Information-Theoretic Variable Selection and Network Inference from Microarray Data

Thèse présentée par
Patrick Emmanuel Meyer

En vue de l'obtention du grade de
Docteur en Sciences
(Département d'Informatique, Faculté des Sciences)

de l'Université Libre de Bruxelles
Bruxelles, Belgique

2008
(Defended december 16)

This thesis has been written under the supervision of Prof. Gianluca Bontempi.
The members of the Jury are:

- Prof. Gianluca Bontempi (Université Libre de Bruxelles, Belgium)
- Prof. Jean Cardinal (Université Libre de Bruxelles, Belgium)
- Prof. Timothy Gardner (Boston University, USA)
- Prof. Tom Lenaerts (Université Libre de Bruxelles, Belgium)
- Prof. Fabrice Rossi (École Nationale Supérieure des Télécommunications, France)
- Prof. Michel Verleysen (Université Catholique de Louvain, Belgium)

In memory of my beloved mother.

The question of whether a computer can think is no more interesting than the question of whether a submarine can swim.

E. W. Dijkstra [37]

Abstract

Statisticians are used to model interactions between variables on the basis of observed data. In a lot of emerging fields, like bioinformatics, they are confronted with datasets having thousands of variables, a lot of noise, non-linear dependencies and, only, tens of samples. The detection of functional relationships, when such uncertainty is contained in data, constitutes a major challenge.

Our work focuses on variable selection and network inference from datasets having many variables and few samples (high variable-to-sample ratio), such as microarray data. Variable selection is the topic of machine learning whose objective is to select, among a set of input variables, those that lead to the best predictive model. The application of variable selection methods to gene expression data allows, for example, to improve cancer diagnosis and prognosis by identifying a new molecular signature of the disease. Network inference consists in representing the dependencies between the variables of a dataset by a graph. Hence, when applied to microarray data, network inference can reverse-engineer the transcriptional regulatory network of cell in view of discovering new drug targets to cure diseases.

In this work, two original tools are proposed MASSIVE (Matrix of Average Sub-Subset Information for Variable Elimination) a new method of feature selection and MRNET (Minimum Redundancy NETwork), a new algorithm of network inference. Both tools rely on the computation of mutual information, an information-theoretic measure of dependency. More precisely, MASSIVE and MRNET use approximations of the mutual information between a subset of variables and a target variable based on combinations of mutual informations between sub-subsets of variables and the target. The used approximations allow to estimate a series of low variate densities instead of one large multivariate density. Low variate densities are well-suited for dealing with high variable-to-sample ratio datasets, since they are rather cheap in terms of computational cost and they do not require a large amount of samples in order to be estimated accurately. Numerous experimental results show the competitiveness of these new approaches. Finally, our thesis has led to a freely available source code of MASSIVE and an open-source R and Bioconductor package of network inference.

Acknowledgments

Ma première pensée va à ma Maman, à qui je dédie ma thèse. J'aurais tant aimé qu'elle me conseille pour ce travail comme elle a toujours su si bien le faire dans tant de domaines.

Ensuite, je tiens à remercier tous les membres de ma famille. Leur soutien inconditionnel m'a été d'une aide inestimable tout au long de ma vie. Je remercie tout d'abord mon Papa. Ensuite, je me permets d'utiliser l'ordre alphabétique: merci aux Abramowicz, Adam, Collignon, Hirsch, Meyer et Samdam.

Je voudrais aussi exprimer toute ma gratitude à

Ma merveilleuse Déborah qui me soutient si bien.

Mon ami Oscar sur qui je peux compter depuis des années.

Mes collègues et amis du département d'informatique qui m'ont régulièrement enrichi de leurs avis. J'espère qu'ils ne m'en veulent pas trop pour les nombreuses occasions où je les ai détournés de leurs propres travaux. Merci à Eythan, Olivier, Yann-Aël, Abhilash, Catharina, Daniele, Gilles, Benjamin, Gwenaël et tous les autres.

After my family and friends, I would like to thank those that contributed more directly to this thesis.

The first one is Gianluca Bontempi, my supervisor, who has taught me the tools of the trade. He has been extremely generous with his time. By his side, I have learnt how to perform experiments, how to write a paper, how to answer to reviewers. He is the one who steered me through the minefield that the fledgling scientist must cross.

I am also grateful to many of my coworkers, who collaborated on papers: Frédéric Lafitte, Colas Schretter, Olivier Caelen, Catharina Olsen, Yann-Aël Le Borgne, Abhilash Miranda, Mauro Birattari and Professor Jacques Van Helden. Without them my publications would not be what they are (whatever that could mean).

I also thank my father. He has contributed to this thesis in two different ways. First, his way with doing research has been a source of inspiration. Secondly, he has often helped me with my approximate English.

I have been very lucky to benefit, beside my colleagues, jury committee and supervisor, from the careful proofreading of two sharp scientific minds: Professors Ariane Szafarz and Raymond Devillers. They have been generous with their time, advice and huge amount of ink spent on my successive drafts.

Last but not least, I would like to thank the members of the jury who have accepted to comment this work, namely Professors Timothy Gardner (from Boston University, whose teaching during the Bristol summer school showed me the way), Fabrice Rossi (Telecom ParisTech, France), Michel Verleysen (UCL, Belgium), Jean Cardinal (ULB, Belgium) and Tom Lenaerts (ULB, Belgium, whose advice have been helpful). They all probably ignore how much their interest in information theory has influenced my research. Their publications have clearly increased the entropy of my desk and the information of my work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This work focuses on methods of variable selection and network inference using mutual information, and their applications to microarray data analysis. In particular, we study the notions of relevance, redundancy and complementarity of variables in order to design new data-mining tools.

The increasing possibilities of collecting massive data are raising a major issue in science: the extraction of information. A promising field to solve this issue is *data-mining* [47] which is an automatized process of knowledge extraction from databases. Nowadays, data-mining techniques are widely used. Fraud detection, credit risk analysis, sales prediction, clients classification, texts, sounds, images or videos identification are among the many problems that are dealt within this area of research [25, 139, 92, 62].

One major field of data-mining is the analysis of biological data, such as the ones collected by *microarrays* [135, 104]. A microarray is a chip composed of thousands of microscopic dots that measures *gene expression* or *gene activity* [59]. Gene expression analysis leads to a better understanding of cells and their genomes [2]. As a consequence, this research area has as goal to discover new diagnosis tools and new targets of treatments for diseases such as cancers [4, 134, 51, 138].

From the data-miner point of view, microarray datasets are challenging, because they are often made of few samples (the cells) and thousands of variables (the gene expressions) measured using a (biological) process known to be noisy. Furthermore, a lot of known gene relationships are non-linear and multivariate[15]. Finally, biomedical scientists are expecting methods that produce *intelligible/understandable* models (by opposition to black-box).

Hence, an effective microarray analysis tool should be able to,

1. return **intelligible/understandable** models,
2. rely on **little a priori knowledge**,

3. deal with **thousands of variables**,
4. detect **non-linear dependencies**,
5. deal with **tens of noisy samples**.

In this work, we focus on *information-theoretic methods* of *variable selection and network inference* because 1) they fulfill the requirements enumerated above, and 2) they are known to be effective [46, 83, 104, 148].

Variable selection is the subdomain of data-mining whose objective is to select, among a set of input variables, those that lead to the best predictive model [57]. Applied to microarray data, variable selection returns a set of genes whose particular expressions characterize the state of a cell. This makes possible the identification of a *cell signature* and it is used for diagnosis, i.e., differentiating malign tumor cells from benign ones, but also for prognosis, i.e., detecting tumor cells sensitive to chemotherapy vs tumor cells not responding to the treatment[136, 138].

Network inference is another area of data-mining. It purports to represent the dependencies between the variables of a dataset with a graph [140]. Applied to gene expression data, network inference returns a graph where arcs are meant to represent gene-gene interactions. This is called *transcriptional regulatory network* inference and it purport to give a global picture of the (regulatory network) cell [51, 135]. The inferred network can be used to identify genes that could be targeted by new treatments[4, 51].

In order to deal with a large amount of variables that have non-linear dependencies, an adequate relevance measure is required. Mutual information, which is an information-theoretic measure of dependency, is chosen. This choice is partly motivated by successful approaches [46, 83, 104, 49].

At this point, we have methods (variable selection and network inference) that return intelligible outputs (cell signature and transcriptional regulatory network), able to deal with non-linearities and large number of variables (using mutual information). However, one problem remains: the **limited amount of samples**. Many state-of-the-art approaches resolve this problem by estimating mutual information between only two random variables [44, 104, 148, 19]. However, it will be shown that resorting to low-variate mutual information neglects variable interactions such as redundancies and complementarities.

Our contribution is to provide a new information-theoretic variable selection and network inference technique able of dealing with variable interactions, without requiring a large number of samples. More precisely, our methods use approximations of the mutual information between a subset of variables and a target variable, based on combinations of mutual informations between sub-subsets of variables and the target.

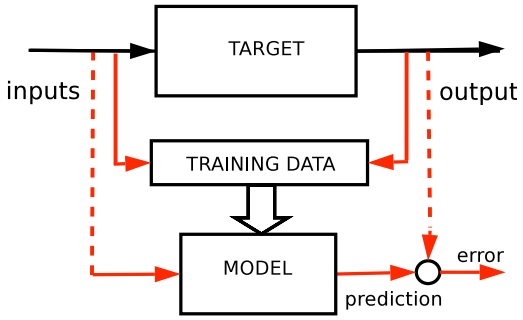


Figure 1.1: Data-mining through supervised learning consists in building a model from data in order to predict a target function.

We shall now hark back to the notions that have been mentioned above and that we treat in our work: the field of data-mining (in Section 1.1), the problems raised by microarray data (in Section 1.2), the approach of variable selection (in Section 1.3) and network inference (in Section 1.4), the strong points of information-theoretic methods (in Section 1.5), the main problem of state-of-the art methods and our own solutions and claims (in Section 1.6).

1.1 Data-mining

Data-mining typically uses *machine learning* to predict and/or understand variables of a dataset. Machine Learning is defined as *the study of computer algorithms that improve automatically through experience* [92]. This field lies at the frontier of statistics and computer science. Modelling a target function connecting the variables of a system is a machine learning task. The output or target variable, denoted by Y, is the one to be predicted and the input variables, denoted by X, are the predictors. The estimated relationship between X and Y is called a *statistical or a predictive model*.

The learning is supervised when the machine is trained with output and inputs values. The aim of supervised machine learning is to estimate an input/output model from the data in view of predicting the output value on new data (see Figure 1.1). It is a *regression* task, if the output is a continuous value and a *classification* task if the output is a class label. In this work, input and output variables are discrete (or discretized). Hence, we focus on supervised classification rather than regression.

Machine learning algorithms can be applied to various domains of research with little tuning. Nowadays, data-mining methods are increasingly used in businesses and industries [25].

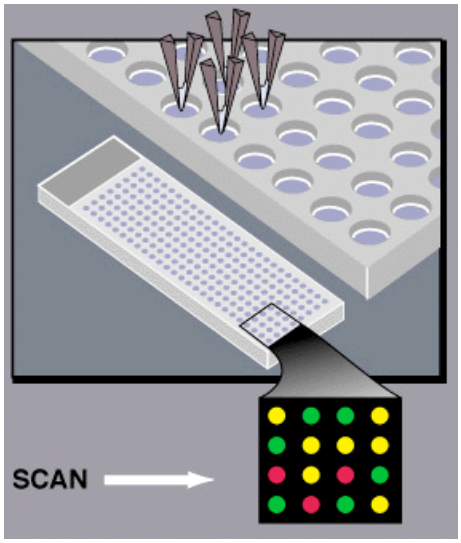


Figure 1.2: Microarray platform [nhg].

