1997)), we eliminate unimportant features to include into the model and save costs by learning from the meaningful variables only.

A RF algorithm runs as a large collection of decision trees (Breiman et al. (1984)). Let $\mathcal{D} = (\mathbf{X}, Y)$ be a data set. The RF algorithm chooses a bootstrap sample with replacement $\mathcal{D}_i$ from $\mathcal{D}$ for $i = 1, ..., ntree$ and constructs a tree $\mathcal{T}_i$ using $\mathcal{D}_i$. The prediction of new instances are obtained by averaging the predictions obtained by each tree $\mathcal{T}_i$ in a regression case or taking their majority vote in a classification case. The important thing in RF is the use of an out of bag (OOB) sample (the set of instances in $\mathcal{D}$ that did not appear in $\mathcal{D}_i$) to construct a new measure for variables importance. Breiman (Breiman (2001)) introduces two measures for variables importance using RF feature selection. First, **Gini importance**, which is the total decrease in node impurities through splitting the variable and averaged over all trees (Díaz-Uriarte and Alvarez de Andrés (2006)), but this measure of importance prefers the variables with many categorical levels. Second, **the permutation importance**, which is the difference between OOB prediction accuracy for each tree before and after permuting the values of variable X_j. So, if $\mathcal{B}^{(\mathcal{T})}$ is the OOB sample of the tree $\mathcal{T}$ with $\mathcal{T} \in \{1, ..., ntree\}$, then the variable importance of variable X_j in tree $\mathcal{T}$ is

$$VI^{(\mathcal{T})}(X_j) = \frac{\sum_{i \in \mathcal{B}^{(\mathcal{T})}} I(y_i = \hat{y}_i^{(\mathcal{T})})}{|\mathcal{B}^{(\mathcal{T})}|} - \frac{\sum_{i \in \mathcal{B}^{(\mathcal{T})}} I(y_i = \hat{y}_{i,\pi_j}^{(\mathcal{T})})}{|\mathcal{B}^{(\mathcal{T})}|}, \tag{3.20}$$

where $\hat{y}_i^{(\mathcal{T})}$ is the predicted class for observation i before permuting, and $\hat{y}_{i,\pi_j}^{(\mathcal{T})}$ is the predicted class for observation i after permuting the values of variable X_j, and $x_{i,\pi_j} = (x_{i,1}, ..., x_{i,j-1}, x_{\pi_j(i),j}, x_{i,j+1}, ..., x_{i,p})$. Note that $VI^{(\mathcal{T})}(X_j) = 0$, if the variable X_j does not exist in the tree $\mathcal{T}$. Then the raw variable importance for each variable is

$$VI(X_j) = \frac{\sum_{\mathcal{T}=1}^{ntree} VI^{(\mathcal{T})}(X_j)}{ntree}. \tag{3.21}$$

We compute the average of $VI(X_j)$ over a high number of iterations with high numbers of trees used to grow to reach the stability.

We use the variables' importance obtained from RF to rank the variables in descending order of importance and to select the optimal subset among them. To do that, we test by cross validation the performance of a sequence of RF classifiers each using the k most important variables ranging from $k = 1, ..., p$. The optimal subset of variables corresponds to the one used in the model showing

the highest accuracy. The procedure is described in Algorithm 4.

For each variable, we compute its importance by averaging over 100 runs of RFs. The variables are ranked in the decreasing order of importance and introduced sequentially in an embedded increasing RF model. The accuracy of each model is estimated by ten cross-validation and the optimal number of important variables to retain the one corresponding to the most accurate model.

Algorithm 4 RF Feature Selection.

1: Let $\mathcal{D}$ be the data set and p the number of features.
2: **for** (i=1:100) **do**
3: $VI = \frac{\sum_{i=1}^{100} VI_i}{100}$, where VI_i is the **V**ariables **I**mportance vector at each iteration i.
4: **end for**
5: Sort the variables according to descending order of importance: $X^{(1)}, ..., X^{(p)}$.
6: Partition $\mathcal{D}$ in 10 stratified cross validation sample: $\mathcal{D}_1, ..., \mathcal{D}_{10}$, let $\mathcal{D}_{-j} = \mathcal{D} \backslash \mathcal{D}_j$,
7: **for** (j=1:10) **do**
8: **for** (k=1:p) **do**
9: $M_j^k = f(X^{(1)}, ..., X^{(k)}, \mathcal{D}_{-j})$
10: $Error_j^k = Test(M_j^k, \mathcal{D}_j)$
11: $Error^k = \frac{1}{10} \sum_{j=1}^{10} Error_j^k$
12: **end for**
13: **end for**
14: $kopt = Argmin_k \{Error^k\}$, where $kopt$ is the optimal number of important variables to keep.

3.3. Experimental methodologies and results

In this section, we compare the efficiency of Bayesian network classifiers, NB, TAN, UBN, and MN to other classical methods such as SVM, RF, and decision trees (CART). Support vector machines depend on two parameters k^* and $C = cost$, which are the kernel parameter and the constant of the regularisation term in the Lagrange formulation respectively. These parameters are tuned and compared with their default values, $k^* = 1/p$ and $C = 1$, to choose the best performance (Karatzoglou et al. (2006)). The range of k^* and C are chosen respectively to be $10^{-6:-1}$ and $10^{1:4}$. For RFs, we choose the default values of the parameters suggested in R packages.

The experiments are done over thirteen data sets from the machine learning UCI repository. As mentioned in previous sections, the structure is learned using BIC score, and the learning parameters are computed using MLE for continuous

data sets and Bayesian estimation for discrete data sets. A short description of these data sets is given in Table 3.1. First, all the explanatory variables are discretised using the ReliefF measure. Important variables are computed using RFs by fixing the number of trees to `ntree=5001` to ensure the stability of variables importance, and it is averagely computed over 100 times as shown in algorithm 4.

To assess the accuracy of the classifiers, we compute the average misclassification errors (**MCE**) using five fold cross-validation. Cross-validation is run fifty times in each case and the average over these runs is reported.

Data sets	# Instances	# Variables	# classes	# Impr-discrete	# Impr-continuous
toys	100	50	2	8	4
breast	683	9	2	6	7
glass	214	9	6	9	7
wine	178	13	3	9	8
vehicle	846	18	4	12	8
pima	768	8	2	3	6
satimage	4435	36	6	35	32
segment	2310	19	7	19	7
vowel	990	10	11	10	10
waveform	5000	40	3	21	30
landsat	6435	36	6	36	28
pendigits	10992	16	10	16	16
letter	20000	16	26	16	15

Table 3.1. – Data sets description, where "# Impr-discrete" and "# Impr-continuous" are numbers of important variables in discrete and continuous cases respectively.

Data sets	SVM	CART	RF	NB	UBN	MN
toys	1.50	12.45	6.35	7.39	3.07	40.94
toys-R	0.04	12.14	1.40	2.40	1.19	0.28
breast	3.05	5.31	2.84	3.82	4.60	4.65
breast-R	3.02	5.27	2.95	3.36	4.61	4.62
glass	28.66	31.06	20.89	61.23	45.03	50.28
glass-R	28.15	30.53	20.53	55.89	45.84	51.25
wine	1.79	11.93	1.98	2.69	1.16	0.76
wine-R	1.71	11.58	2.10	2.90	1.29	1.28
vehicle	14.82	31.87	24.86	54.24	15.59	16.01
vehicle-R	21.20	32.06	25.22	50.11	25.52	25.97
pima	22.9	25.67	23.56	24.62	25.05	26.06
pima-R	23.29	25.79	24.10	24.36	24.54	24.99
satimage	8.14	18.96	8.82	20.35	14.54	14.58
satimage-R	8.33	18.91	8.86	20.37	14.54	14.57
segment	3.13	8.09	2.15	20.29	11.84	85.72
segment-R	2.64	8.15	1.62	11.26	7.70	7.56
vowel	1.14	39.83	4.00	33.24	16.29	14.95
vowel-R	1.14	39.83	4.00	33.24	16.29	14.95
waveform	13.61	26.55	14.39	20.01	14.71	14.78
waveform-R	13.36	26.55	14.23	20.02	14.66	14.63
landsat	7.77	18.87	8.24	20.39	14.75	14.60
landsat-R	8.32	18.88	8.30	20.33	14.47	14.51
pendigits	0.38	10.15	0.14	12.01	1.00	24.99
pendigits-R	0.38	10.15	0.14	12.01	1.00	24.99
letter	2.51	60.01	3.70	41.57	12.69	13.94
letter-R	2.61	51.70	3.18	32.60	10.22	10.11

Table 3.2. – Experimental results with (MCE) five fold CV (averaged over fifty runs). Data is continuous and "R" denotes the reduced data set by feature selection.

Table 3.1 gives a summary of the data sets used in the experiments including the number of instances, number of variables, number of labels, and the number of important variables retained by feature selection for both the discretised and continuous versions of data sets. Table 3.2 and Table 3.3 give the missclassification errors of all the compared models for the original continuous data sets and for their discretised version respectively. In both cases, MCE are reported for each data set and its reduced version by feature selection.

Table 3.4 shows the differences of MCE between the continuous and the discrete cases for both reduced and non-reduced data sets. Whereas discretisation does not contribute to increasing the performance of the classical machine learn-

ing approaches, its contribution for BN classifiers, mainly NB and MN are very significant in most cases.

To see clearly whether the feature selection procedure gives rise to better models, we compute the difference of MCE before and after feature selection. These differences are reported in Table 3.5 and Table 3.6 for continuous and discrete cases respectively. We can see that despite the unrestricted BN approach, in most cases, we gain in accuracy when performing feature selection. The gain is higher in general for the discretised version of the data sets.