1.2 Microarray Data

A particular type of biological data, called microarray, has given rise to a challenging problem in data-mining. At first, let us introduce some biological notions[1].

A cell is made of millions of proteins. Proteins are generated from the DNA (the genes). Every cell of the same (multicellular) organism contains the same DNA, but cells can be very different (for example, a tumor cell vs a normal cell) because of the different concentrations of each protein. A protein is a molecular machine that achieves some specific action. Each situation faced by the cell requires different proteins. The amount of each type of protein in the cell is continuously adapted as a function of the environment [2, 59]. Hence, by measuring the protein quantities of a cell in different situations, a model of the cell can be estimated.

A microarray is a blade composed of thousands of microscopic dots containing DNA oligonucleotides (Figure 1.2). Each dot measures a *gene expression* or *gene activity*, which occurs as an intermediary step in the production process of proteins. By using this technology, genome-wide patterns of gene expression can be studied. In other words, a microarray represents the state of a cell by a vector where each dimension is a different gene and each component of the vector quantifies the activity of that gene in a cell. As a result, microarray analysis is a key issue toward the understanding of genomics and cell biology.

These measures suffer nonetheless from several drawbacks. First, the data are noisy because of the highly variable biological process measured [59]. Second, microarray exper-

[1] A more detailed biological background is available in Appendix A.1.

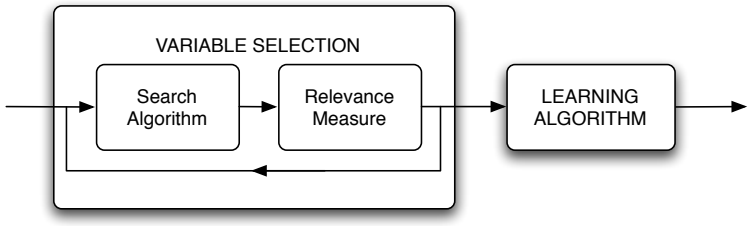


Figure 1.3: Variable selection is a preprocessing step of learning composed of a search algorithm combined with an evaluation function.

iments are expensive (around one thousand euros per experiment). For this reason, the number of samples in datasets is relatively low. The fact that the data are noisy, and that there are a small number of samples, make it difficult to estimate a statistical model. If biomedical scientists have some knowledge of the underlying process that has generated the data, they generally use that knowledge to validate the inferred model, rather than as a priori knowledge on it. This is why there is a need for data-mining techniques, able to return intelligible outputs (for biomedical scientists) dealing with thousands of variables, tens of noisy samples, non-linear dependencies between variables and little a priori knowledge.

1.3 Variable Selection

In order to produce such intelligible models, variable selection techniques are often used. *Variable selection* is the subdomain of data-mining whose goal is to select inputs among a set of variables which lead to the best predictive model. In most cases, it is a preprocessing step to learning (see Figure 1.3). On the one hand, eliminating variables can lead to information losses. On the other hand, as shown in [11, 74, 76, 56], selecting variables can:

1. improve the accuracy of a model (by improving the ability of generalizing).
2. increase the intelligibility of the model.
3. decrease the time and the memory taken by the learning and utilization of the model.

Algorithms for variable selection are typically made of a search algorithm exploring different combinations of variables and of a performance measure evaluating the selected variables. In this work, we focus on sequential search algorithms (rather than stochastic search) and performance measures based on information theory because both are fast, efficient and quite easy to understand for non-specialists.

A problem generally encountered in microarray analysis is the prediction of effects of medical treatments on patients. The predicted variable Y expresses the reaction of the patient to treatment, (for instance, 1 if the treatment succeeded and 0 if not) and the source of input data X for that kind of problems could be the expression of the genes in a tumor cell tissue. Recently, by using microarray techniques, a 70-gene prognosis profile has been identified as a predictor of early development of a distant metastasis in young patients [136]. This profile has been generated using only 78 tumor samples of patients and has been shown to outperform all clinical parameters in predicting distant metastasis [134].

The issue of cell/patient classification can be addressed, by selecting the genes whose activities allow to discriminate cells or patients. This is an example of a *variable selection task*.

1.4 Network Inference

Network inference is another data-mining technique that returns intelligible models. It represents dependencies between variables in a dataset with a graph [140]. Each variable of the dataset is represented by a node in the graph. There is a link between two variables, if these variables exhibit a particular form of dependency (the form of dependency depends on the inference methods).

A gene can produce a protein that can activate or repress the production of another protein (see Appendix A.1.2). Hence, there are circuits coded in the DNA of a cell. A convenient representation of the cell circuitry is a graph, where the nodes represent the genes and the arcs represent the interactions between them. As a result, a network inferred from microarray data can be interpreted as a transcriptional regulatory network. Transcriptional regulatory network inference aims at giving a global picture of the (transcriptional network) cell. Those networks also delineate the interactions between the components of a biological systems, and tell us how these interactions give rise to the functions and behaviors of that global system. This type of network inference differs from the so-called "reductionist approach" that focuses on the characteristics of the building blocks of a system rather than on the global behavior.

The reverse engineering of transcriptional regulatory networks from expression data alone appears quite difficult, because of the combinatorial nature of the problem and of the poor information of the data [135]. Moreover, by only focusing on transcript data, the inferred network should not be considered as a biochemical regulatory network, but as a

"The reductionist approach has successfully identified most of the components and many of the interactions but, unfortunately, offers no convincing concepts or methods to understand how system properties emerge...the pluralism of causes and effects in biological networks is better addressed by observing, through quantitative measures, multiple components simultaneously and by rigorous data integration with mathematical models"
Sauer et al. [117].

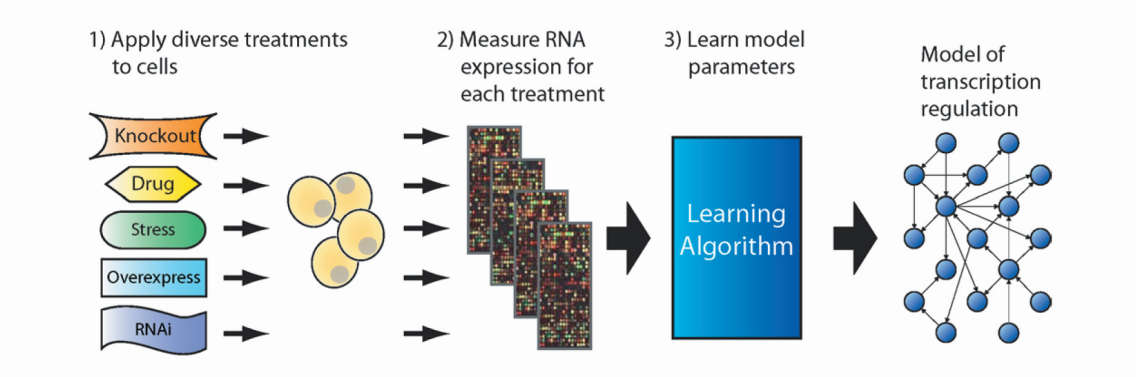


Figure 1.4: "The general strategy for reverse engineering transcription control systems: 1) The experimenter perturbs cells with various treatments to elicit distinct responses. 2) After each perturbation, the experimenter measures the expression (concentration) of many or all RNA transcripts in the cells 3) A learning algorithm calculates the parameters of a model that describes the transcription control system underlying the observed responses. The resulting model may then be used in the analysis and prediction of the control system function". Figure and caption from Gardner and Faith [51].

gene-to-gene network, where many physical connections between macromolecules might be hidden by short-cuts. In spite of these obvious limitations, the bioinformatics community has made important progresses in this domain over the last few years, by using machine learning techniques of *network inference* [51] (see Figure 1.4).

By putting a cell in different situations, we can observe the different states corresponding to each situation and infer the regulatory interactions present in the cell. Recently, the reconstruction of regulatory networks from expression profiles of human B cells has been proposed [4]. Validation of the network against available data led to the identification of a particular gene as a major hub, which controls a network containing known interactions as well as new ones which were biochemically validated.

Finally, the study of transcriptional regulatory network can lead to the discovery of new drugs that can

1. block an interaction, such as an interaction leading the cell from a normal state to a tumor state,

2. activate an interaction, such as a transition from the state "tumor cell" to the state "dead cell".

1.5 Information-Theoretic Methods

Both fields expanded above, variable selection and network inference, are subdomains of the data-mining field. However, few methods in these fields can deal with i) non-linearity and ii) large number of variables, that are present in microarray data. We therefore need to resort to more specific techniques. *Information-theoretic methods* offer an effective solution to these two issues[104, 49, 46, 83]. These methods use *mutual information,* which is an information-theoretic measure of dependency. First, mutual information is a model-independent measure of information that has been used in data analysis for defining concepts like variable *relevance* [130, 8, 75], *redundancy* [84, 141, 148] and *interaction* [84, 67, 75]. It is widely used to redefine theoretic machine learning concepts [82]. Secondly, mutual information captures non-linear dependencies, an interesting feature in biology where many biological interactions are believed to be non-linear [15, 59]. Finally, mutual information is rather fast to compute. Therefore, it can be computed a high number of times in a reasonable amount of time, as required by datasets having a large number of variables.

1.6 Problem and claim

Let us briefly summarize our generic goal and see how it is possible to solve it. In view of analyzing microarray data, our objective is to build methods able to return *intelligible* models, i.e., that biomedical scientists can directly or easily understand and manipulate. Those methods should deal with *thousands of variables, tens of noisy samples, non-linear dependencies* and *little a priori knowledge.*

- Data-mining algorithms help us to deal with little or even no a priori knowledge.
- Variable selection and network inference are data-mining techniques adequate for producing intelligible outputs (cell signature and transcriptional regulatory network).
- Mutual information is an evaluation function adapted to large datasets and non-linear relationships.

But, at this point, one problem remains: how can we deal with the limited amount of noisy samples? Mutual information is difficult to estimate accurately with too few samples[100, 97]. In order to solve this issue, state-of-the-art techniques rely on the estimation of

mutual information on two or three random variables[44, 104, 49, 19]. An instance of such an approach is the ranking approach that ranks variables by their mutual information with the output. Low-variate mutual information does not require a large amount of samples in order to be estimated accurately. Hence, it is well-suited for dealing with microarray data. However, relying on bivariate mutual information ignores informations coming from interactions between the variables such as redundancies and complementarities.

Our thesis formalizes these notions of redundancy and complementarity. We then offer a new approximation of the mutual information between a subset of variables and a target variable. More precisely, we will use approximations of the mutual information between a subset of variables and a target variable based on combinations of mutual informations between sub-subsets of variables and the target. This will give rise to two new tools that deal explicitely with variable interactions:

1. an original variable selection method, called MASSIVE (Matrix of Average Sub-Subset Information for Variable Elimination)[90], based on a new information-theoretic selection criterion (kASSI, the k-Average Sub-Subset Information) [86, 87].

2. an original network inference method, called MRNET (Minimum Redundancy NETwork), inspired by a recently proposed variable selection technique [88]. Research on MRNET has led to an open source R/Bioconductor package called MINET (Mutual Information NETworks inference) [89] that will also be described in this work.

Numerous experimental results show the competitiveness of these new approaches with state-of-the-art information-theoretic methods [90, 88, 99].

1.7 Outline

The next chapter explains the preliminary theories required for the thesis such as the statistical foundations of machine learning or the basics of information theory. The state-of-the-art in information-theoretic variable selection and network inference is discussed in the third chapter. The fourth chapter contains variable selection contributions (MASSIVE), network inference contributions (MRNET), as well as experimental contributions showing the effectiveness of these new approaches. Finally, the fifth chapter presents the conclusions of the thesis.

1.8 Contributions

1.8.1 Theoretical contributions

In this thesis, we develop an information-theoretic framework (Chapter 3) of

- variable selection theory (Section 3.1),
- causality concepts (Section 3.6.1).

We also present (Chapter 3) a state-of-the-art of

- information-theoretic variable selection algorithms (Section 3.3).
- information-theoretic network inference algorithms (Section 3.5).

Original contributions are introduced in chapter 4:

1. a new criterion for variable selection, named the kASSI (Section 4.1).
2. the exploitation of two particular cases of the kASSI in variable selection algorithms, i.e. $(k = d - 1)$ PREEST and $(k = 2)$ DISR (Section 4.2 and 4.3).
3. the MASSIVE algorithm which is an extension of DISR (Section 4.4).
4. MRNET, a new method of network inference based on variable selection (Section 4.6).
5. the open-source R/Bioconductor package MINET (Section 4.7).

1.8.2 Experimental contributions

The experimental contributions are exposed in chapter 4. These include

- a comparison of the forward selection with a variant (PREEST) on 7 datasets coming from the UCI Machine Learning datasets repository (Section 4.2.1).
- a comparison of five state-of-the-art information-theoretic variable selection methods on 11 public microarray datasets for cancer classification (Section 4.5.1).
- a comparison of MASSIVE and MRMR, two variable selection methods, on 15 syntactic datasets (Section 4.5.1).
- an experimental comparison of three state-of-the-art of information-theoretic network inference methods on 30 syntactic datasets (Section 4.8).
- an experimental comparison of entropy estimators in a network inference task using 12 synthetic datasets (Section 4.8).
- the application of network inference techniques on biological datasets (Section 4.8).

1.8.3 Software Contributions

1. The C++ code of MASSIVE, a new variable selection algorithm freely distributed on internet.

2. The open-source R/Bioconductor package MINET, conceived to infer transcriptional regulatory networks from microarray data.

1.8.4 Publications in International Peer-Reviewed Journal

1. MINET: an R/Bioconductor Package for Network Inference using Mutual Information
   Patrick E. Meyer, Frederic Lafitte and Gianluca Bontempi.
   Accepted for publication
   In BMC Bioinformatics, 2008

2. On the Impact of Entropy Estimation on Transcriptional Regulatory Network Inference Based on Mutual Information
   Catharina Olsen, Patrick E. Meyer and Gianluca Bontempi.
   Accepted for publication
   In EURASIP Journal on Bioinformatics and Systems Biology, 2008

3. Information-Theoretic Feature Selection Using Variable Complementarity in Microarray Data
   Patrick E. Meyer, Colas Schretter and Gianluca Bontempi.
   In IEEE Journal of Selected Topics in Signal Processing, Volume 2, Issue 3,
   Special Issue on Genomic and Proteomic Signal Processing, June 2008.

4. Information-Theoretic Inference of Large Transcriptional Regulatory Networks
   Patrick E. Meyer, Kevin Kontos, Frederic Lafitte and Gianluca Bontempi.
   In EURASIP Journal on Bioinformatics and Systems Biology, Volume 2007, Issue 1,
   Special Issue on Information-Theoretic Methods for Bioinformatics, June 2007.

1.8.5 Publications in Lecture Notes in Computer Science and in Books

1. A model-based relevance estimation approach for feature selection in microarray datasets
   Gianluca Bontempi and Patrick E. Meyer.
   Artificial Neural Networks (ICANN 2008).
   In Lecture Notes Computer Science, volume 5164, pp. 21-31, Springer, 2008.

2. Biological Network Inference Using Redundancy Analysis
   Patrick E. Meyer, Kevin Kontos and Gianluca Bontempi.
   1st International Conference on Bioinformatics Research and Development (BIRD 07).
   In Lecture Notes in Computer Science, volume 4414, pp. 16-27, Springer, 2007

3. On the Use of Variable Complementarity for Feature Selection in Cancer Classification
   Patrick E. Meyer and Gianluca Bontempi.
   4th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics (EvoBIO 06).
   In Lecture Notes in Computer Science, volume 3907, pp. 91-102, Springer, 2006

4. Combining Lazy Learning, Racing and Subsampling for Effective Feature Selection
   Gianluca Bontempi, Mauro Birattari and Patrick E. Meyer.
   7th International Conference on Adaptive and Natural Computing Algorithms (ICANNGA 05).
   Springer Computer Science, Springer, pp. 393-396, 2005.

5. Collective Retrieval by Autonomous Robots
   Patrick E. Meyer.
   2th Starting AI Researchers' Symposium (STAIRS 04).
   In Frontiers in Artificial Intelligence and Applications, volume 109, pp. 199-204, IOS Press, 2004.

1.8.6 Workshops - Conferences - Technical Reports

1. Inferring mutual information networks using the minet package
   Patrick E. Meyer, Frédéric Lafitte and Gianluca Bontempi.
   In R/Bioconductor package - R news may 2008, volume 8.

2. On the Impact of Entropy Estimator in Transcriptional Regulatory Network Inference
   Catharina Olsen, Patrick E. Meyer and Gianluca Bontempi.
   In Proceedings of the Fifth International Workshop on Computational Systems Biology (WCSB 2008).

3. Speeding up Feature Selection by Using an Information Theoretic Bound
   Patrick E. Meyer, Olivier Caelen and Gianluca Bontempi.

In Proceedings of the 17th Belgian-Dutch Conference on Artificial Intelligence (BNAIC 05).

4. Information-Theoretic Filters for Feature Selection
   Patrick E. Meyer.
   Technical Report nr 548 of the Computer Science Department of the Université Libre de Bruxelles, 2005.

5. Information-Theoretic Feature Selection Using Variable Complementarity.
   Patrick E. Meyer, Colas Schretter and Gianluca Bontempi.
   Technical Report nr 575 of the Computer Science Department of the Université Libre de Bruxelles, 2007.

1.9 Notations

Throughout the thesis, random variables are written with capital letters (Y) and their realizations are in lower-case letters (y). Statistical estimations wear a hat ($\hat{Y}$) or have an index m (θ_m). $p(Y)$ denotes the probability distribution of the random variable Y (with some abuse of notation). Matrices and vectors are also in capital letters (T) and their elements are in small letters with indices (t_{ij}).

1.9.1 Probability and information theory

- Y: unidimensional discrete random variable representing the output, or the target, of a model
- $\mathcal{Y}$: the domain of the random variable Y
- y: a realization of the random variable Y
- X: a multidimensional discrete random variable representing the inputs of the model
- $\mathcal{X}$: the domain of the random variable X
- x : a realization of X (in the discrete set $\mathcal{X}$)
- $p(y)$: probability that the discrete random variable Y takes the value y, i.e., $P(Y = y)$
- $p(Y)$: probability mass distribution of the random variable Y
- $\tilde{Y}$: discrete random variable also defined on the domain $\mathcal{Y}$

- $\tilde{p}(y)$:$P(\tilde{Y} = y)$
- $\tilde{p}(Y)$: probability mass distribution of the random variable $\tilde{Y}$
- $u(Y)$: uniform probability mass distribution defined on the domain $\mathcal{Y}$, i.e., $P(U = y) = \frac{1}{|\mathcal{Y}|}, y \in \mathcal{Y}$
- $|\mathcal{X}|$: the number of elements of the set $\mathcal{X}$
- $p(y|x)$: the conditional probability that $Y = y$ knowing that $X = x$
- $p(Y|x)$: the conditional distribution function of the target Y knowing that $X = x$
- $p(Y|X)$: the conditional distribution function of Y given X
- $L^*(X)$: Bayes error on the input X
- $\mathrm{KL}(p; q)$: Kullback-Leibler divergence
- $\mathrm{LL}(p)$: negative log-likelihood or cross-entropy
- $H(p)\, or\, H(X)$: entropy
- $H(Y|X)$: conditional entropy
- $I(X; Y)$: mutual information
- $I(X; Y|X)$: conditional mutual information
- $R(X; Y; Z)$: multiinformation
- $C(X; Y; Z)$: Interaction information

1.9.2 Model / Estimation

- Λ: a parametric class of models
- Θ: set of parameters
- θ: vector of parameters ($\theta \in \Theta$)
- D_m: the dataset, composed of m realizations $(y_r, x_r), r \in \{1, 2, ..., m\}$ of the random variables (Y, X)
- m: number of samples in the dataset

- x_r: with $r \in \{1, 2, ..., m\}$, denotes the rth realization of the variable X in the dataset D_m
- $\hat{Y}$: statistical estimation of Y
- θ_m: estimation of the parameters θ from D_m
- θ_{-K}: estimation of the parameters θ from $D_m \backslash K$ with K a subset of the dataset D_m

- $\hat{p}(y \mid \theta_m, x)$: estimation of the conditional probability that $Y = y$, knowing that $X = x$ using parameters θ_m
- $\hat{p}(Y|\theta_m, X)$: the estimated conditional probability distribution of Y given X
- $\hat{y} = g(x, \theta_m)$: prediction of the value of Y at input point x (typically $\hat{y} = g(x, \theta_m) = \arg\max_{y \in \mathcal{Y}} \hat{p}(y \mid \theta_m, x)$)
- $E_X[f]$ *or* $E_{p(X)}[f]$: expectation of $f : \mathcal{X} \to \mathbb{R}$ averaged over all values of X
- $E_{D_m}[g]$: expectation of $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ averaged over all datasets of size m
- $\#(x)$: the number of times that the value $X = x$ is observed in the dataset D_m
- $R_m(\theta)$: empirical risk function
- $L(y_r, g(x_r, \theta))$: loss function
- $\arg\max_\theta$: argument of the maximum

1.9.3 Variable Set Notation

- n: dimension of the input space $\mathcal{X}$, i.e., $X = (X_1, X_2, ..., X_n)$
- A: set of indices denoting the variables in X, i.e., $A = \{1, 2, ..., n\}$
- X_j : jth component of the vector X ($j \in A$)
- $\mathcal{X}_j$: domain of X_j
- x_j: a realization of the input variables X_j
- x_{r_j}: with $j \in A$ and $r \in \{1, 2, ..., m\}$ denotes the rth realization (in D_m) of the jth input variables X_j
- $X_{i,j}$: subset of variables composed of X_i and X_j, i.e., $X_{i,j} = (X_i, X_j)$

- X_{-i}: subset of variables composed of all the variables in X except the variable X_i
- X_S: subset of variables selected by a selection method
- X_R: subset of remaining variables in a variable selection method
- X_{M_i}: subset of variables denoting a Markov blanket of a variable X_i
- $X_{-(i,S)}$: subset of variables composed of all the variables in X except the variable X_i and those in X_S ($X_i \notin X_S$)
- X_{S-i}: subset of variables composed of all the variables in X_S except the variable X_i ($X_i \in X_S$)

1.9.4 Network inference notation

- T: adjacency matrix of the true (reference) network (with elements t_{ij})
- W: weighted adjacency matrix of a network having as elements w_{ij}
- θ: threshold (parameter) of a network inference algorithm
- $\hat{T}$: adjacency matrix of an inferred network, usually obtained by thresholding W, i.e., $\hat{t}_{ij}(\theta) = \begin{cases} 0 & \text{if} \quad w_{ij} < \theta \\ 1 & \text{otherwise} \end{cases}$
- CM: confusion matrix with elements cm_{ij}
- MIM: matrix of (paired) mutual information with elements $\text{mim}_{ij} = I(X_i; X_j)$
- tpr: true positive rate
- fpr: false positive rate
- pr: precision
- re: recall

1.9.5 Acronyms

- FCBF: Fast Correlation Based Filter, variable selection method.
- REL: Relevance criterion, variable evaluation function.
- RANK: Ranking algorithm, variable selection method.