Data sets	SVM	CART	RF	NB	UBN	MN	TAN
toys	2.23	12.07	3.63	2.51	6.21	2.08	7.15
toys-R	2.67	11.99	3.29	2.33	4.35	2.53	4.00
breast	2.21	5.31	2.73	2.49	3.03	2.95	3.13
breast-R	2.20	5.30	2.37	2.47	2.98	2.68	3.19
glass	22.57	30.90	20.49	26.91	34.60	23.97	22.50
glass-R	22.57	30.90	20.49	26.91	34.60	23.97	22.50
wine	1.49	9.90	1.82	0.99	1.62	2.09	2.23
wine-R	1.67	9.94	1.63	1.12	1.67	1.89	1.79
vehicle	28.84	34.64	29.86	40.62	34.30	29.07	29.35
vehicle-R	26.65	34.27	27.50	38.97	37.64	29.05	29.53
pima	23.34	24.83	23.64	24.88	24.41	24.60	25.40
pima-R	23.14	23.79	21.82	22.30	22.29	22.33	23.65
satimage	10.4	19.06	10.43	19.91	19.01	14.00	13.70
satimage-R	10.46	19.05	10.43	19.93	19.72	13.95	13.46
segment	4.48	11.23	4.47	9.58	6.74	6.53	5.97
segment-R	4.48	11.23	4.47	9.58	6.74	6.53	5.97
vowel	13.64	45.51	13.00	35.01	50.85	26.98	24.71
vowel-R	13.64	45.51	13.00	35.01	50.85	26.98	24.71
waveform	17.36	26.93	18.14	20.43	20.52	21.15	21.14
waveform-R	17.05	26.93	17.77	20.39	20.96	21.11	20.51
landsat	10.00	19.19	9.87	20.26	18.77	13.66	13.38
landsat-R	10.00	19.19	9.87	20.26	18.77	13.66	13.38
pendigits	1.52	20.60	1.77	14.24	6.41	3.09	5.17
pendigits-R	1.52	20.60	1.77	14.24	6.41	3.09	5.17
letter	8.03	53.36	7.59	33.92	24.34	18.20	23.14
letter-R	8.03	53.36	7.59	33.92	24.34	18.20	23.14

Table 3.3. – Experimental Results (MCE) with five fold CV (averaged over fifty runs). Data is discretised and "R" denotes the reduced data set by features selection.

Data sets	SVM	CART	RF	NB	UBN	MN
toys	−0.73	0.38	2.72	4.88	−3.14	38.86
toys-R	−2.63	0.15	−1.89	0.07	−3.16	−2.25
breast	0.84	0.00	0.11	1.33	1.57	1.70
breast-R	0.82	−0.03	0.58	0.89	1.63	1.94
glass	6.09	0.16	0.40	34.32	10.43	26.31
glass-R	5.58	−0.37	0.04	28.98	11.24	27.28
wine	0.30	2.03	0.16	1.70	−0.46	−1.33
wine-R	0.04	1.64	0.47	1.78	−0.38	−0.61
vehicle	−14.02	−2.77	−5.00	13.62	−18.71	−13.06
vehicle-R	−5.45	−2.21	−2.28	11.14	−12.12	−3.08
pima	−0.44	0.84	−0.08	−0.26	0.64	1.46
pima-R	0.15	2.00	2.28	2.06	2.25	2.66
satimage	−2.26	−0.10	−1.61	0.44	−4.47	0.58
satimage-R	−2.13	−0.14	−1.57	0.44	−5.18	0.62
segment	−1.35	−3.14	−2.32	10.71	5.10	79.19
segment-R	−1.84	−3.08	−2.85	1.68	0.96	1.03
vowel	−12.5	−5.68	−9.00	−1.77	−34.56	−12.03
vowel-R	−12.5	−5.68	−9.00	−1.77	−34.56	−12.03
waveform	−3.75	−0.38	−3.75	−0.42	−5.81	−6.37
waveform-R	−3.69	−0.38	−3.54	−0.37	−6.30	−6.48
landsat	−2.23	−0.32	−1.63	0.13	−4.02	0.94
landsat-R	−1.68	−0.31	−1.57	0.07	−4.30	0.85
pendigits	−1.14	−10.45	−1.63	−2.23	−5.41	21.90
pendigits-R	−1.14	−10.45	−1.63	−2.23	−5.41	21.90
letter	−5.52	6.65	−3.89	7.65	−11.65	−4.26
letter-R	−5.42	−1.66	−4.41	−1.32	−14.12	−8.09

Table 3.4. – Difference between MCE with continuous and discrete data.

Data sets	SVM	CART	RF	NB	UBN	MN
toys	1.46	0.31	4.95	4.99	1.88	40.66
breast	0.03	0.04	−0.11	0.46	−0.01	0.03
glass	0.51	0.53	0.36	5.34	−0.81	−0.97
wine	0.08	0.35	−0.12	−0.21	−0.13	−0.52
vehicle	−6.38	−0.19	−0.36	4.13	−9.93	−9.96
pima	−0.39	−0.12	−0.54	0.26	0.51	1.07
satimage	−0.19	0.05	−0.04	−0.02	0.00	0.01
segment	0.49	−0.06	0.53	9.03	4.14	78.16
vowel	0.00	0.00	0.00	0.00	0.00	0.00
waveform	0.25	0.00	0.16	−0.01	0.05	0.15
landsat	−0.55	−0.01	−0.06	0.06	0.28	0.09
pendigits	0.00	0.00	0.00	0.00	0.00	0.00
letter	−0.10	8.31	0.52	8.97	2.47	3.83

Table 3.5. – Difference between MCE with and without feature selection for continuous data. Positive values mean that models are more accurate with feature selection.

Data sets	SVM	CART	RF	NB	UBN	MN	TAN
toys	−0.44	0.08	0.34	0.18	1.86	−0.45	3.15
breast	0.01	0.01	0.36	0.02	0.05	0.27	−0.06
glass	0.00	0.00	0.00	0.00	0.00	0.00	0.00
wine	−0.18	−0.04	0.19	−0.13	−0.05	0.20	0.44
vehicle	2.19	0.37	2.36	1.65	−3.34	0.02	−0.18
pima	0.20	1.04	1.82	2.58	2.12	2.27	1.75
satimage	−0.06	0.01	0.00	−0.02	−0.71	0.05	0.24
segment	0.00	0.00	0.00	0.00	0.00	0.00	0.00
vowel	0.00	0.00	0.00	0.00	0.00	0.00	0.00
waveform	0.31	0.00	0.37	0.04	−0.44	0.04	0.63
landsat	0.00	0.00	0.00	0.00	0.00	0.00	0.00
pendigits	0.00	0.00	0.00	0.00	0.00	0.00	0.00
letter	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 3.6. – Difference between MCE with and without feature selection for discrete data. Positive values mean that models are more accurate with feature selection.

3.4. Application of Bayesian classifier on PET scan data

To illustrate the efficiency of our approach, we apply it to PET (Positron Emission Tomography)scanning data obtained for **fifty four** patients suffering from **four** different types of epilepsy. Each of the **thirty seven** variables in the data corresponds to the signal intensity measured at a region of interest (ROI) in the brain (Guedj et al. (2015)). It is supposed that the connectivity, whether it exists or not, between the ROIs is very important to identify the class of epilepsy. The graphical models used for classification of such data sets coming from such images have shown to be very powerful (Mumford and Ramsey (2014); Smith et al. (2011); Ramsey et al. (2010)).

Epileptic patients are followed by brain PET scan imaging. The images are segmented according to predefined anatomical regions (variables) in the brain. Thirty seven regions were considered for fifty four patients followed at La Timone hospital in Marseille, France. Each patient belongs to one of the four categories of Epilepsy: BILATERAL, LATERAL, MESIAL, or PLUS. The distribution of these labels in our sample are sixteen, seven, seventeen, and fourteen, respectively. For more details on this data set, see (Guedj et al. (2015)).

The aim is to predict the Epilepsy category using the intensity measure of the thirty-seven regions of interest (ROIs). A particular focus in this application is about the complex connectivity of the ROIs in the brain. Statistical models should take into account this connectivity. As it can be seen in Table 3.7 and 3.8, the models fitted over the continuous data sets show very poor performance.

Here we run the classical methods together with the Bayesian classifiers in the previous sections using the same discretisation and feature selection approaches. Since the sample size is very small and as before the MCE, we report using five fold cross-validation as well as using the leave one out (LOO) approach.

Data sets	SVM	CART	RF	NB	UBN	MN	TAN
Epilepsy	31.87	45.67	34.24	43.59	55.22	70.31	–
Epilepsy-R	22.58	43.94	26.10	29.74	36.68	50.52	–
Epilepsy-D	22.03	39.03	20.87	26.75	31.45	26.58	24.25
Epilepsy-D-R	20.25	36.14	18.11	22.03	23.47	17.65	26.03

Table 3.7. – Experimental Results (MCE) with five fold CV (averaged over fifty runs) for Epilepsy data set, "R" denotes the reduced data set by features selection and "D" denotes the discrete data set.

Data sets	SVM	CART	RF	NB	UBN	MN	TAN
Epilepsy	29.63	42.59	31.48	44.44	59.26	70.37	–
Epilepsy-R	22.22	42.59	25.93	27.78	33.33	50.00	–
Epilepsy-D	24.07	46.30	20.37	25.93	24.07	24.07	24.07
Epilepsy-D-R	18.52	42.59	18.52	22.22	22.22	16.67	27.78

Table 3.8. – Experimental Results (MCE) with LOO for Epilepsy data set, "R" denotes the reduced data set by features selection and "D" denotes the discrete data set.

For all models, the feature selection applied to the Epilepsy data decreases significantly their MCE. SVM has the best performance with MCE equal to 31.87% and 22.58% over non-reduced and reduced data respectively in their continuous version.

Except for SVM, MCE are reduced very significantly when the data set is discretised. Finally, using feature selection, MCE is thus reduced reaching 17.65% for multinets Bayesian network classifier (whereas with the original continuous data set it had 70.31%). LOO estimation for MCE are yet lower but show the same patterns.

4. Graphical models for time series

Inferring networks from temporal data (related to time) are widely used processes in the field of graphical model. Many types of dynamic graphical models are used to model this process. We focus on the case where the number of variables p is greater than the number of cases n. In this chapter, we view the most important recent works for inferring the interaction networks.

4.1. Introduction

Modelling interactions between variables is a common task in statistics. This is often done using graphical models (Lauritzen (1996)) where vertices correspond to variables and edges to interaction between the corresponding variables. Such models may be inferred from data using different approaches. Among these approaches *covariance graphs* are the simplest as they infer the network by applying a threshold to the estimated correlation matrix (Cox and Wermuth (1996),Butte et al. (2000)). *Graphical Gaussian models* (GGMs) (Whittaker (1990); Dempster (1972); Lauritzen (1996)) consider rather partial correlations obtained from the inverse of the covariance matrix (Schäfer and Strimmer (2005a,b)).

Many dynamic models are modelled on various probabilistic models such as multivariate autoregressive process (Opgen-Rhein and Strimmer (2007)), State Space, Hidden Markov Models (Beal et al. (2005)), nonparametric additive regression model (Imoto et al. (2002)), among others.

Bayesian networks (BN) (Friedman et al. (1998)) infer interactions by estimating the conditional independence between the variables based on a specific factorisation of the joint probabilities of the variables. Recently, the neighbourhood lasso (Meinshausen and Bühlmann (2006)) and graphical lasso (Friedman et al. (2008)) suggest fitting a regression model for each variable using the others. A variable is connected in the graph to the set of its explanatory variables whose coefficient in the regression model are not zero.

These approaches have been extended to time series data. *Dynamic Bayesian networks* (DBN) (Friedman et al. (1998)) are such direct extensions of Bayesian networks. For the neighbourhood lasso, a variable X at time point $t+1$ is regressed on all the variables observed at time point t.

In this section, we give a brief summary of the recent works on graphical models for time series. Let $\mathbf{X}(t) = (X_1(t), ..., X_p(t))$ be a vectorial real p-dimensional Gaussian process observed at time $t = 1, ..., n$. $\mathbf{X}(t)$ assumed to follow a normal distribution $\mathcal{N}(\mu, \Sigma)$, where μ is the mean vector, and Σ is the covariance matrix. All the approaches described in this section make the following assumption:

First, we assume that $\mathbf{X}$ is first order Markovian, that is

$$P(X(t+1)|X(1),...,X(t)) = P(X(t+1)|X(t)). \tag{4.1}$$

This means that the variables $\mathbf{X}(t+1)$ depend only on the past variables $\mathbf{X}(t)$.

Second, we assume that the process is stationary; that is

$$P(X(t+1)|X(t)) \text{ is independent of } t. \tag{4.2}$$

Third, the variables observed at the same time are conditionally independent, given the others in the past time; that is

$$X_i(t) \perp\!\!\!\perp X_j(t)|\mathbf{X}(t' < t) \text{ , where } i \neq j, t, t' \geq 1. \tag{4.3}$$

These assumptions ensure the existence of a directed acyclic graph (DAG) $G = (\mathbf{X}(t), E(G))$, where $\mathbf{X}(t)$ is the set of variables or nodes, and $E(G) \subseteq (\mathbf{X}(t) \times \mathbf{X}(t))$ is the set of edges. Then, a Bayesian network corresponds to the following representation of the joint distribution of $\mathbf{X}$

$$f(\mathbf{X}(t)) = \prod_{j=1}^{p} \prod_{t=1}^{n} f(X_j(t)|\pi(X_j(t), G)), \tag{4.4}$$

where $\pi(X_i(t), G)$ is the set of parents of $X_i(t)$ in the graph G.

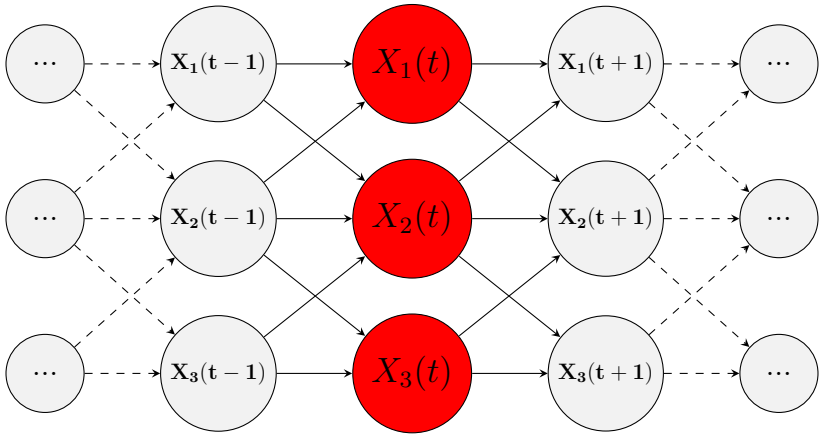


Figure 4.1. – Graphical representation of a time series dynamic Bayesian network.

In this chapter, we introduce various types of dynamic models depending on the concept of the covariance matrix estimation in addition to the dynamic Bayesian networks (DBN). This chapter is organised as follows: section 2 describes vector autoregressive model; section 3 describes the ridge regression for a dynamic process; section 4 describes the lasso approach for a dynamic process (Meinshausen and Bühlmann (2006)); section 5 describes a graphical lasso

approach for a dynamic process (Friedman et al. (2008)); section 6 describes the DBN approach (Friedman et al. (1998); Murphy and Mian (1999)); section 7 describes the shrinkage approach (Schäfer and Strimmer (2005a,b,c); Opgen-Rhein and Strimmer (2007); Opgen-rhein and Strimmer (2006)); section 8 describes the low order conditional dependence approach (G1DBN), and section 9 describes the statistical inference for modular networks approach (SIMoNe).

4.2. Vector autoregressive model (VAR)

The vector autoregressive model of order p^* (lag order) is given by

$$\mathbf{X}(t) = B + \sum_{j=1}^{p^*} \mathbf{A}_j \mathbf{X}(t - jL) + \epsilon_j, \tag{4.5}$$

where p^* is the order of the VAR process; L the time lag, $\mathbf{A}_j = (a_{ij})$ is a $p \times p$ matrix of variables coefficients that represent the dynamical structure, and thus contain the information relevant for reading off the causal relationships; B is a $1 \times p$ vector of means, and ϵ_j is a white noise vector with size p and has a mean equal to zero and positive $p \times p$ definite covariance matrix Σ.

A special case considered in this dissertation is when L and p^* are set to 1. Then the above equation reduces to the VAR(1) process

$$\text{VAR(1)} = \mathbf{X}(t+1) = B + \mathbf{A}\mathbf{X}(t) + \epsilon, \tag{4.6}$$

which is a linear regression of the $\mathbf{X}(t+1)$ variables with the last $n-1$ cases over the variables $\mathbf{X}(t)$ with the first $n-1$ cases. Sometimes we denote the matrices of observations corresponding to $\mathbf{X}(t+1)$, $\mathbf{X}(t)$ by $\mathbf{X}_F$ ("future"), and $\mathbf{X}_P$ ("past"), respectively.

The VAR process can be used directly to infer the interaction networks by testing the connection between each response variable $X_j(t+1)$ at time point $t+1$ and all predictors variables $\mathbf{X}(t)$ at time point t. This is done by testing the significance of the estimated regression coefficients $\hat{\mathbf{A}}^{OLS} = (a_{ij})$ using hypothesis tests, where $\hat{\mathbf{A}}^{OLS}$ is estimated by ordinary least squares (OLS)

$$H_0 : a_{ij} = 0 \quad vs \quad H_1 : a_{ij} \neq 0 \quad \forall i, j = 1, ..., p, \tag{4.7}$$

where this hypothesis can be tested using various tests, such as the student's t-test to compute the *p-value,* p_{ij},

$$t_{cal} = \frac{\hat{a}_{ij}}{\sqrt{var(\hat{a}_{ij})}} \sim t(n-2), \tag{4.8}$$

and then comparing the values with significant level α. Accepting the null hy-

pothesis means that there is no significant relation between the predictor and the response one. We can fill the adjacent matrix with *p-values,* p_{ij}, then choose the best significant connection by sparsing (values under α are set to zero) the matrix under threshold α.

The problem of using this approach is the inefficiency when the number of variables is greater than the number of cases, and thus the coefficients' estimation do not exist. Additionally, there is no optimal value of α to determine the best level of significance.

4.3. Ridge regression model approach for time series

Ridge regression (Hastie et al. (2016)) or L_2 norm regression model is similar to least square, but it shrinks the estimated coefficients towards zero and not exactly equal to zero. The coefficient estimation for the ridge regression is

$$\hat{\mathbf{A}}^{ridge} = \underset{\mathbf{A}\in\mathcal{R}^p}{argmin} \underbrace{\parallel \mathbf{X}(t+1) - \mathbf{A}\mathbf{X}(t) \parallel_2^2}_{\text{Loss}} + \lambda \underbrace{\parallel \mathbf{A} \parallel_2^2}_{\text{Penalty}} \tag{4.9}$$

$$= (\mathbf{X}(t)^T\mathbf{X}(t) + \lambda I_p)^{-1}\mathbf{X}(t)^T\mathbf{X}(t+1). \tag{4.10}$$

The first term in equation 4.9 is called the loss function, and the second term is called the penalty term. Here $\lambda \geq 0$ is a tuning parameter, which controls the strength of the penalty term. Ridge regression penalises the coefficient terms in addition to the residual sum of squared. Note that when $\lambda = 0$, we get the linear regression coefficient estimation, and when $\lambda = \infty$, we get $\hat{\mathbf{A}}^{ridge} = 0$. Moreover, the variance decreases and the bias increases as λ increases. We can choose the best value of λ by cross validation as follows:

$$\hat{\lambda} = \underset{ncv}{min} \frac{\sum_{i=1}^{ncv} MSE(X(t+1) - \hat{X}(t+1))}{ncv}, \tag{4.11}$$

where ncv is the number of folds. We can apply the ridge regression recursively as in the previous section and determine which predictor variables relate to the response one. We are looking for small values of coefficients in $\mathbf{A}$. Since there is no criterion to determine the most related variables with the response one, we can apply test statistics to study the significance of the estimated regression coefficients as in the previous section.

4.4. Lasso regression model approach for time series

Least absolute shrinkage and selection operator (LASSO) estimation or L_1 norm for time series data (Tibshirani (1994)) is defined by

$$\hat{\mathbf{A}}^{lasso} = \underset{\mathbf{A}\in\mathcal{R}^p}{argmin} \underbrace{\parallel \mathbf{X}(t+1) - \mathbf{A}\mathbf{X}(t) \parallel_2^2}_{\text{Loss}} + \lambda \underbrace{\parallel \mathbf{A} \parallel_1}_{\text{Penalty}}, \tag{4.12}$$

where $\parallel \mathbf{A} \parallel_1 = \sum_{j=1}^{p} |\mathbf{A}_j|$ and $\lambda \geq 0$ is a tuning parameter, which controls the strength of the penalty term. The solution of the above equation is complex since it is a non-smooth problem. There are many algorithms for solving the LASSO equation,such as Least angle regression (LAR) algorithm (Efron et al. (2004); Hettigoda (2016)) and shooting algorithm (Fu (1998)). The existence of an L_1 norm allows the LASSO model to perform feature selection by shrinking the coefficients towards exactly zero. As in ridge regression, the variance decreases and the bias increases as λ increases; also, $\hat{\mathbf{A}}^{lasso} = \hat{\mathbf{A}}^{OLS}$, when $\lambda = 0$ and $\hat{\mathbf{A}}^{lasso} = 0$, when $\lambda = \infty$.

LAR algorithm is an efficient algorithm for computing the entire lasso path. At each step, it chooses the best variable to include in the active set and then updates the least squares coefficient to include all the active variables. The idea behind it is to move the coefficient estimates in the direction in which the predictor variable is most correlated with the remaining residual (centered response variable). The coefficients' paths for LAR change continuously as it moves from a vector of zeros to the least square solution. You can look at the LAR algorithm as a forward step-wise regression algorithm.