- CMIM: Conditional Mutual Information Minimization, variable selection method.
- MRMR: Minimum Redundancy - Maximum Relevance, variable selection method.
- kASSI: k-Average Sub-Subset Information, subset evaluation function.
- PREEST: Algorithm based on a pre-estimation of the mutual information.
- DISR: Double Input Symetrical Relevance, subset evaluation function.
- MASSIVE: Matrix of Average Sub-Subset Information for Variable Elimination, variable selection method.
- RELNET: Relevance Network, network inference method.
- CLR: Context Likelihood of Relatedness, network inference method.
- ARACNE: Algorithm for the Reconstruction of Accurate Cellular NEtworks, network inference method.
- MRNET: Minimum Redundancy Network, network inference method.
- MINET: Mutual Information Networks, a R/Bioconductor package for network inference.

Chapter 2

Preliminaries

This chapter is composed of three parts. The first one (Sections 2.1, 2.2 and 2.3) introduces some information-theoretic measures required for this thesis. The second part (Section 2.4) addresses the statistical foundations of pattern recognition. The third part (Sections 2.5, 2.6 and 2.7) discusses various algorithms of machine learning, entropy estimation and discretization methods. The notions of information theory and pattern recognition are required in order to define formally variable selection and network inference. Machine learning algorithms, entropy estimation methods and discretization methods are tools that will be used in the experimental sessions of this work.

2.1 Elements of Information Theory

Claude E. Shannon [124], the father of information theory, introduced in 1948, a theory of data transmission and signal compression. Later, notions like entropy, mutual information or interaction information gained wide interest in areas such as statistics, physics or economics [26].

This section defines the notions of Kullback-Leibler divergence, entropy and mutual information for discrete variables. We will also show how these notions can be used to define conditional mutual information, interaction information and multiinformation which will be used, in chapter 3, in order to define relevance, redundancy and complementarity of variables. Elementary concepts of probability theory needed for what follows are given in Appendix B.1.

Let X and Y denote two discrete random variables having

- Finite alphabet $\mathcal{X}$ and $\mathcal{Y}$,
- Joint probability mass distribution $p(X, Y)$,
- Marginal probability mass distributions $p(X)$ and $p(Y)$,

Let $\tilde{X}$ and $\tilde{Y}$ denote two discrete random variables defined respectively on $\mathcal{X}$ and $\mathcal{Y}$, with probability mass distributions $\tilde{p}(X)$ and $\tilde{p}(Y)$ and joint probability mass distribution $\tilde{p}(X,Y)$ defined on $\mathcal{X} \times \mathcal{Y}$. Note that $p(y) = P(Y = y)$ and $\tilde{p}(y) = P(\tilde{Y} = y)$.

2.1.1 Kullback-Leibler divergence

The Kullback-Leibler (KL) divergence [26] is a non-commutative measure of the difference between two discrete probability distributions.

Definition 2.1: *The* Kullback-Leibler divergence *between* $p(Y)$ *and* $\tilde{p}(Y)$ *is*

$$KL(p(Y);\tilde{p}(Y)) = \sum_{y\in\mathcal{Y}} p(y)\log\frac{p(y)}{\tilde{p}(y)} \tag{2.1}$$

$$= E_Y\left[\log\frac{p(y)}{\tilde{p}(y)}\right] = E_Y[\log p(y) - \log\tilde{p}(y)] \tag{2.2}$$

In the above definition, we use the convention (from [26]) that $0\log\frac{0}{0} = 0$ and the convention (based on continuity arguments) that $0\log\frac{0}{\tilde{p}} = 0$ and $p\log\frac{p}{0} = \infty$.

The definition of the Kullback-Leibler divergence can be easily extended to a pair of random variables since the latter can be considered as single random vector.

Definition 2.2: *[26] The* joint Kullback-Leibler divergence *between two probability mass functions* $p(X,Y)$ *and* $\tilde{p}(X,Y)$ *is*

$$KL(p(X,Y);\tilde{p}(X,Y)) = \sum_{x\in\mathcal{X}} p(x)\sum_{y\in\mathcal{Y}} p(y|x)\log\frac{p(x,y)}{\tilde{p}(x,y)} \tag{2.3}$$

Definition 2.3: *[26] The* conditional Kullback-Leibler divergence *between* $p(Y|X)$ *and* $\tilde{p}(Y|X)$ *is defined as*

$$KL(p(Y|X);\tilde{p}(Y|X)) = \sum_{x\in\mathcal{X}} p(x)\sum_{y\in\mathcal{Y}} p(y|x)\log\frac{p(y|x)}{\tilde{p}(y|x)} \tag{2.4}$$

This means that a conditional (discrete) Kullback-Leibler divergence can also be defined as the expected value of the KL-divergence of the conditional probability mass functions, averaged over the conditioning random variables.

Theorem 2.1: [Non negativity of Kullback-Leibler divergence] *[26] Given two distributions* p *and* $\tilde{p}$*, it can be shown that*

$$KL(p;\tilde{p}) \geq 0 \tag{2.5}$$

with equality $KL(p;\tilde{p}) = 0$ *iff* $p = \tilde{p}$ *(where* p *can denote* $p(Y)$*,* $p(X)$*,* $p(X,Y)$ *or* $p(Y|X)$ *and equivalently for* $\tilde{p}$*).*

The link between KL divergence, joint KL divergence and conditional KL divergence is provided by a chain rule:

Theorem 2.2: [Chain rule for Kullback-Leibler divergence]*[26]*

$$KL(p(X,Y);\tilde{p}(X,Y)) = KL(p(Y);\tilde{p}(Y)) + KL(p(X|Y);\tilde{p}(X|Y)) \tag{2.6}$$

2.1.2 Entropy

Let us decompose the KL divergence into two terms:

$$\mathrm{KL}(p(Y);\tilde{p}(Y)) = \sum_{y\in\mathcal{Y}} p(y)\log p(y) - \sum_{y\in\mathcal{Y}} p(y)\log\tilde{p}(y) \tag{2.7}$$

The first term of the decomposition is the opposite of a particular information-theoretic quantity called the entropy of Y.

Definition 2.4: *[26] The* entropy *of a discrete random variable* Y *with probability mass function* $p(Y)$ *is defined by :*

$$H(Y) = H(p(Y)) = -\sum_{y\in\mathcal{Y}} p(y)\log p(y) = E_Y[\log\frac{1}{p(y)}] \tag{2.8}$$

The usual unit of the entropy is the *bit*. However, other units are sometimes chosen for this measure. The unit depends on the base taken for the logarithm of Eq 2.8, base 2 for bit, base 10 for *ban*. The *deciban* (one tenth of a ban) is also known as a useful measure of belief since 10 decibans correspond to an odds ratio of 10:1; 20 decibans to 100:1 odds, 30 decibans to 1000:1, etc [69]. The natural logarithm (base e) is increasingly used for computational reasons and in this case the unit is the *nat*.

Note that the value of the entropy depends on the distribution of $p(Y)$ and is maximum for a uniform distribution $u(Y)$, [26] $H(u(Y)) = \log|\mathcal{Y}|$.

Note that by replacing $\tilde{p}(Y)$ by the uniform distribution $u(Y)$ in (2.7), we obtain

$$H(p(Y)) = \log|\mathcal{Y}| - \mathrm{KL}(p(Y);u(Y)) \tag{2.9}$$

This equation expresses the entropy of a random variable Y as the logarithm of the size of the support set minus the KL divergence between the probability distribution of Y and the uniform distribution on the same domain $\mathcal{Y}$. The closer the probability distribution is to a uniform distribution, the higher is the entropy. In other words, entropy measures the

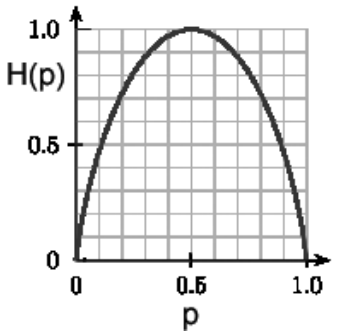


Figure 2.1: $H(p)$ as a function of p (Bernoulli distribution)

"randomness" of Y. Figure 2.1 shows the graph of $H(p)$ as a function of the probability distribution p of a binary variable (Bernoulli distribution). The entropy is a concave function that equals 0 when $p = 1$ or $p = 0$, i.e. when the variable is deterministic. The entropy is maximal when $p = \frac{1}{2}$, which corresponds to the largest level of uncertainty [26]. Other interpretations of the entropy are available in Appendix B.2.

2.1.3 Multivariate entropy and conditional entropy

Let us consider a pair of discrete random variables (Y, X) with a joint distribution $p(Y, X)$,

Definition 2.5: *[26] The* joint entropy $H(Y, X)$ *is,*

$$H(Y, X) = -\sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} p(y, x) \log p(y, x) \tag{2.10}$$

Note that the maximal joint entropy is reached with independent variables (i.e., $p(Y, X) = p(Y)p(X)$). In this case, the entropy of the joint probability distribution is equal to the sum of their respective entropies. An upper bound on $H(Y, X)$ is provided by the following theorem,

Theorem 2.3: [Independence bound on entropy]*[26]*

$$H(Y, X) \leq H(Y) + H(X) \tag{2.11}$$

with equality iff X and Y are independent.

Given a conditional probability distribution $p(Y|X)$, a *conditional entropy* can be defined,

Definition 2.6: *[26] The* conditional entropy *of Y given X is,*

$$H(Y|X) = -\sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}} p(x,y)\log p(y|x) \tag{2.12}$$

This quantity measures the uncertainty of a variable once another one is known.

As for the KL divergence, the relation between entropy, conditional entropy and joint entropy is characterized by a chain rule[1].

Theorem 2.4: [Chain rule for entropy]*[26]*

Let $X_1, X_2, ..., X_n$ be drawn according to $p(X_1, X_2, ..., X_n)$. Then,

$$H(X_1, X_2, ..., X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}, ..., X_1) \tag{2.13}$$

From the independence bound (Theorem 2.3) and the chain rule for entropy (2.4), it follows

$$H(Y, X) = H(X) + H(Y|X) \leq H(X) + H(Y) \tag{2.14}$$

This is made explicit by the following theorem which shows that "conditioning reduces entropy".

Theorem 2.5: [Conditioning reduces entropy]*[26]*

$$H(Y|X) \leq H(Y) \tag{2.15}$$

with equality iff X and Y are independent.

In a prediction perspective where Y is a target variable and X is a predictor, this means that adding variables can only decrease the uncertainty on the target Y.

2.1.4 Mutual Information

The reduction of entropy due to conditioning can be quantified by a symmetric measure called *mutual information* [26]:

$$H(Y) - H(Y|X) = I(Y;X) = I(X;Y) = H(X) - H(X|Y) \tag{2.16}$$

[1] A word of caution about the notation is worthy here. In information-theoretic notations, the symbols ';' and '|' are used to separate random vectors whereas the symbol ',' is used to separate random components of a random vector.

Definition 2.7: *[26] The* mutual information *between X and Y is,*

$$I(Y;X) = \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log(\frac{p(x,y)}{p(x)p(y)}) \tag{2.17}$$

Mutual information can be also expressed as the Kullback-Leibler divergence between the joint distribution $p(X,Y)$ and the product distribution $p(X)p(Y)$ [26].

$$I(X;Y) = \mathrm{KL}(p(X,Y);p(X)p(Y)) \tag{2.18}$$

or between the marginal distribution $p(X)$ and a conditional distribution $p(X|Y)$

$$I(Y;X) = \mathrm{KL}(p(X|Y);p(X)) = \mathrm{KL}(p(Y|X);p(Y)) \tag{2.19}$$

As a consequence, when two variables are independent, their mutual information is null. In other words, the higher the dependency between two variables, the higher the value of the mutual information. When the two variables are identical, this measure reaches its maximum and is equal to the entropy of the variable, i.e., $I(X;X) = H(X)$.

It follows from the non-negativity of the KL-divergence that mutual information, in a discrete case, is also non-negative.

Theorem 2.6: *[26] Non-negativity of mutual information*

$$I(Y;X) \geq 0 \tag{2.20}$$

with equality if X and Y are independent.

The Kullback-Leibler divergence and the mutual information are often used as measures of dissimilarity and similarity, respectively, but do not fulfill the triangle inequality, hence they are not distances in the mathematical sense [26].

2.2 Information Measures over Multiple Sets of Random Variables

So far, we have considered measures having **at most** two random vectors as arguments. In this section, measures having **at least** two random vectors are introduced. These measures play a major role in this thesis since they will be used in the chapter 3 in order to define notions like relevance, redundancy and complementarity of variables.

2.2.1 Conditional Mutual Information

Definition 2.8: *[26] The* conditional mutual information *of the random variable X and Y given Z is,*

$$\begin{aligned} I(X;Y|Z) &= H(Y|Z) - H(Y|X,Z) \\ &= H(X|Z) - H(X|Z,Y) = I(Y;X|Z) \end{aligned} \tag{2.21}$$

This quantity measures the reduction of uncertainty on Y (or X) due to the other variable X (or Y), when Z is given.

Definition 2.9: *[95]Given random variables X, Y and Z, X is* conditionally independent *of Y given Z, if and only if the probability distribution of X is independent of the value of Y given Z that is*

$$p(X|Y,Z) = p(X|Z) \tag{2.22}$$

Conditional mutual information can be shown to be non-negative since conditioning reduces entropy.

Theorem 2.7: [Non-negativity of conditional mutual information]*[26]*

$$I(X;Y|Z) \geq 0 \tag{2.23}$$

with equality iff X and Y are conditionally independent given Z.

In terms of KL divergence, the conditional mutual information can be written as [140]:

$$\begin{aligned} I(X;Y|Z) &= \mathrm{KL}(p(X,Y,Z); p(X|Z)p(Y|Z)p(Z)) \\ &= \mathrm{KL}(p(X|Y,Z)p(Y|Z)p(Z); p(X|Z)p(Y|Z)p(Z)) \\ &= \mathrm{KL}(p(X|Y,Z); p(X|Z)) \end{aligned} \tag{2.24}$$

As well as for entropy, a chain rule formulation holds.

Theorem 2.8: [Chain rule for information]*[26] Let $X_1, X_2, ..., X_n$ be drawn according to $p(X_1, X_2, ..., X_n)$. Then,*

$$I(X_1, X_2, ..., X_n; Y) = \sum_{i=1}^{n} I(X_i; Y|X_{i-1}, ..., X_1) \tag{2.25}$$

This chain rule leads to the following inequality.

Theorem 2.9: [Data processing inequality]*[26] if G and Q are conditionally independent given D, then $I(G;Q) \leq min(I(G;D), I(D;Q))$.*

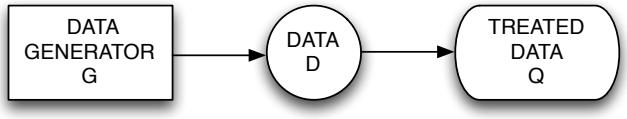


Figure 2.2: Data processing inequality representation

PROOF: By the chain rule

$$I(G;D,Q) = I(G;Q) + I(G;D|Q) = I(G;D) + I(G;Q|D) \tag{2.26}$$

Since G and Q are conditionally independent given D, we have $I(G;Q|D) = 0$. Since $I(G;D|Q) \geq 0$, we have

$$I(G;Q) \leq I(G;D) \tag{2.27}$$

The same reasoning applied to $I(Q;D,G)$ yields

$$I(G;Q) \leq I(Q;D) \tag{2.28}$$

hence $I(G;Q) \leq \min(I(G;D), I(D;Q))$ ■

This theorem is called the *data processing inequality* because if D denotes the data and Q is a function of the data (alone), then the data generating process G is conditionally independent of Q given D (see Figure 2.2). This means that no (automatic) clever manipulation Q of the data D can increase the information about the generating process G (on average).

Another important property of conditional mutual information is that it is context dependent, i.e. , it depends on the conditioning variable. We present here two illustrative examples where the context (i.e. the conditioning) variable influences the information existing between two variables.

Example 2.1: *Let X, Y, Z and W be four discrete random variables and Y be a deterministic function of (X, Z) and W be a deterministic function of X.*

Since $Y = f(X, Z)$, we have $I(X;Y|Z) > 0$, $I(X;Y) > 0$, $I(Z;Y) > 0$ and as $W = X$, then $I(X;Y|Z,W) = 0$ (see Fig. 2.3). Therefore, conditioning can reduce the information.

At the same time, it is possible to increase the information of a variable by appropriate conditioning, as shown in the following example.

Example 2.2: *Let Y and X be independent random variables and Z be a deterministic function of Y and X (see Figure 2.4).*

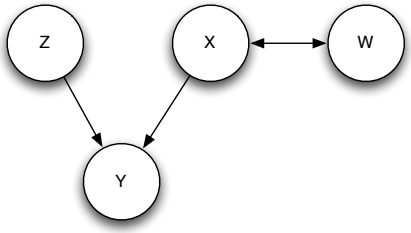


Figure 2.3: Graphical representation of the relationships between the variables of example 2.1

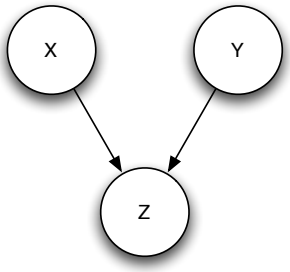


Figure 2.4: Graphical representation of the relationships between the variables of Example 2.2

As X and Y are independent, we have $I(X;Y) = 0$ and since $Z = f(X,Y)$, we obtain $I(X;Y|Z) > 0$. As a result, the conditional mutual information is higher than the mutual information, i.e. $I(X;Y|Z) > I(X;Y)$, which means that conditioning can also increase information.

2.2.2 Interaction Information

Definition 2.10: *The interaction information among n sets of random variables,* $X_1, X_2, ..., X_n$ *is defined as:*

$$C(X_1; X_2; ...; X_n) = \sum_{k=1}^{n} \sum_{S \subseteq \{1,2...,n\}:|S|=k} (-1)^{k+1} H(X_S) \tag{2.29}$$

The interaction information comes from the seminal paper [84] and from [27].

For $n = 2$, we have

$$C(X;Y) = H(X) + H(Y) - H(X,Y) = I(X;Y)$$

X_1	X_2	$Y = X_1 \oplus X_2$
1	1	0
1	0	1
0	1	1
0	0	0

Table 2.1: Truth table of the XOR problem

while for $n = 3$:

$$C(Y;X;Z) = H(Y)+H(X)+H(Z)-H(Y,X)-H(Y,Z)-H(X,Z)+H(Y,X,Z) \quad (2.30)$$

$$C(Y;X;Z) = I(X;Y) + I(Z;Y) - I(X,Z;Y) \quad (2.31)$$

Note that the interaction information is order independent.

$$\begin{aligned} C(Y;X;Z) &= C(X;Y;Z) = C(X;Z;Y) \\ &= C(Y;Z;X) = C(Z;X;Y) = C(Z;Y;X) \\ &= I(X;Y) - I(X;Y|Z) = I(X;Z) - I(X;Z|Y) = I(Y;Z) - I(Y;Z|X) \end{aligned} \quad (2.32)$$

When the measure of interaction is strictly positive, it indicates that the n sets of variables share a common information. Interaction can also be negative as in Example 2.2. In this case, it means that the interacting variables are more informative together than independently. We shall go back to negative interaction in chapter 3 in order to define variable complementarity. Another known example of negative interaction is given by the XOR problem.

Example 2.3: *[XOR problem][74] Let us consider three binary random variables X_1, X_2, Y where Y is a deterministic function of X_1 and X_2. More precisely, the variable Y is obtained as an* exclusive disjunction *of X_1 and X_2, see Table 2.1.*

In this example, X_1 and X_2 have a null mutual information with Y, separately, i.e., $I(X_1;Y) = 0$, $I(X_2;Y) = 0$, whereas the couple (X_1, X_2) has maximal mutual information with Y, i.e. $I(X_1, X_2; Y) = H(Y) > 0$. Hence,

$$\underbrace{I(X_1;Y)}_{0} + \underbrace{I(X_2;Y)}_{0} - I(X_1, X_2; Y) = C(X_1; X_2; Y) < 0$$

The link between mutual information and interaction information is provided by the following theorem.

Theorem 2.10: [mutual and interaction information relationship]*[75] Mutual information between Y and n random variables $X_i,\ i \in A = \{1, 2, ..., n\}$, can be written as,*

$$I(X;Y) = \sum_{i \in A} I(X_i;Y) - \sum_{i,(j>i) \in A} C(X_i;X_j;Y) + \sum_{i,(j>i),(k>j) \in A} ... + (-1)^{n+1} C(X_1;X_2;...;X_n;Y) \tag{2.33}$$

Mutual information can be seen as a series where the higher order terms are corrective terms that represent the effect of the simultaneous interaction of several variables.

2.2.3 Multiinformation

Multiinformation, also called *total correlation* [67], will be used in Chapter 3, as a measure of redundancy between variables. It comes from the seminal paper [84] but may also be found in [129].

Definition 2.11: *Multiinformation among n sets of random variables, $X_1, X_2, ..., X_n$ is defined as:*

$$R(X_1;X_2;...;X_n) = \sum_{i=1}^{n} H(X_i) - H(X_1, X_2, ..., X_n) \tag{2.34}$$

This measure is symmetric, non-negative and non-decreasing with the number of variables.

Multiinformation can also be written as the KL divergence between the joint distribution of n input variables and their product distribution,

$$\begin{aligned} R(X;Y) &= \mathrm{KL}(p(X_1, X_2, ..., X_n); p(X_1), p(X_2), ..., p(X_n)) \\ &= E\left[\log \tfrac{p(X_1,X_2,...,X_n)}{p(X_1)p(X_2)...p(X_n)}\right] \end{aligned} \tag{2.35}$$

Hence, for $n = 2$, multiinformation boils down to the mutual information,

$$R(X;Y) = H(X) + H(Y) - H(X,Y) = I(X;Y) \tag{2.36}$$

As shown by the following theorem, multiinformation can be decomposed in a sum of interaction information.