Meinshausen and Bühlmann (Meinshausen and Bühlmann (2006)) used the Lasso approach for inferring the adjacent matrix. They applied the lasso regression for each variable at time point $t+1$ as a response variable, given the others in earlier time points

$$\hat{A}^{\lambda,j} = \underset{A^j}{argmin} \left[\parallel X_j(t+1) - A^j\mathbf{X}(t) \parallel_2^2 + \lambda \parallel A^j \parallel_1 \right]. \tag{4.13}$$

Solving the above equation for A^j and for optimal value of λ will give all the neighbours (parents) of $X_j(t+1)$ that have non-zero coefficients

$$nebr(X_j(t+1)) = \{X_j(t) \in \mathbf{X}(\mathbf{t}) : a_{ij}(X_j(t)) \neq 0, i, j \in \{1, ..., p\}\}, \tag{4.14}$$

where $nebr$ denotes the neighbours of the variable.

4.5. Graphical lasso model approach for time series

Friedman *et al.* (Friedman et al. (2008); Banerjee et al. (2008)) suggest the graphical lasso regression *(Glasso)* model instead of computing p lasso regression model as in (Meinshausen and Bühlmann (2006)). This method reduces the cost of computation, especially when p is high.

Graphical lasso (Glasso) is a Gaussian graphical model (GGM) that represents the relations between Gaussian random variables. Briefly, consider a Gaussian random vector $\mathbf{X}(\mathbf{t}) = (X_1(t), ..., X_p(t)) \sim \mathcal{N}(\mu, \Sigma)$ and denote the inverse covariance matrix (adjacent matrix) $\mathbf{\Theta} = \Sigma^{-1}$, and let $\mathbf{S}$ be the empirical covariance matrix,and the problem is to maximise Gaussian log-likelihood of the data

$$\log det(\mathbf{\Theta}) - tr(\mathbf{\Theta S}) - \rho \parallel \mathbf{\Theta} \parallel_1, \tag{4.15}$$

where tr denotes the trace, and $\parallel \mathbf{\Theta} \parallel_1$ is the L_1 norm.

4.6. Dynamic Bayesian networks (DBN)

Freidman *et.al.* (Friedman et al. (1998); Murphy and Mian (1999)) suggest two parts to model the process $\mathbf{X}(t)$ using DBN. The first is a prior network BN_0, which determines the distribution of the initial states $X(1)$, and the second is a transition network $BN_{\rightarrow}$, which determines the transition probability $P(X(t+1)|X(t))$ for all t. That is,

$$P_{BN_{\rightarrow}}(\mathbf{X}(1), ..., \mathbf{X}(n)) = P_{BN_0}(\mathbf{X}(1)) \prod_{t=1}^{n-1} P_{BN_{\rightarrow}}(\mathbf{X}(t+1)|\mathbf{X}(t)). \tag{4.16}$$

The structure of a DBN is optimised using BIC is defined as follows:

$$BIC(\mathbf{X}(t), G) = BIC_0 + BIC_{\rightarrow BN}, \tag{4.17}$$

where BIC_0 is the BIC score of the prior network BN_0; $BIC_{\rightarrow BN}$ is the BIC score for the transition network; $BN_{\rightarrow}$; l is the number of parameters in G, and $\hat{f}(X_j(t+1)|X_j(t))$ is the local conditional distribution for each variable.

For more details about dynamic Bayesian approaches for inferring the dynamic network see (Rau et al. (2010)).

4.7. Shrinkage Approach

Schäfer and Strimmer (Schäfer and Strimmer (2005a,b,c); Opgen-Rhein and Strimmer (2007); Opgen-rhein and Strimmer (2006)) propose to apply James Stein type shrinkage estimator to get efficient estimates for a partial correlation matrix in the small sample size when applying the vector autoregressive model (VAR(1)). The use of these estimates come from the inefficient estimators that can be obtained by the least squares and maximum likelihood methods, especially when the number of sample size n is small.

Consider the time series data $\mathbf{X}(t) = (X_1(t), ..., X_p(t))$; the VAR(1) model assumes that the value of $\mathbf{X}(t)$ is a linear combination of the variables in the previous time with noise

$$\mathbf{X}(t+1) = B + \mathbf{A}\mathbf{X}(t) + \epsilon, \tag{4.18}$$

where $\mathbf{X}(t+1)$ is the variables in the next time (future) and denoted by $\mathbf{X}_F$, $\mathbf{X}(t)$ is the variables in the past time and denoted by $\mathbf{X}_P$ and ϵ, Σ^2 as before.

$$\mathbf{X(t+1)} = \mathbf{X_F} = \begin{bmatrix} X_1(2) & \dots & \dots & X_p(2) \\ X_1(3) & \dots & \dots & X_p(3) \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ X_1(n) & \dots & \dots & X_p(n) \end{bmatrix}, \tag{4.19}$$

$$\mathbf{X(t)} = \mathbf{X_P} = \begin{bmatrix} X_1(1) & \dots & \dots & X_p(1) \\ X_1(2) & \dots & \dots & X_p(2) \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ X_1(n-1) & \dots & \dots & X_p(n-1) \end{bmatrix}. \tag{4.20}$$

Note that the ordinary least square (OLS) estimate is given by

$$\hat{\mathbf{A}}^{OLS} = (\mathbf{X}_P^T \mathbf{X}_P)^{-1} \mathbf{X}_P^T \mathbf{X}_F. \tag{4.21}$$

Shäffer and Strimmer proposed to use the James Stein type shrinkage estimator (Efron and Morris (1973)) by shrinking the empirical correlation r_{ij} towards zero and the empirical variance v_i towards the median; i.e, if $\mathbf{X}$ is the centered data with an unbiased empirical estimator of the covariance matrix

$$S = \frac{\mathbf{X}^T \mathbf{X}}{n-1}, \tag{4.22}$$

then the shrinkage estimate S^* is

$$S^*_{ij} = r^*_{ij}\sqrt{v^*_i v^*_j}, \tag{4.23}$$

where

$$\begin{cases} r^*_{ij} = (1-\hat{\lambda}^*_1) r_{ij}, \\ v^*_i = \hat{\lambda}^*_2 v_{median} + (1-\hat{\lambda}^*_2) v_i, \end{cases} \tag{4.24}$$

such that

$$\begin{cases} \hat{\lambda}^*_1 = min\left(1, \dfrac{\sum\limits_{i\neq j} \hat{var}(r_{ij})}{\sum\limits_{i\neq j} r^2_{ij}}\right) \\ \hat{\lambda}^*_2 = min\left(1, \dfrac{\sum\limits_{i=1}^{p} \hat{var}(r_{ij})}{\sum\limits_{i=1}^{p} (v_i - v_{median})^2}\right) \end{cases} . \tag{4.25}$$

Algorithm 5 Shrinkage Approach Algorithm.

1: Construct the augmented matrix $\psi = [X_P X_F]$;
2: Compute $S = \psi^T \psi$ and extract the two submatrices $S_1 = X_P^T X_P$ and $S_2 = X_P X_F$;
3: Compute $\hat{\mathbf{A}}^{shrink} = (S^*_1)^{-1} S^*_2$;
4: Find the $p \times p$ matrix of partial correlation coefficient $\widetilde{r} = (\widetilde{r}_{ij})$ as in equation 4.31;
5: Compute the local fdr and delete the non-significant edges ($fdr \geq 0.2$) by replacing its weight by 0 to obtain the score (adjacent) matrix..

Shrinkage approach as shown in algorithm 5 starts by combining the centred observation $\mathbf{X}_P$ and $\mathbf{X}_F$ using the joint matrix

$$\psi = [X_P X_F] = \left[\begin{array}{cccc|cccc} X_1(1) & \dots & \dots & X_p(1) & X_1(2) & \dots & \dots & X_p(2) \\ X_1(2) & \dots & \dots & X_p(2) & X_1(3) & \dots & \dots & X_p(3) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_1(n) & \dots & \dots & X_p(n) & X_1(n-1) & \dots & \dots & X_p(n-1) \end{array}\right]. \tag{4.26}$$

Note that the matrix ψ has dimension equal to $(p-1) \times 2p$. Then, the empirical covariance is given by

$$S = \psi^T \psi \tag{4.27}$$

$$= (\mathbf{X}_P \mathbf{X}_F)^T (\mathbf{X}_P \mathbf{X}_F). \tag{4.28}$$

Also, note that S will be a matrix with dimension $2p \times 2p$ and contain the four submatrices as follows:

$$\psi = \begin{matrix} \mathbf{X}_P & \mathbf{X}_F \\ \begin{bmatrix} \mathbf{X}_P^T\mathbf{X}_P & \mathbf{X}_P^T\mathbf{X}_F \\ \mathbf{X}_F^T\mathbf{X}_P & \mathbf{X}_F^T\mathbf{X}_F \end{bmatrix} & \begin{matrix} \mathbf{X}_P \\ \mathbf{X}_F \end{matrix} \end{matrix},$$

where $S_1 = \mathbf{X}_P^T\mathbf{X}_P$, and $S_2 = \mathbf{X}_P^T\mathbf{X}_F$.

Based on that, the ordinary least square estimate of VAR coefficients is

$$\hat{\mathbf{A}}^{OLS} = (S_1)^{-1}S_2. \tag{4.29}$$

Replacing the empirical covariance matrix S by a shrinkage estimate to get

$$\hat{\mathbf{A}}^{Shrink} = (S_1^*)^{-1}S_2^*. \tag{4.30}$$

Instead of using a statistical test to identify the most significant coefficient in $\hat{\mathbf{A}}^{Shrink}$, they test the corresponding partial correlation coefficients that study the dependencies among the estimated coefficients.

Consider the successive fitting of VAR(1) model for each variable at time point $t+1$ on all the variables at the previous time point t. Then the partial correlation between the estimated regression coefficient (Whittaker (1990)) is

$$\widetilde{r}_{ij} = sign(\hat{a}_i^{(j)})\sqrt{\hat{a}_i^{(j)}\hat{a}_j^{(i)}}, \tag{4.31}$$

where $\hat{a}_j^{(i)}$ denotes the estimated regression coefficient of the predictor variable X_j for response X_i, and $sign(\hat{a}_i^{(j)})$ is the sign of $\hat{a}_i^{(j)}$. After computing the $p \times p$ partial correlations matrix, Schafer and Strimmer use the *local false discovery rate approach (fdr)* (Efron (2007); Hotelling (1953)) to identify the significant partial correlations. The edges are considered to be significant, if its local fdr is less than 0.2.