Theorem 2.11: [multiinformation decomposition]*[27]*

$$R(X_1;X_2;...;X_n) = \sum_{i,(j>i) \in A}^{n} C(X_i;X_j) - \sum_{i,(j>i)(k>j) \in A}^{n} C(X_i;X_j;X_k) + ... + (-1)^n C(X_1;X_2;...;X_n) \tag{2.37}$$

2.3 Normalized Measure of Information

Mutual information measures (usually in bits) the reduction of uncertainty on a variable Y due to the variable X (or the other way round since it is symmetric). However, it is sometimes useful to know which portion of the uncertainty of Y is "explained" by X. For example, let the unknown entropy of the probability distribution of the random variable Y be 6 bits. Sometimes it can be sufficient to know that X reduces the uncertainty of Y by 50% instead of knowing that the mutual information between X and Y amounts to 3 bits. As a result, normalized mutual information measures have been proposed in the literature.

Definition 2.12: *[116] The asymmetric uncertainty coefficients are given by:*

$$AU_Y(X;Y) = \frac{I(X;Y)}{H(Y)} \tag{2.38}$$

$$AU_X(X;Y) = \frac{I(X;Y)}{H(X)} \tag{2.39}$$

In a predictive setting where X is an input variable and Y is the output one, the coefficient AU_Y , comprised between 0 and 1, indicates the amount of uncertainty of Y that is "explained" by X. AU_X can be interpreted as the information of the input variable X compared to its entropy. In other words, the lower the entropy $H(X)$ of the input (with the same amount of information) the higher the value returned by the coefficient.

There also exists symmetric normalized measures of mutual information: the symmetric uncertainty and the symmetric relevance,

Definition 2.13: *[116] The Symmetric Uncertainty*

$$SU(X;Y) = \frac{2I(X;Y)}{H(X) + H(Y)} \tag{2.40}$$

[147] The Symmetric Relevance

$$SR(X;Y) = \frac{I(X;Y)}{H(X,Y)} \tag{2.41}$$

Conditional mutual information can be normalized in the same way. However, in the asymmetric case, normalization can be made w.r.t. $H(Y)$ or $H(Y|Z)$.

Definition 2.14: *Conditional asymmetric uncertainty coefficients*

$$CAU_Y(X;Y|Z) = \frac{I(X;Y|Z)}{H(Y)} \tag{2.42}$$

$$CAU_{Y|Z}(X;Y|Z) = \frac{I(X;Y|Z)}{H(Y|Z)} \tag{2.43}$$

Both coefficients take values between 0 and 1. CAU_Y yields the amount of Y explained by $X|Z$ and $CAU_{Y|Z}$ measures the reduction of uncertainty of $Y|Z$ given by variable X.

We will see in the next chapter how these normalized measures may be used by variable selection algorithms in order to improve the selection.

2.4 Introduction to Pattern Recognition

Let us consider a prediction problem where we want to study the relationship between a set of inputs X and an output Y. A model is adequate if it returns "good" predictions for the output Y on new input data. *"The capability of a model to realize good predictions on independent test data is called the generalization performance"* [60]. Actually, an infinity of models can be constructed from some data points though only few of them will benefit from good generalization performances.

In the following sections, we will sketch the basics of supervised learning in a classification task. The presentation relies on some elementary notions of estimation theory which are given in Appendix B.1.

2.4.1 Supervised Learning

Let $Y \in \mathcal{Y}$ denote a discrete valued random output variable and let $X \in \mathcal{X}$ denote a discrete valued random vector of inputs. Let $p(Y|X)$ be the target probability distribution which maps the input values x onto an output value y. The aim of pattern recognition is to create a classifier $g(x) : \mathcal{X} \rightarrow \mathcal{Y}$ which reliably guesses y once given x. An error occurs at input point x if $g(x) \neq y$.

Definition 2.15: *[35]The Bayes error $L^*(X)$ is defined as,*

$$L^*(X) = P(g^*(X) \neq Y) \tag{2.44}$$

It can be shown that the optimal classifier, w.r.t. Bayes error, is given by the Bayes decision function.

Definition 2.16: *[35]The Bayes decision function $g^*(x)$ is defined as*

$$g^*(x) = \arg\max_{y \in \mathcal{Y}} p(y|x) \tag{2.45}$$

In a supervised learning problem, the target distribution $p(Y|X)$ and the input distribution $p(X)$ are fixed but unknown probability distributions. However, a dataset D_m made of m independent and identically distributed (i.i.d.) samples $(y_r, x_r), r \in \{1, 2, ..., m\}$ drawn according to the joint distribution $p(Y, X) = p(Y|X)p(X)$ is supposed to be available.

A learning procedure is conventionally made of two steps: parametric identification and structural identification.

2.4.1.1 Parametric identification

In a parametric approach, one assumes that the conditional probability distribution is of a known form (i.e., it belongs to the class of model Λ) but has an unknown vector of parameters θ, i.e., $p(Y|\theta, X)$ [139]. Hence, the parametric identification problem consists in identifying the probability distribution $\hat{p}(Y \mid \theta_m, X)$ or the function $g(x, \theta_m) = \arg\max_{y \in \mathcal{Y}} \hat{p}(y \mid \theta_m, x)$ which "best" explains the data (with θ_m the estimated parameters). The function $g(x, \theta_m)$ or the associated probability distribution $\hat{p}(Y|\theta_m, X)$ is called a *classifier* or a *model*.

Once the input variables X and the class of model Λ are fixed, a strategy to build a model $g(x, \theta_m)$ that fits the data D_m, consists in minimizing an empirical risk function $R_m(\theta)$ [60],

$$\theta_m = \theta(D_m) = \arg\min_{\theta \in \Theta} R_m(\theta) \tag{2.46}$$

where Θ is the parameter set and the empirical risk is usually of the form,

$$R_m(\theta) = \frac{1}{m} \sum_{r=1}^{m} L(y_r, g(x_r, \theta)) \tag{2.47}$$

where $L(y, g(x, \theta))$ is a loss function between the target value y and the model at an input point x. This can be seen as a non-linear optimization problem in a multidimensional space.

A convenient loss function in a classification problem is the 0-1 loss [39]:

$$L(y_r, g(x_r, \theta)) = \begin{cases} 0 \text{ if } y_r = g(x_r, \theta) \\ 1 \text{ if } y_r \neq g(x_r, \theta) \end{cases} \tag{2.48}$$

Another empirical risk function is given by the (empirical) negative log-likelihood function $\mathrm{LL}_m(\theta)$ (also called cross-entropy)

$$\mathrm{LL}_m(\theta) = \mathrm{LL}_m(\hat{p}) = -\sum_{r=1}^{m} \log(\hat{p}(y_r|\theta, x_r)) \tag{2.49}$$

The Maximum Likelihood Estimators (MLE) are the values of the parameters of a model that maximize the probability of obtaining a dataset. If the samples in a dataset D_m are drawn independently and identically distributed (i.i.d.) from the distribution $p(X, Y)$, then the probability $p(D_m)$ of observing the dataset D_m is given by,

$$p(D_m) = \prod_{r=1}^{m} p(y_r, x_r) = \prod_{r=1}^{m} p(y_r|x_r)p(x_r) \tag{2.50}$$

Hence,

$$\arg\max_{\theta\in\Theta} \hat{p}(D_m|\theta) = \arg\max_{\theta\in\Theta} \prod_{r=1}^{m} \hat{p}(y_r|\theta, x_r)p(x_r) = \arg\max_{\theta\in\Theta} \prod_{r=1}^{m} \hat{p}(y_r|\theta, x_r) = \theta^{MLE} \tag{2.51}$$

The values of the parameters θ that minimizes the (empirical) negative log-likelihood $\mathrm{LL}_m(\hat{p})$ are precisely the maximum likelihood estimators (MLE), since

$$\arg\min_{\theta\in\Theta} -\sum_{r=1}^{m} \log \hat{p}(y_r|\theta, x_r) = \arg\max_{\theta\in\Theta} \prod_{r=1}^{m} \hat{p}(y_r|\theta, x_r) \tag{2.52}$$

The minimum of the negative log-likelihood $\mathrm{LL}(\hat{p})$ also corresponds to the minimum of the empirical Kullback-Leibler divergence $\mathrm{KL}(p;\hat{p})$ [17]. Indeed, the Kullback-Leibler divergence between $p(Y|X)$ and $\hat{p}(Y|\theta, X)$ is given by (2.4)

$$\mathrm{KL}(p;\ \hat{p}) = E_{X,Y}\left[\log \frac{p(y|x)}{\hat{p}(y|\theta, x)}\right] \tag{2.53}$$

This can be decomposed into two terms (2.7) [115],

$$\mathrm{KL}(p;\ \hat{p}) = E_{X,Y}[\log p(y|x)] - E_{X,Y}[\log \hat{p}(y|\theta, x)] \tag{2.54}$$

where the first term is the opposite of the conditional entropy,

$$H(Y|X) = E_{X,Y}\left[\log \frac{1}{p(y|x)}\right] = H(p) \tag{2.55}$$

and the second term of (2.54) is $LL(\hat{p})$, the negative log-likelihood of $\hat{p}$. (2.54) can be written as

$$\mathrm{KL}(p;\ \hat{p}) = \mathrm{LL}(\hat{p}) - H(p) \tag{2.56}$$

where the entropy term is independent from the model $\hat{p}$ and, as a result, the argument of the minimum of $\mathrm{LL}(\hat{p})$ corresponds to the argument of the minimum of $\mathrm{KL}(p;\hat{p})$.

2.4.1.2 Structural identification

The second step of a machine learning procedure aims to identify the best family of models able to represent the stochastic dependency between inputs and output.

A typical problem of the parametric identification is that it does not take into account the sampling variability. In other words, a different dataset D_m would give a different estimate θ_m of the parameters [139].

In the "non-parametric" and/or "semi-parametric" approaches that are adopted in this work, the objective is to minimize $E_{D_m}[\mathrm{LL}_m(\hat{p})]$ rather than $\mathrm{LL}_m(\hat{p})$ in order to take into account sampling variability.

By choosing a family of models, assumptions are made on the unknown distribution of the target (e.g. neural network model or polynomial model). The more restrictive the assumptions are, the smaller the class Λ of models is. To reduce the class of models one can limit the number of parameters (for example, number of hidden neurons or linear model instead of a second order polynomial). One can also reduce the number of potential inputs X in order to reduce the number of parameters. Indeed, in a linear model there is one parameter per input variable whereas most non-linear models use more (than one) parameters per input variables (for example, neural networks). As a consequence, the smaller the number of inputs, the smaller Λ is.

One can think that the best class of models is the largest one; however, this is not the case because the empirical negative log-likelihood estimate $\mathrm{LL}_m(\hat{p})$ has a tendency to underestimate $E_{D_m}[\mathrm{LL}_m(\hat{p})]$ for models with a large number of parameters. In other words, the performances of a model $\hat{p}$ are over-optimistic when using $\mathrm{LL}_m(\hat{p})$ with a high number of parameters.

Indeed, the expected value of the negative log-likelihood, over the ensemble of the training sets with m samples, can be decomposed as in (2.56) (see also [57]),

$$E_{D_m}[\mathrm{LL}_m(\hat{p})] = E_{D_m}[\mathrm{KL}_m(p;\hat{p})] + H(p) \tag{2.57}$$

where (2.54),

$$E_{D_m}[\mathrm{KL}_m(p;\hat{p})] = E_{D_m}\Big[E_{X,Y}[\log p(y|x)] - E_{X,Y}[\log \hat{p}(y|\theta_m, x)]\Big] \tag{2.58}$$

Adding and subtracting $E_{X,Y}\Big[\log E_{D_m}[\hat{p}(y|\theta_m, x)]\Big]$ we obtain

$$\begin{aligned} &= E_{D_m}\Big[E_{X,Y}[\log p(y|x)]\Big] - E_{D_m}\Big[E_{X,Y}[\log \hat{p}(y|\theta_m, x)]\Big] \\ &- E_{X,Y}\Big[\log E_{D_m}[\hat{p}(y|\theta_m, x)]\Big] + E_{X,Y}\Big[\log E_{D_m}[\hat{p}(y|\theta_m, x)]\Big] \end{aligned} \tag{2.59}$$

and reorganizing the terms,

$$\begin{aligned} &= E_{X,Y}\Big[\log p(y|x) - \log E_{D_m}[\hat{p}(y|\theta_m, x)]\Big] \\ &+E_{X,Y}\Big[\log E_{D_m}[\hat{p}(y|\theta_m, x)] - E_{D_m}[\log \hat{p}(y|\theta_m, x)]\Big] \end{aligned} \tag{2.60}$$

$$\Rightarrow E_{D_m}[\mathrm{KL}_m(p;\hat{p})] = \mathrm{KL}(p; E_{D_m}[\hat{p}]) + E_{D_m,X,Y}\Big[\log \frac{E_{D_m}[\hat{p}(y|\theta_m, x)]}{\hat{p}(y|\theta_m, x)}\Big] \tag{2.61}$$

From (2.57) and (2.61), we have

$$\Rightarrow \quad \begin{aligned} E_{D_m}[\mathrm{LL}_m(\hat{p})] &= \overbrace{H(p)}^{noise} + \overbrace{\mathrm{KL}(p; E_{D_m}[\hat{p}])}^{bias} \\ &+ \underbrace{E_{D_m,X,Y}\Big[\log \frac{E_{D_m}[\hat{p}(y|\theta_m, x)]}{\hat{p}(y|\theta_m, x)}\Big]}_{variance} \end{aligned} \tag{2.62}$$

This formulation is known as the log-loss bias-variance trade-off [142]. The terms are named bias and variance following the well-known squared error loss decomposition [60], where the bias term $\mathrm{KL}(p; E_{D_m}[\hat{p}])$ measures how far (using the Kullback-Leibler divergence) the average of the estimates is from the parameters being estimated, and where the variance term $E_{D_m,X,Y}\Big[\log \frac{E_{D_m}[\hat{p}(y|\theta_m,x)]}{\hat{p}(y|\theta_m,x)}\Big]$ measures an average distance between the estimates and the expected value of the estimates. The term $H(p)$ represents the intrinsic noise of the target distribution. Another decomposition of the bias-variance trade-off is available in Appendix B.3.

Qualitatively, the higher the number of parameters of a model, the easier it becomes to fit the training set. Hence, with enough parameters, it becomes possible to fit the noise in the dataset (overfitting). On the one hand, decreasing the number of parameters of a model tends to increase its bias. On the other hand, the variance of the prediction increases with the number of parameters (see figure 2.5). This problem is illustrated by two extreme modelling situations, known as *underfitting* and *overfitting* (see figure 2.6).

Consider a classification problem where the optimal decision function is represented by the non-linear dotted line in figure 2.6:

- Underfitting (Figure 2.6 (a)) happens with a too "simple" linear model having many errors on the data points.
- Overfitting (Figure 2.6 (b)) happens with a too "complex" model which has no error on the training set but that generalizes poorly.

Hence, a model with either a large bias or variance will typically generalize poorly.

Another way of presenting the results of a classifier is given by the confusion matrix CM. The confusion matrix of a model g is the $|\mathcal{Y}| \times |\mathcal{Y}|$ matrix whose elements cm_{ij} are

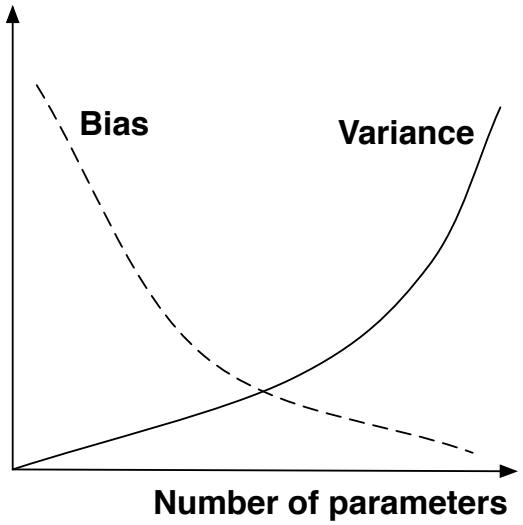


Figure 2.5: Bias-variance dilemma

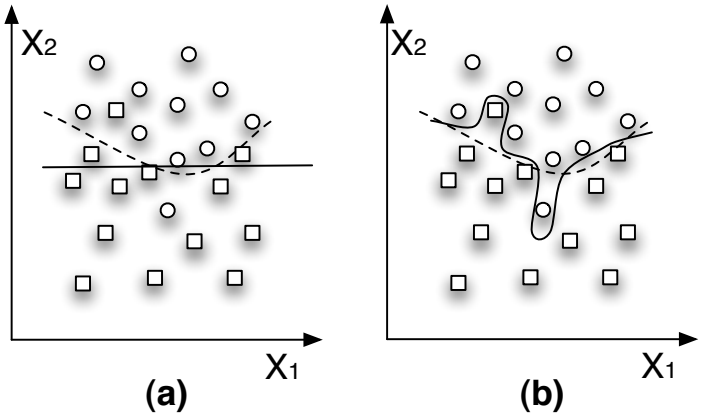


Figure 2.6: The output Y can take two values: square or bullet, the continuous line is the model, the dashed line is the target function. In (a) a linear model that underfits the target function. In (b) a complex non-linear model that overfits the target function (fits the noise).

Model\Truth	1	0
1	True Positive (tp)	False Positive (fp)
0	False Negative (fn)	True Negative (tn)

Table 2.2: Confusion matrix CM for a binary classification problem

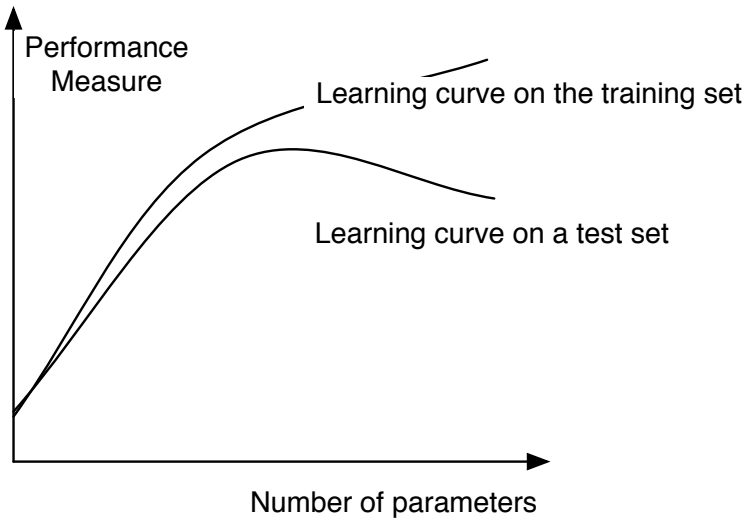


Figure 2.7: Learning curves on training set and test set

the number of elements of category $y_i \in \mathcal{Y}$ that have been put in the category $y_j \in \mathcal{Y}$ by g (see Tab. 2.2). The sum of non-diagonal terms of the confusion matrix CM divided by m is equal to the empirical risk of the 0-1 loss.

As for the negative log-likelihood, the empirical risk using a 0-1 loss has been shown to be a biased estimate of the expected value [39]. Hence, the use of the empirical log-likelihood or empirical risk to avoid overfitting is not recommended.

Several analytical criteria have been proposed to estimate the risk function. However, these estimations are often based on strong assumptions (e.g. large sample datasets [115]).

Validation strategies are widely used because of less restrictive assumptions. The simplest form of validation consists in splitting the dataset into two parts: the *training set* and the *validation set*.

In practice, it is possible to increase the performances of a learning algorithm on the learning set just by increasing the number of parameters. However, the performances on an independent test set start decreasing when variance becomes too large compared to the bias, see Figure 2.7 [25].

In the validation strategy, the training set is used to adjust the parameters of several

models (parametric identification), then the validation set is used to choose the model with the best generalization performances (structural identification). The most popular method is the *K-fold cross-validation*. The principle is the following: the dataset is splitted into K parts. First, the test set is the Kth part of the dataset and the training set is made of all the other $(K-1)$ parts. The parameters are identified with the training set (θ_{-K}). That operation is repeated K times, the test set being each time another of the K different parts. The cross-validation estimator of the error is the combination of the various validation errors:

$$R_m^{cv}(\theta) = \frac{1}{m} \sum_{r=1}^{m} L(y_r, g(\theta_{-K(r)}, x_r)) \tag{2.63}$$

where $K(r)$ denotes the part of the dataset containing sample r.

The particular case $K = m$ is called *leave-one-out cross-validation (loo)* (see [60]). The validation set is then made of one sample and the training set, of all the other $m-1$ samples. The loo cross-validation estimator of the 0-1 loss, R_m^{loo}, is known to be an unbiased estimator of the 0-1 loss, R. However, the loo cross-validation procedure can be computationally expensive for non-linear models [60]. The use of cross-validation criteria allows the selection of the structure expected to generalize the best.

In practice, we consider a nested sequence of classes of hypotheses $\Lambda_1 \subset \Lambda_2 \subset ... \subset \Lambda_s \subset ...$ and the objective of the structural identification is to identify the class Λ_s leading to the best performances.

$$s^{cv} = \arg\min_s R_m^{min} \tag{2.64}$$

with

$$R_m^{min} = \min_\theta R_m^{cv}(\theta) \tag{2.65}$$

After completing the structural identification, the parametrical identification is performed on the complete dataset.

2.5 Machine Learning Algorithms

We briefly review here the principles of three popular classifiers, namely the naive Bayes, the k-nearest-neighbour and the support vector machine. Additional details about these algorithms can be found in [62, 25, 60, 92, 139].

2.5.1 The Naive Bayes Classifier

This algorithm is a simple way to implement the classifier $g(x) = \arg\max_{y\in\mathcal{Y}} \hat{p}(y|x)$, when x is multivariate.

From the Bayes theorem, we have

$$g(x) = \arg\max_{y\in\mathcal{Y}} \frac{\hat{p}(x|y)\hat{p}(y)}{\hat{p}(x)} \tag{2.66}$$

Since the denominator does not depend on y, it follows that

$$g(x) = \arg\max_{y\in\mathcal{Y}} \hat{p}(x|y)\hat{p}(y) \tag{2.67}$$

The naive Bayes classifier assumes conditional independence in order to reduce the number of parameters.

If the n input variables $X_1, X_2, ..., X_n$ are conditionally independent given the output Y, (2.67) becomes

$$g(x) = \arg\max_{y\in\mathcal{Y}} \hat{p}(y) \prod_{i=1}^{n} \hat{p}(x_i|y) \tag{2.68}$$

The probability distribution estimated by the naive Bayes approach is given by [93].

$$\hat{p}(y|x_1, x_2, ..., x_n) = \frac{\hat{p}(y)\prod_{i=1}^{n}\hat{p}(x_i|y)}{\sum_{y\in\mathcal{Y}}\hat{p}(y)\prod_{i=1}^{n}\hat{p}(x_i|y)} \tag{2.69}$$

The independence assumption is expected to increase the bias of the classifier at the cost of a reduced variance. In practice, the naive Bayes model is surprisingly accurate in front of few samples [40].