4.8. Low order conditional dependence (G1DBN)

Lèbre (Lèbre (2009)) proposed an inference method for DBNs based on the idea of a low order conditional dependence graph G1DBN. This approach is performed in two steps. First, find the adjacent matrix by computing the first order partial dependencies for the graph $G^{(1)}$ and choose the most significant edges based on a user chosen parameter α_1 (perform dimension reduction). This step is performed using statistical tests. Consider fitting the VAR(1) model recursively for each variable in next time over the variables on the past time as follows:

$$X_i(t+1) = b_{ijk} + a_{ij|k}X_j(t) + a_{ik|j}X_k(t) + \epsilon_{ijk}, \quad \forall i,j,k = 1,...,p \; and \; j \neq k. \tag{4.32}$$

Then, measure the conditional dependence between the variables $X_i(t+1)$ and $X_j(t)$, given $X_k(t)$, $\forall i,j,k \neq j$ through testing the partial regression coefficient $a_{ij|k}$

$$H_0 : a^{(1)}_{ij|k} = 0 \quad H_1 : a^{(1)}_{ij|k} \neq 0, \quad \frac{\hat{a}^{(1)}_{ij|k}}{v\hat{a}r(\hat{a}^{(1)}_{ij|k})} \sim t(n-4), \tag{4.33}$$

where $t(n-4)$ is the student's test with $n-4$ degrees of freedom, and $v\hat{a}r(\hat{a}_{ij|k})$ is the variance estimation for $\hat{a}_{ij|k}$. The result of this step will be a score matrix with $p \times p$ dimension with *p-values* p_{ij} entering from the above test such that

$$p_{ij} = \underset{i,j,k \neq j}{max} \; p_{ij|k}. \tag{4.34}$$

Note that the smallest *p-values* represent the most significant edges on $G^{(1)}$, depending on the significant level α_1. Moreover, $p_{ij} \neq 0$ means there is an arc from node j to node i.

Second, in this step, the real structure G of DBN that reflects the relevant dependence that exists in $G^{(1)}$ is learned. Model selections are performed using standard estimation and tests over the edges of $G^{(1)}$. Here the number of parents for each variable in $G^{(1)}$ must be less than $n-1$ by controlling the value of α_1. Choosing the variables $X_i(t+1)$ that have parents less than $n-1$ and test the regression coefficients of the VAR(1) under significant level α_2

$$X_i(t+1) = b_i + \sum_{j \in Pa(X_i(t+1), \hat{G}^{(1)})} a^{(2)}_{ij} X_j(t) + \epsilon_{ij}, \tag{4.35}$$

$$H_0 : a^{(2)}_{ij} = 0 \quad H_1 : a^{(2)}_{ij} \neq 0; \tag{4.36}$$

note that

$$\frac{\hat{a}^{(2)}_{ij}}{v\hat{a}r(\hat{a}^{(2)}_{ij})} \sim t(n-1-|Pa(X_i(t+1), \hat{G^{(1)}})|), \tag{4.37}$$

where $|Pa(X_i(t), \hat{G}^{(1)})|$ is the number of parents of $X_i(t+1)$ in $\hat{G}^{(1)}$. The final DAG will have edges with *p-values* scores equal to $p_{ij} < \alpha_2$.

4.9. Statistical Inference for Modular Networks, SIMoNe

Ambroise *et.al.* (Ambroise et al. (2009)) suggest an algorithm called *SIMoNe* (Statistical Inference for Modular Networks) to estimate the non-zero entries of the concentration matrix equivalent to reconstructing the Gaussian graphical model. They assume a latent structure on the concentration matrix equivalent to a hidden structure over the network (whose edges' weights correspond to the entries of the concentration matrix). Finally, they use an EM algorithm together with a L_1 norm to get the concentration matrix estimate.

Other approaches based in mutual information criterion were also proposed; for details, see (Altay and Emmert-Streib (2010); Basso et al. (2005); Faith et al. (2007); Meyer et al. (2007); Peng et al. (2005); Reverter and Chan (2008)).

In next chapter, we propose two new approaches for inferring dynamic graphical networks: neighbourhood SVM (nSVM), and restricted Bayesian networks (RBN).

5. Graphical model for time series using support vector machines

In this chapter, we consider modelling the interaction between a set of variables in the context of time series and high dimension. We suggest two approaches. The first is similar to the neighborhood lasso when the lasso model is replaced by a support vector machine (SVM). The second is a restricted Bayesian network adapted for time series. We show the efficiency of our approaches by simulations using linear and nonlinear data set and a mixture of both.

SVMs) (Vapnik (1995)) may be used for regression and share many features with the classification version. Suppose we have the training data set $\mathcal{D} = \{(x_1, y_1), ..., (x_n, y_n)\} \subset \mathcal{R}^p \times \mathcal{R},\ p \geq 1$. In epsilon SVM regression ($\epsilon - SVM$), we aim to estimate a function $f(x)$ that has at most ϵ deviation from the actually obtained targets y_i for all the training data. Let f be a linear functions with the form

$$f(x) =< w, x > +b, \qquad w \in \mathcal{R}^p, b \in \mathcal{R}, \tag{5.1}$$

$\epsilon-$SVM solve the following optimization problem

$$\begin{aligned} &\underset{w}{\text{minimize}} \quad \frac{1}{2} \parallel w \parallel^2 + C \sum_{i=1}^{n} (\zeta_i + \zeta_i^*), \\ &\text{subject to} \begin{cases} y_i - < w, x_i > -b \leq \epsilon + \zeta_i \\ < w, x_i > +b - y_i \leq \epsilon + \zeta_i^* \\ \zeta_i, \zeta_i^* \geq 0 \end{cases}, \end{aligned} \tag{5.2}$$

where $C > 0$ is a constant that controls the penalty imposed on observations, which lie outside the ϵ margin and prevent overfitting, and ζ_i, ζ_i^*, are slake variables controlling the relaxation of the constrains. The linear ϵ-insensitive loss function ignores errors that are within ϵ distance of the observed value by treating them as equal to zero. The loss function is a measure based on the distance between the observed value y and the ϵ boundary

$$L_\epsilon = \begin{cases} 0, & |y - f(x)| \leq \epsilon \\ |y - f(x)| - \epsilon, & otherwise \end{cases}. \tag{5.3}$$

Solving the optimization problem (5.2) is done by solving its Lagrange dual for-

mulation (Fletcher (1987); Mangasarian (1969); McCormick (1983)) and gives

$$w = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) x_i, \tag{5.4}$$

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) < x_i, x > + b, \tag{5.5}$$

where α_i and α_i^* are the Lagrange multipliers. The constant b can be computed by the so-called Karush Kuhn Tucker (KKT) conditions (Kuhn and Tucker (1951); Karush (1939)), which state that the point of solution of the product between dual variables and constrains has to vanish

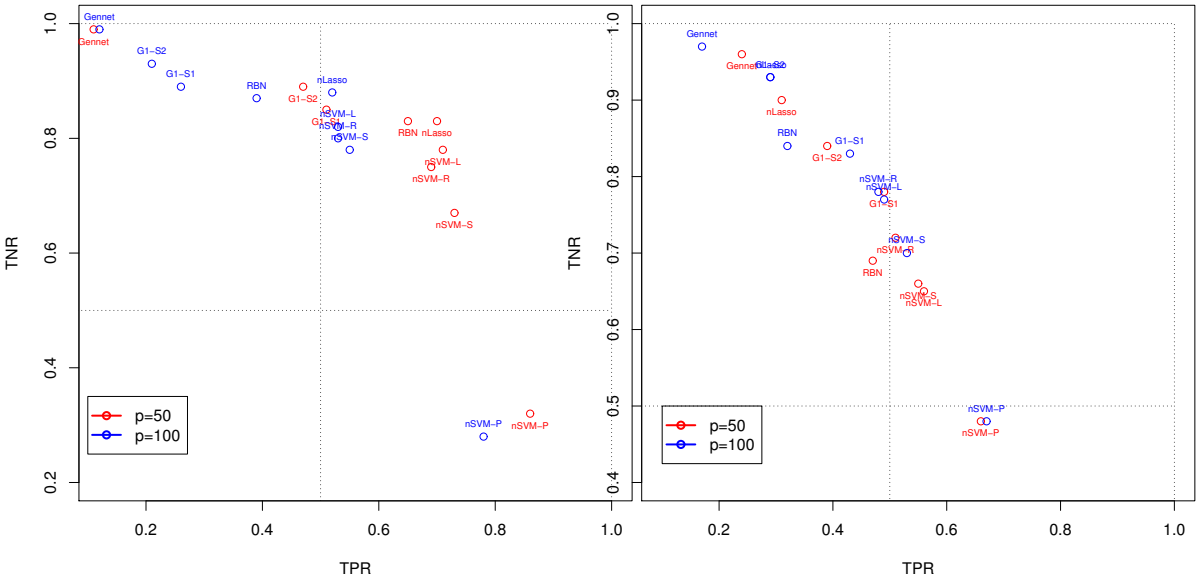
$$\begin{aligned} \alpha_i(\epsilon + \zeta_i - y_i + < w, x_i > + b) &= 0, \\ \alpha_i^*(\epsilon + \zeta_i^* + y_i - < w, x_i > - b) &= 0, \\ (C - \alpha_i)\zeta_i &= 0, \\ (C - \alpha_i^*)\zeta_i^* &= 0. \end{aligned} \tag{5.6}$$

So w can be completely described as a linear combination of the training patterns x_i and the complexity of a function's representation by support vectors. Also, it depends on the number of support vectors but not on the dimension p.

In the case of nonlinear SVMs, the Lagrange dual formulation is extended to a nonlinear function. The nonlinear SVM regression model can be obtained by replacing the dot product term $< x_j, x >= x_j^T x$ with a nonlinear kernel function $K(x_1, x_2) =< \phi(x_1), \phi(x_2) >$, where $\phi(x)$ is a transformation that maps x to a high dimensional space. The common kernels are

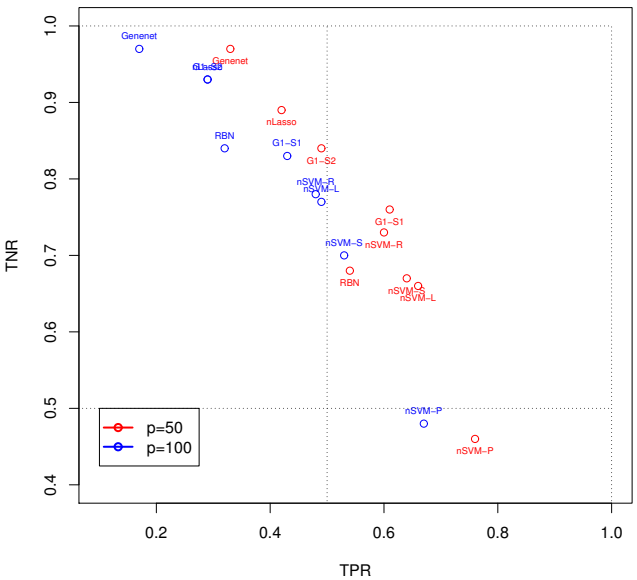
$$K(X_i, X_j) = \begin{cases} \text{Polynomial kernel} = (k^* < x_i, x_j > + const)^d \\ \text{Gaussian radial basis function} = e^{-k^* \|x_i - x_j\|^2} \\ \text{Sigmod kernel} = tanh(k^* < x_i, x_j > + const) \end{cases}, \tag{5.7}$$

where k^* is the kernel parameter; d is the degree of polynomial kernel, and $const$ is a random constant.



(a) Linear simulation: position of approaches with respect to TPR and TNR for $p = 50, 100$.

(b) Nonlinear simulation: position of approaches with respect to TPR and TNR for $p = 50, 100$.



(c) Mixed simulation: position of approaches with respect to TPR and TNR for $p = 50, 100$.

Figure 5.1. – Approaches positions in linear, nonlinear simulation, and mixture of both for $p = 50, 100$ and $n = 20$.

5.1. Neighborhood Support vector machine, (nSVM)

Following the idea of neighborhood lasso, we suggest a procedure here based on p SVM regression models, where each variable observed at time point $t+1$ plays the role of the output, and the other variables observed at previous time point t are used as input variables. The difference from neighborhood lasso is that the feature selection step is done separately. For each regression model, the optimal subset of input variables is selected by a step-wise type procedure. First input variables are ranked according to their decreasing order of importance. The importance of variable X_j based on SVM is computed using $||w||^{(-j)}$, (Cristianini and Shawe-Taylor (2000); Rakotomamonjy (2003); Yu and Kim (2012); Lin (2008)), which is the norm of the weight vector omitting its jth coordinate. Once the input variables are ordered, we construct a sequence of embedded models beginning with the most important variable and adding the others one by one (Ishak (2007)). The mean square error (MSE) of each model is computed by leave one out cross validation (LOOCV). The model minimises the MSE that corresponds to the best subset selection of input variables, thus the optimal neighbor. The algorithm is summarised in algorithm 6.

Algorithm 6 Neighborhood Support vector machine algorithm.

Let $\mathcal{D}$ be a data set; p is the number of features; $error$ and $Error$ be vectors of MSE with length p ;
for (j=1:p) **do**
    Build a SVM model f for each response variable $X_j(t+1)$ and predictor variables $\mathbf{X}(t)$;
    Compute the variable importance (VI) with respect to the SVM model;
    Sort the variables according to their descending order of importance: $X^{(1)}(t), ..., X^{(p)}(t)$;
    Partition $\mathcal{D}$ using LOOCV and let $\mathcal{D}_{-i} = \mathcal{D} \backslash \mathcal{D}_i$;
    Initialize $Error = 0$
    for (i=1:n) **do**
        for (k=1:p) **do**
            $M_i^k = f(response = X_j(t), predictors = X^{(1)}(t), ..., X^{(k)}(t), \mathcal{D}_{-i})$;
            $error_i^k = Test(M_i^k, \mathcal{D}_i)$;
        end for
        $Error = Error + error_i$;
    end for
    $Error = \frac{1}{n} Error$;
    $kopt = \underset{k}{argmin}\{Error\}$, where $kopt$ is the optimal number of important variables to keep in the model.
end for

5.2. Restricted Bayesian Networks (RBN)

Our idea here is to use the classical static Bayesian network approach augmenting the data set by adding one time shift for each variable. Therefore, we built a Bayesian network over the set of $2p$ variables $(X_1(t+1), ..., X_p(t+1), X_1(t), ..., X_p(t))$ constraining the network to satisfy the assumptions, as given in section 4.1. Figure 4.1 illustrates these restrictions. There are no dependencies within each time, and the arcs between times are only in one direction $(t \to t+1)$. Besides, the graph is acyclic.

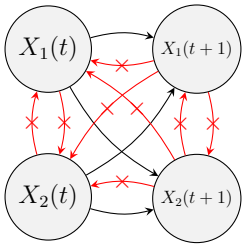


Figure 5.2. – Restricted dynamic Bayesian network, RDBN.

5.3. Experiments

In this section, we suggest three different simulation models for linear and nonlinear time series and a mixture of both.

5.3.1. Simulation models

For the linear time series simulation, we use a first order vector autoregressive model, VAR(1)

$$X_j(t+1) = A\mathbf{X}(\mathbf{t}) + B + \epsilon_j, \ j = 1, ..., p, \tag{5.8}$$

where $t = 1, .., n$, $\mathbf{X}(\mathbf{t}) \in \mathcal{R}^p$ and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. The matrix $A_{p\times p}$ represents the true network structure. Its elements are chosen uniformly, fixing the true edges proportion $pi \in (0, 1)$ of non zeros entries. The vector B of intercepts is also chosen uniformly. Details are given in algorithm 7 (Lèbre (2009); Opgen-Rhein and Strimmer (2007)).

For nonlinear time series, we follow the simulation scheme given in (Fujita et al. (2008))) and use the following transformations:

$$\begin{cases} f_1(X(t+1)) = \sin(X(t)) \\ f_2(X(t+1)) = \cos(X(t)) \\ f_3(X(t+1)) = \sqrt[3]{X^2(t)} - 2^{\sin(X(t))} \end{cases} . \tag{5.9}$$

The initial values $X(1) \in \mathcal{R}^p$ are drawn randomly using standard Gaussian distribution with zero mean and variance σ^2.

The p nonlinear functions are drawn randomly from the above transformations (equation 5.9) and applied to each dimension of $\mathbf{X}$ at time t. A matrix A as generated in the linear case is applied after nonlinear transformation. The process is described in algorithm 8.

To mix the linear and nonlinear simulation, we use the following set of transformations:

$$\begin{cases} f_4(X(t+1)) = \sin(X(t)) \\ f_5(X(t+1)) = \frac{1}{2}X(t) \\ f_6(X(t+1)) = \sqrt[3]{X^2(t)} - 2^{\sin(X(t))} \\ f_7(X(t+1)) = -0.8X(t) \end{cases}, \tag{5.10}$$

Then, we proceed exactly like in the nonlinear case.

To test the performance of our approaches, we compared the efficiency of neighborhood support vector machines approach (nSVM) and restricted Bayesian networks approach (RBN) with first order dependencies approach (G1DBN) (Lèbre (2009)), shrinkage to large scale covariance matrix estimation approach (Genenet) (Schäfer and Strimmer (2005c); Opgen-Rhein and Strimmer (2007)), and neighborhood lasso approach (nlasso) (Meinshausen and Bühlmann (2006)). Support vector machines depend on two parameters k^* and $C = cost$, which are the kernel parameter and the constant of regularisation in the Lagrange formulation, respectively. These parameters are tuned and compared with their default values, $k^* = 1/p$ and $C = 1$, to choose the best performance (Karatzoglou et al. (2006)). The range of k^* and C are chosen respectively to be from 10^{-6} to 10^{-1} and from 10^1 to 10^6. Also, for polynomial kernel, the parameter d is tuned to choose the best degree within $d = \{1, .., 5\}$.

5.4. Results

We now compare the different approaches described above; $G1(S1$ and $S2)$ that correspond to the two steps of G1DBN approach, neighborhood lasso (nlasso) approach, the Genenet approach with our approaches restricted Bayesian network approach (RBN), and neighborhood SVM approach with different kernels; linear (L), radial (R), sigmod (S), and polynomial (P).

For these comparisons, we compute the true positive rate (TPR), false positive rate (FPR), true negative rate (TNR), and false negative rate (FNR) defined in equation 5.13 and average their values over 100 runs.

Algorithm 7 Simulation of linear networks.

1: Let $X = Zero_{p\times n}$ the p-dimensional time series initialised to zero; n the number of instances; $pi \in (0,1)$ the true edges proportion; $A = Zero_{p\times p}$ the adjacency matrix that represents the simulated graph (initialized to zero); B the intercept term; ϵ_i a white noise with zero mean and variance equal to σ^2; and nEdges the number of nonzero edges in the network;
2: nEdges=$\lfloor p^2 \times pi \rfloor$;
3: Select randomly the nonzero edges in A from p^2 edges;
4: Fill the nonzero edges in $A = (a_{ij})$ uniformly;
5: Define the true network

$$Tnet = \begin{cases} 1, if & a_{ij} \neq 0 \\ 0, if & a_{ij} = 0 \end{cases} ; \tag{5.11}$$

6: Draw the intercept term B and the variance σ^2 uniformly;
7: Draw the initial value $X(1)$ normally with zero mean and variance equal to σ^2;
8: **for** (i=2:n) **do**
$X(i) = AX(i-1) + B + \epsilon_i, ; \qquad where \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2)$
9: **end for**
10: X^T is the simulated times series.

$$\begin{aligned} TPR &= \frac{TP}{TP+FN}, \quad FPR = \frac{FP}{FP+TN} \\ TNR &= \frac{TN}{TN+FP}, \quad FNR = \frac{FN}{FN+TP} \end{aligned} \tag{5.13}$$

Table 5.1, 5.2 and 5.3 present the results for the linear, nonlinear, and mixture cases respectively for two values of p ($p = 50$, $p = 100$) and n being fixed to 20.

As expected in all cases, the performances decrease for high dimensions ($p = 100$) and is quite good for all the methods in the linear case. The worst performance for all the method is observed in the nonlinear case. Given the low rate (5%) of edges present in the true network, the hardest task is to retrieve these edges thus to get high TPRs. High values of TNR are quite easy to achieve and correspond systematically to low values of the MCE rates.

The nSVM is the only approach where TPR is above 50%. As there is no global index to measure fairly the performances of these approaches, we try in general to have a good trade off between TPR and TNR.

Figure 5.1 shows the position of the approaches we have compared in the space (TPR-TNR).

For the linear results, Table 5.1 shows that the average number of edges that are correctly included into the estimate of the edge set is high in nSVM-S, nlasso, nSVM-L, nSVM-R, and RBN, respectively, when $p = 50$. These high average

Algorithm 8 Simulation of nonlinear and mixture networks.

1: Let $X = Zero_{p\times n}$ be the p-dimensional time series initialized to zero ; n the number of instances; $pi \in (0, 1)$ the true edges proportion; $A = Zero_{p\times p}$ the adjacency matrix that represents the simulated graph (initialized to zero); ϵ_i a white noise with zero mean and variance equal to σ^2; and nEdges the number of nonzero edges in the network;
2: nEdges=$\lfloor p^2 \times pi \rfloor$;
3: Select randomly the nonzero edges in A from p^2 edges;
4: Fill the nonzero edges in $A = (a_{ij})$ uniformly;
5: Define the true network

$$Tnet = \begin{cases} 1, if & a_{ij} \neq 0 \\ 0, if & a_{ij} = 0 \end{cases} \quad ; \tag{5.12}$$

6: Choose the transformation function f_j randomly from equation 5.9 or equation 5.10 for each variable.
7: Draw the initial value $X[, 1]$ normally with zero mean and random variance and set $X[, 1] = 2 \times \sin(X[, 1])$
8: **for** (i=2:p) **do**
9: **for** (j=1:p) **do**
$X[j, i] = f_j(X[j, i - 1])$
10: **end for**
$X[, i] = A \times X[, i] + \epsilon_i, ; \qquad where \quad \epsilon_i \sim (0, \sigma^2)$
11: **end for**
12: X^T is the simulated times series.

values also correspond to high average values of edges that are not correctly included into the estimate of the edge set and to low average values of misclassification error; this is obvious from the dots in the upper right-hand side square in figure 5.1a with red color .

When $p = 100$, the average number of edges that are correctly included and not included into the estimate of the edge set breaks down in the RBN approach and is out of performance, but still significant in nSVM-S, nSVM-L, nSVM-R, and nlasso, respectively; see the blue dots in the upper right-hand side square in figure 5.1a.

In nonlinear simulation, nSVM still gives highly significant results in both cases, when $p = 50$ or $p = 100$. Note that the average number of edges that are correctly included and not included into the estimate of the edge set when $p = 50$ is high and close to its corresponds one when $p = 100$ in nSVM with sigmod kernel which shows the stability of the results that are function of the number of variables p. See the red and blue dots in the upper right-hand side square in figure 5.1b.

	G1-S1	G1-S2	Gennet	nlasso	RBN	nSVM-L	nSVM-R	nSVM-S	nSVM-P
Number of variables $p = 50$									
TPR	0.51	0.47	0.11	0.70	0.65	0.71	0.69	0.73	**0.86**
FPR	0.15	0.11	**0.01**	0.17	0.17	0.22	0.25	0.33	0.68
TNR	0.85	0.89	**0.99**	0.83	0.83	0.78	0.75	0.67	0.32
FNR	0.49	0.53	0.89	0.30	0.35	0.29	0.31	0.27	**0.14**
MCE	0.17	0.13	**0.05**	0.18	0.18	0.23	0.25	0.33	0.66
Number of variables $p = 100$									
TPR	0.26	0.21	0.12	0.52	0.39	0.53	0.53	0.55	**0.78**
FPR	0.11	0.07	**0.01**	0.12	0.13	0.18	0.20	0.22	0.72
TNR	0.89	0.93	**0.99**	0.88	0.87	0.82	0.80	0.78	0.28
FNR	0.74	0.79	0.88	0.48	0.61	0.47	0.47	0.45	**0.22**
MCE	0.14	0.11	**0.06**	0.14	0.16	0.19	0.22	0.23	0.69

Table 5.1. – Linear simulated data, p=50, 100, n=20, pi=0.05, the last four columns correspond to the neighborhood SVM approach (nSVM) using different kernels (L: Linear, R: Radial, S: Sigmod, P: Polynomial).

In Table 5.3, which is the simulated results of mixture linear and nonlinear time series data, nSVM approach also gives highly significant results in both cases of different number of variables, especially in nSVM-S. On the other side, RBN and G1-S1 break down when the number of variables increase. See the red and blue dots in the upper right hand side square in figure 5.1c.

In all the simulations, the MCE decreases as the number of variables increase. Moreover, nSVM approach, especially SVM-S, shows the best performance in finding the most correct edges that are included into the estimate of the edge set. RBN is sensitive to the linearity assumption and the number of variables, especially when it is higher than the number of instances; this is due to the likelihood function that used to estimate the network.

	Number of variables $p = 50$								
	G1-S1	G1-S2	Gennet	nlasso	RBN	nSVM-L	nSVM-R	nSVM-S	nSVM-P
TPR	0.49	0.39	0.24	0.31	0.47	0.56	0.51	0.55	**0.66**
FPR	0.22	0.16	**0.04**	0.10	0.31	0.35	0.28	0.34	0.52
TNR	0.78	0.84	**0.96**	0.90	0.69	0.65	0.72	0.66	0.48
FNR	0.51	0.61	0.76	0.69	0.53	0.44	0.49	0.45	**0.34**
MCE	0.24	0.18	**0.07**	0.13	0.32	0.35	0.29	0.35	0.51
	Number of variables $p = 100$								
TPR	0.43	0.29	0.17	0.29	0.32	0.49	0.48	0.53	**0.67**
FPR	0.17	0.07	**0.03**	0.07	0.16	0.23	0.22	0.30	0.52
TNR	0.83	0.93	**0.97**	0.93	0.84	0.77	0.78	0.70	0.48
FNR	0.57	0.71	0.83	0.71	0.68	0.51	0.52	0.47	**0.33**
MCE	0.19	0.10	**0.07**	0.10	0.19	0.24	0.24	0.31	0.52

Table 5.2. – Noninear simulated data, p=50, 100, n=20, pi=0.05, the last four columns correspond to the neighborhood SVM approach (nSVM) using different kernels (L: Linear, R: Radial, S: Sigmod, P: Polynomial).

	Number of variables $p = 50$								
	G1-S1	G1-S2	Gennet	nlasso	RBN	nSVM-L	nSVM-R	nSVM-S	nSVM-P
TPR	0.61	0.49	0.33	0.42	0.54	0.66	0.60	0.64	**0.76**
FPR	0.24	0.16	**0.03**	0.11	0.32	0.34	0.27	0.33	0.54
TNR	0.76	0.84	**0.97**	0.89	0.68	0.66	0.73	0.67	0.46
FNR	0.39	0.51	0.67	0.58	0.46	0.34	0.40	0.36	**0.24**
MCE	0.25	0.18	**0.07**	0.13	0.33	0.34	0.28	0.34	0.53
	Number of variables $p = 100$								
TPR	0.46	0.31	0.14	0.34	0.34	0.53	0.51	0.56	**0.66**
FPR	0.17	0.07	**0.02**	0.07	0.16	0.22	0.22	0.29	0.49
TNR	0.83	0.93	**0.98**	0.93	0.84	0.78	0.78	0.71	0.51
FNR	0.54	0.69	0.86	0.66	0.66	0.47	0.49	0.44	**0.34**
MCE	0.19	0.10	**0.06**	0.10	0.19	0.24	0.23	0.30	0.48

Table 5.3. – Linear and nonlinear simulated data, p=50, 100, n=20, pi=0.05, the last four columns correspond to the neighborhood SVM approach (nSVM) using different kernels (L: Linear, R: Radial, S: Sigmod, P: Polynomial).

6. Conclusion and Perspectives

Conclusion

In this dissertation, we have shown that Bayesian networks classifiers are very accurate models when compared to other classical machine learning methods. Discretising input variables often increases the performance of Bayesian networks classifiers, so does a feature selection procedure. This is probably due to the fact that discrete Bayesian networks are less sensitive to the underlying distribution of the data and are easier to estimate in low dimensions. One specific advantage of Bayesian networks classifiers is that they directly estimate the distribution of the data and take into account the high-order interactions between the variables. The models may also be graphically presented.

SVM criterion is an efficient approach for finding the interaction points to approximate the structure of the data sets. The ability to use different types of kernels allow SVMs to find the best active sets that construct the model according to the types of the input data. Also, SVM is not much sensitive to the number of variables inserted, as we have shown in nSVM-S.

Perspectives

First, future work should aim to use a more specific discretisation approach, preserving the dependencies structure within the original data and including the latent variables accounting for hidden clusters in the data to obtain more accurate results on classification using Bayesian networks.

Second, further work should aim to apply the results obtained from our approaches applied to infer interactions networks on gene expression data and compare it with approximate true structures.

Appendices

A.List of Publications

1. Jebreen, K. and Ghattas, B. (2016). Bayesian network classification: Application to epilepsy type prediction using PET scan data. In 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), pages 965–970.
2. Jebreen, K. and Ghattas, B. (2017). Inferring linear and nonlinear Interaction networks using neighborhood support vector machines, ICMLA, CANCUN, MEXICO, (Accepted).

Bibliography

Akaike, H. (1973). {Information theory and an extension of the maximum likelihood principle}. In Petrov, B. and Csaki, F., editors, {*Information theory and an extension of the maximum likelihood principle*}, pages 267–281. Akadémiai Kiado.

Altay, G. and Emmert-Streib, F. (2010). Inferring the conservative causal core of gene regulatory networks. *BMC Systems Biology*, 4:132.

Ambroise, C., Chiquet, J., and Matias, C. (2009). Inferring sparse gaussian graphical models with latent structure. *Electronic Journal of Statistics*, 3:205–238.

Banerjee, O., Ghaoui, L. E., and d'Aspremont, A. (2008). Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516.

Basso, K., Margolin, A. A., Stolovitzky, G., Klein, U., Dalla-Favera, R., and Califano, A. (2005). Reverse engineering of regulatory networks in human b cells. *Nature Genetics*, 37(4):382–390.

Beal, M. J., Falciani, F., Ghahramani, Z., Rangel, C., and Wild, D. L. (2005). A bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics (Oxford, England)*, 21(3):349–356.

Bielza, C. and Larrañaga, P. (2014). Discrete bayesian network classifiers: A survey. *ACM Comput. Surv.*, 47(1):5:1–5:43.

Breiman, L. (2001). Random Forests. *Mach. Learn.*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and Regression Trees*. Chapman and Hall/CRC, 1 edition edition.

Butte, A. J., Tamayo, P., Slonim, D., Golub, T. R., and Kohane, I. S. (2000). Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. *Proceedings of the National Academy of Sciences of the United States of America*, 97(22):12182–12186.

Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467.

Cooper, G. F. and Herskovits, E. (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347.

Cox, D. R. and Wermuth, N. (1996). Multivariate dependencies: Models, analysis and interpretation. *CRC Press*.

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Dempster, A. P. (1972). Covariance Selection. *Biometrics*, 28(1):157–175.

Díaz-Uriarte, R. and Alvarez de Andrés, S. (2006). Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7:3.

Edera, A., Strappa, Y., and Bromberg, F. (2014). The Grow-Shrink strategy for learning Markov network structures constrained by context-specific independences. *arXiv:1407.8088 [cs]*.

Edwards, D. (2000). Introduction to graphical modelling. *Springer*.

Efron, B. (2007). Size, power and false discovery rates. *The Annals of Statistics*, 35(4):1351–1377.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, 32(2):407–499.

Efron, B. and Morris, C. (1973). Stein's estimation rule and its competitors–an empirical bayes approach. *Journal of the American Statistical Association*, 68(341):117–130.

Faith, J. J., Hayete, B., Thaden, J. T., Mogno, I., Wierzbowski, J., Cottarel, G., Kasif, S., Collins, J. J., and Gardner, T. S. (2007). Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLOS Biology*, 5(1):e8.

Fayyad, U. and Irani, K. (1993). Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. *In 13th International Joint Conference on Uncertainly in Artificial Intelligence(IJCAI93)*, pages 1022–1029.

Fletcher, R. (1987). *Practical Methods of Optimization; (2Nd Ed.)*. Wiley-Interscience.

Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics (Oxford, England)*, 9(3):432–441.

Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, 29(2):131–163.

Friedman, N., Murphy, K., and Russell, S. (1998). Learning the structure of dynamic probabilistic networks. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, pages 139–147. Morgan Kaufmann Publishers Inc.

Fu, W. J. (1998). Penalized regressions: The bridge versus the lasso. *Journal of Computational and Graphical Statistics*, 7(3):397–416.

Fujita, A., Sato, J. R., Garay-Malpartida, H. M., Sogayar, M. C., Ferreira, C. E., and Miyano, S. (2008). Modeling nonlinear gene regulatory networks from time series gene expression data. *Journal of Bioinformatics and Computational Biology*, 6(5):961–979.

Geiger, D. and Heckerman, D. (1996). Knowledge representation and inference in similarity networks and bayesian multinets. *Artificial Intelligence*, 82(1):45–74.

Guedj, E., Bonini, F., Gavaret, M., Trébuchon, A., Aubert, S., Boucekine, M., Boyer, L., Carron, R., McGonigal, A., and Bartolomei, F. (2015). 18fdg-PET in different subtypes of temporal lobe epilepsy: SEEG validation and predictive value. *Epilepsia*, 56(3):414–421.

Hartemink, A. J. A. J. (2001). Principled computational methods for the validation discovery of genetic regulatory networks. *Massachusetts Institute of Technology*.

Hastie, T., Tibshirani, R., and Friedman, J. (2016). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer, 2nd edition edition.

Heckerman, D. (1991). Probabilistic Similarity Networks. *Networks*.

Heckerman, D. (1997). Bayesian networks for data mining. *Data Mining and Knowledge Discovery*, 1(1):79–119.

Heckerman, D., Geiger, D., and Chickering, D. M. (1995). Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243.

Hettigoda, S. (2016). Computation of least angle regression coefficient profiles and LASSO estimates. *Electronic Theses and Dissertations*.

Hogg, R. (1978). *Introduction to Mathematical Statistics (4th) Fourth Edition*. Macmillan, fourth edition edition edition.

Hotelling, H. (1953). New Light on the Correlation Coefficient and its Transforms. *Journal of the Royal Statistical Society. Series B (Methodological)*, 15(2):193–232.

Hruschka, E. R., Hruschka, E. R., and Ebecken, N. F. F. (2004). Feature Selection by Bayesian Networks. In *Advances in Artificial Intelligence*, pages 370–379. Springer, Berlin, Heidelberg.

Imoto, S., Goto, T., and Miyano, S. (2002). Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 175–186.

Ishak, A. B. (2007). *Sélection de Variables par les machines à vecteurs supports pour la discrimination binaire et multiclasse en grande dimension.* PhD thesis, Aix-Marseille université and Tunis université.

Jebreen, K. and Ghattas, B. (2016). Bayesian network classification: Application to epilepsy type prediction using PET scan data. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 965–970.

Johnson, R. A. and Bhattacharyya, G. K. (2014). *Statistics: Principles and Methods*. Wiley, 7 edition edition.

Johnson, R. A. and Wichern, D. W. (2007). *Applied Multivariate Statistical Analysis*. Pearson, 6 edition edition.

Jordan, M. I. (1998). *Learning in Graphical Models*. A Bradford Book, 1st edition edition.

Karatzoglou, A., Meyer, D., and Hornik, K. (2006). Support vector machines in r. *Journal of Statistical Software*, 015.

Karush, W. (1939). Minima of functions of several variables with inequalities as side constraints. *Dept.∼of Mathematics, Univ.∼of Chicago*.

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 1 edition edition.

Kononenko, I. (1994). Estimating attributes: Analysis and extensions of RELIEF. In *Machine Learning: ECML-94*, pages 171–182. Springer, Berlin, Heidelberg.

Korb, K. B. and Nicholson, A. E. (2010). *Bayesian Artificial Intelligence, Second Edition*. CRC Press, 2 edition edition.

Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50.

Kuhn, H. W. and Tucker, A. W. (1951). *Nonlinear Programming*. The Regents of the University of California.

Lam, W. and Bacchus, F. (1994). Learning bayesian belief networks: An approach based on the mdl principle. *Computational Intelligence*, 10(3):269–293.

Langley, P. and Sage, S. (1994). Induction of selective bayesian classifiers. In *Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence*, UAI'94, pages 399–406. Morgan Kaufmann Publishers Inc.

Lauritzen, S. L. (1996). *Graphical Models*. Clarendon Press, 1 edition edition.

Lin, C.-J. (2008). Feature ranking using linear svm. *Causation and Prediction Challenge Challenges in Machine Learning, Volume 2*.

Lèbre, S. (2009). Inferring Dynamic Genetic Networks with Low Order Independencies. *Statistical Applications in Genetics and Molecular Biology*, 8(1):1–38.

Mangasarian, O. L. (1969). *Nonlinear Programming*. Society for Industrial and Applied Mathematics.

Maron, M. E. and Kuhns, J. L. (1960). On relevance, probabilistic indexing and information retrieval. *J. ACM*, 7(3):216–244.

Martínez, A. P. (2010). *Supervised classification in continuous domains with bayesian networks*. http://purl.org/dc/dcmitype/Text, Universidad del País Vasco - Euskal Herriko Unibertsitatea.

McCormick, G. P. (1983). *Nonlinear Programming: Theory, Algorithms, and Applications*. John Wiley & Sons, Inc.

Meinshausen, N. and Bühlmann, P. (2006). High dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3):1436–1462.

Meyer, P. E., Kontos, K., Lafitte, F., and Bontempi, G. (2007). Information-theoretic inference of large transcriptional regulatory networks. *EURASIP journal on bioinformatics & systems biology*, page 79879.

Minsky, M. (1961). Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30.

Mitchell, T. M. (1990). *Machine Learning*. PN, 1st edition edition.

Mumford, J. A. and Ramsey, J. D. (2014). Bayesian networks for fMRI: A primer. *NeuroImage*, 86:573–582.

Murphy, K. and Mian, S. (1999). Modelling gene expression data using dynamic bayesian networks. *Technical report, Computer Science Division, University of California, Berkeley, CA., 1999*.

Nagarajan, R., Scutari, M., and Lèbre, S. (2014). *Bayesian Networks in R: with Applications in Systems Biology*. Springer Science & Business Media.

Neapolitan, R. E. (2003). *Learning Bayesian Networks*. Pearson, Upper Saddle River, NJ.

Opgen-rhein, R. and Strimmer, K. (2006). Using regularized dynamic correlation to infer gene dependency networks from time-series microarray data. In *In Proceedings of the 4th International Workshop on Computational Systems Biology (WCSB 2006*, pages 12–13.

Opgen-Rhein, R. and Strimmer, K. (2007). Learning causal networks from systems biology time course data: an effective model selection procedure for the vector autoregressive process. *BMC Bioinformatics*, 8:S3.

Pazzani, M. and Billsus, D. (1997). Learning and revising user profiles: The identification of interesting web sites. *Machine Learning*, 27(3):313–331.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1 edition edition.

Pearl, J. (2009). Causality. *Cambridge University Press*.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238.

Rakotomamonjy, A. (2003). Variable selection using svm based criteria. *J. Mach. Learn. Res.*, 3:1357–1370.

Ramsey, J. D., Hanson, S. J., Hanson, C., Halchenko, Y. O., Poldrack, R. A., and Glymour, C. (2010). Six problems for causal inference from fMRI. *NeuroImage*, 49(2):1545–1558.

Rau, A., Jaffrézic, F., Foulley, J.-L., and Doerge, R. W. (2010). An empirical bayesian method for estimating biological networks from temporal microarray data. *Statistical Applications in Genetics and Molecular Biology*, 9:Article 9.

Reunanen, J. (2003). Overfitting in Making Comparisons Between Variable Selection Methods. *Journal of Machine Learning Research*, 3(Mar):1371–1382.

Reverter, A. and Chan, E. K. F. (2008). Combining partial correlation and an information theory approach to the reversed engineering of gene co-expression networks. *Bioinformatics (Oxford, England)*, 24(21):2491–2497.

Rissanen, J. (2007). *Information and Complexity in Statistical Modeling*. Springer Publishing Company, Incorporated, 1 edition.

Santafé, G. (2010). *Advances in Supervised and Unsupervised Learning of Bayesian Networks: Application to Population Genetics*. LAP LAMBERT Academic Publishing.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464.

Schäfer, J. and Strimmer, K. (2005a). An empirical bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6):754–764.

Schäfer, J. and Strimmer, K. (2005b). Learning Large-Scale Graphical Gaussian Models from Genomic Data. In *In Science of Complex Networks: From Biology to the Internet and WWW*.

Schäfer, J. and Strimmer, K. (2005c). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4:Article32.

Scutari, M. and Denis, J.-B. (2014). *Bayesian Networks: With Examples in R*. Chapman and Hall/CRC, 1 edition edition.

Sikonja, M. R. and Kononenko, I. (1995). Discretization of continuous attributes using relieff. *Proceedings of ERK'95*, pages 1022–1029.

Smith, S. M., Miller, K. L., Salimi-Khorshidi, G., Webster, M., Beckmann, C. F., Nichols, T. E., Ramsey, J. D., and Woolrich, M. W. (2011). Network modelling methods for FMRI. *NeuroImage*, 54(2):875–891.

Spirtes, P., Glymour, C., and Scheines, R. (2001). *Causation, Prediction, and Search, Second Edition*. A Bradford Book, second edition edition edition.

Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., and Zeileis, A. (2008). Conditional variable importance for random forests. *BMC Bioinformatics*, 9:307.

Suzuki, J. (1993). A construction of bayesian networks from databases based on an MDL principle. *Proceedings of the Ninth International Conference on Uncertainty in Artificial Intelligence*, pages 266–273.

Tibshirani, R. (1994). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc.

Verma, T. and Pearl, J. (1991). *Equivalence and Synthesis of Causal Models*. UAI '90. Elsevier Science Inc., New York, NY, USA.

Whittaker, J. (1990). *Graphical Models in Applied Multivariate Statistics*. Wiley, Chichester England ; New York, 1 edition edition.

Yu, H. and Kim, S. (2012). SVM tutorial — classification, regression and ranking. In Rozenberg, G., Bäck, T., and Kok, J. N., editors, *Handbook of Natural Computing*, pages 479–506. Springer Berlin Heidelberg.