2.5.2 K-Nearest-Neighbors (KNN)

In KNN, the classification of the output variable Y given input values x is obtained by a majority vote of the nearest neighbors of x. The point x being predicted is assigned to the class most common amongst its k nearest neighbors. More formally, given a binary output, the KNN rule [35] is given by,

$$g(x) = \begin{cases} 1 & \text{if} \quad \sum_{r=1}^{m} w_r \delta(Y=1) > \sum_{r=1}^{m} w_r \delta(Y=0) \\ 0 & otherwise \end{cases} \tag{2.70}$$

where $\delta(\cdot)$ is the indicator function and $w_r = \frac{1}{k}$ if x_r is among the k nearest neighbor of x and $w_r = 0$ elsewhere. k is typically a small odd positive integer [92, 43].

In order to identify neighbors, the objects are represented by vectors in a multidimensional variable space and the Euclidean distance is usually chosen. The best choice of k depends on the data. If $k = 1$, then the object is simply assigned to the class of its nearest neighbor. As k increases, the effect of noise is reduced but the boundaries between classes become less distinct. Hence, there is a bias-variance trade-off in the choice of k (in practice $k = 3$ is often taken). A strategy to select the number of neighbors on the basis of a loo criterion is provided by the Lazy Learning algorithm [14].

2.5.3 Support Vector Machine (SVM)

Let us consider a binary classification problem where $Y \in \{-1, 1\}$. Then a hyperplane $y = f(x, \theta) \in \mathbb{R}^n$ in the multidimensional space of inputs implements the following prediction function:

$$\hat{y} = \begin{cases} 1 \ if \ f(x,\theta) > 0 \\ -1 \ if \ f(x,\theta) < 0 \end{cases} \tag{2.71}$$

Such an hyperplane is a linear combination of the inputs,

$$f(x, \theta) = \sum_{i=1}^{n} \theta_i x_i + b = \theta \cdot x + b \tag{2.72}$$

(2.71) combined with (2.72) gives the following inequality:

$$y(\theta \cdot x + b) - 1 \geq 0 \tag{2.73}$$

If two classes of points are linearly separable then there usually exists an infinity of hyperplane that separate the data points. The optimal separating hyperplane is the hyperplane that maximizes the distance between two parallel hyperplanes that are as far as possible from each other but still separating the data. In a SVM setting, we assume that the larger the distance between the two parallel hyperplanes (also called margin in the SVM literature) is the better the generalisation error of the classifier is. The distance between the two hyperplanes is $\frac{2}{\|\theta\|}$. As a result, maximizing the margin is equivalent to minimizing $\frac{1}{2}\|\theta\|^2$ subject to $y(\theta \cdot x + b) - 1 \geq 0$. This optimization problem can be written as [25, 60]

$$L_P = \frac{1}{2}\|\theta\|^2 - \sum_{r=1}^{m} \lambda_r (y_r(\theta \cdot x_r + b) - 1) \tag{2.74}$$

which is called the primal form of the Lagrangian, where λ_r are the Lagrangian multipliers.

Differentiating with respect to θ and equating to zero yields to the following optimiza-

tion (called the dual form of the Lagrangian) [25, 60],

$$L_D = \sum_{r=1}^{m} \lambda_r - \frac{1}{2} \sum_{r=1}^{m} \sum_{s=1}^{m} \lambda_r \lambda_s y_r y_s (x_r \cdot \ x_s) \tag{2.75}$$

where $\forall r,\ \lambda_r \geq 0$ and $\sum_{r=1}^{m} \lambda_r y_r = 0$. Only the λ_r corresponding to the closest points are non-zero, these form the support vectors. Hence, the solution can be found by solving a convex quadratic programming problem.

Non-linear classifications is possible by replacing the classical dot product $(x_r \cdot x_s)$ with a non-linear kernel function $K(x_r, x_s) = (\Phi(x_r) \cdot \Phi(x_s))$. For example, the Gaussian kernel is given by

$$K(x_r, x_s) = e^{-\|x_r - x_s\|^2 / 2\sigma^2} \tag{2.76}$$

where σ is a positive Gaussian kernel width. Note that the kernel function prevents from dealing explicitly with an infinite dimensional space transformation Φ. Hence, the algorithm fits the maximum-margin hyperplane in a higher dimensional feature space. As a result, the classification by an hyperplane in the transformed space may be non-linear in the original input space [25, 60].

2.6 Fast Entropy Estimation

Mutual information computation requires the determination of three entropy terms (see Def. 2.7):

$$I(X;Y) = H(X) + H(Y) - H(X,Y)$$

An effective entropy estimation is then essential for computing mutual information. Entropy estimation has gained much interests over the last decade [30] and most approaches focus on reducing the bias inherent to entropy estimation.

For microarray datasets, bias reduction should not be the only criterion to choose an estimator and bias reduction should be traded with speed/computational complexity of the estimator. Indeed, in variable selection and network inference, mutual information estimation routines are expected to be called a huge number of times and used for estimating tasks with the same number of variables and the same amount of samples. A similar conclusion has been reached in [83].

For this reason, this section focuses on the fastest and most used entropy estimators. We refer the reader to [100, 30, 7, 97, 28] for alternative approaches. These estimators will be used in experiments of Chapter 4.

2.6.1 Empirical Estimator and the Miller-Madow Correction

The empirical estimator (also called "plug-in", "maximum likelihood" or "naive", [100]) is the entropy of the empirical distribution.

$$\hat{H}^{emp}(X) = -\sum_{x\in\mathcal{X}} \frac{\#(x)}{m} \log \frac{\#(x)}{m} \tag{2.77}$$

where $\#(x)$ is the number of data points having value x. Because of the convexity of the logarithmic function, underestimates of $p(x)$ cause errors on $E[\frac{1}{\log p(x)}]$ that are larger than errors due to overestimations. As a result, entropy estimators are biased downwards, that is,

$$E\big[\hat{H}^{emp}(X)\big] \leq H(X). \tag{2.78}$$

It has been shown in [100] that

1. the variance of the empirical estimator is upper-bounded by a term $\left(\frac{(\log m)^2}{m}\right)$ which depends only on the number of samples
2. the asymptotic bias is $-\frac{|\mathcal{X}|-1}{2m}$ and depends on the number of bins $|\mathcal{X}|$ [100]. As $|\mathcal{X}| \gg m$, this estimator can still have a low variance but the bias can become very large [100].

The computation of $\hat{H}^{emp}(X)$ has an $O(m)$ complexity cost.

The Miller-Madow correction is given by the following formula which is the empirical entropy corrected for the asymptotic bias,

$$\hat{H}^{mm}(X) = \hat{H}^{emp}(X) + \frac{|\mathcal{X}|-1}{2m} \tag{2.79}$$

where $|\mathcal{X}|$ is the number of bins with non-zero probability. This correction, while adding no computational cost, reduces the bias without changing variance. As a result, the Miller-Madow estimator is often preferred to the naive empirical entropy estimator.

A jackknifed version of the entropy estimator has also been proposed in the literature [100]

$$\hat{H}^{jk}(X) = m\hat{H}^{emp}(X) - \frac{m-1}{m}\sum_{i=1}^{m} \hat{H}^{emp(-i)}(X) \tag{2.80}$$

where $\hat{H}^{emp(-i)}$ is the empirical estimator computed without sample i.

While less biased, this estimator remains biased since it is an average of biased estimators [100]. Also, it has an $O(m^2)$ complexity cost since it requires the computation of m

empirical entropies, each on $m-1$ samples. This makes this estimator less interesting in tasks requiring a large number of entropy estimations.

2.6.2 Shrink estimator

The rationale of the shrink estimator [61], is to combine two different estimators, one with low variance and one with low bias, by using a weighting factor $\lambda \in [0,1]$

$$\hat{p}_\lambda(x) = \lambda \frac{1}{|\mathcal{X}|} + (1-\lambda)\frac{\#(x)}{m} \tag{2.81}$$

Shrinkage is a well-known technique commonly used to improve estimators for small sample sizes [118, 119]. Let λ^* be the value minimizing the mean square error, see [61],

$$\lambda^* = \arg\min_{\lambda\in[0,1]} E\big[\sum_{x\in\mathcal{X}}(\hat{p}_\lambda(x) - p(x))^2\big] \tag{2.82}$$

It has been shown in [118, 119] that the optimal λ is given by

$$\lambda^* = \frac{|\mathcal{X}|(m^2 - \sum_{x\in\mathcal{X}}\#(x)^2)}{(m-1)(|\mathcal{X}|\sum_{x\in\mathcal{X}}\#(x)^2 - m^2)} \tag{2.83}$$

The entropy can then be estimated by

$$\hat{H}^{shrink}(X) = -\sum_{x\in\mathcal{X}} \hat{p}_{\lambda^*}(x)\log\hat{p}_{\lambda^*}(x) \tag{2.84}$$

For values of λ^* close to one, the estimated entropy is moved toward the maximal entropy (uniform probability) whereas when λ^* is zero the estimated entropy boils down to the empirical one.

2.6.3 The Schurmann-Grassberger Estimator

The Schurmann-Grassberger estimator is a Bayesian parametric method that assumes samples distributed following a Dirichlet distribution. The Dirichlet distribution is the multivariate generalization of the Beta distribution [95]. More precisely, the density of a Dirichlet distribution takes the following form

$$p(X;\theta) = \frac{\prod_{i\in\{1,2,...|\mathcal{X}|\}}\Gamma(\theta_i)}{\Gamma(\sum_{i\in\{1,2,...|\mathcal{X}|\}}\theta_i)} \prod_{i\in\{1,2,...|\mathcal{X}|\}} x_i^{\theta_i - 1} \tag{2.85}$$

where θ_i is the prior probability of an event x_i, x_i being the ith element of the set $\mathcal{X}$ and $\Gamma(\cdot)$ is the gamma function (see [61, 143, 97] for more details). In case of a priori ignorance, the θ_i are all set to a fixed number N ($\theta_i = N,\ i \in \{1,2,...|\mathcal{X}|\}$) to mean that no event

becomes more probable than another. Note that using a Dirichlet prior with parameter N is equivalent to adding $N \geq 0$ "pseudo-counts" to each event $i \in \{1, 2, ...|\mathcal{X}|\}$. The prior actually provides the estimator the information that $|\mathcal{X}|N$ counts have been observed in previous experiments. From that viewpoint, $|\mathcal{X}|N$ becomes the a priori sample size.

The entropy of a Dirichlet distribution can be computed directly with the following equation:

$$\hat{H}^{dir}(X) = \frac{1}{m + |\mathcal{X}|N} \sum_{x \in \mathcal{X}} (\#(x) + N)(\psi(m + |\mathcal{X}|N + 1) - \psi(\#(x) + N + 1)) \quad (2.86)$$

with $\psi(z) = \frac{d \ln \Gamma(z)}{dz}$ is the digamma function [61, 143, 97].

Various choices of prior parameters have been proposed in the literature:

1. $N = 1$ known as Laplace's prior [6],
2. $N = 0$ which boils down to consider frequencies as probabilities [143],
3. $N = 1/2$ known as Jeffrey's prior [78],
4. $N = \frac{1}{|\mathcal{X}|}$, the Schurmann-Grassberger estimator [121], which is a less biased prior [97].

Although most choices of prior have been used for entropy estimation, it was shown in [97] that the choice of the prior often dominates the entropy estimation. As a result, an algorithm has been proposed in [97] to choose a prior so that the distribution of the entropy is close to being uniform. The latter estimator, called NSB, was shown to outperform the other entropy estimators based on Dirichlet priors [97]. However, higher accuracy comes at the cost of a complexity $O(m^2)$.

2.6.4 Entropy of a Normal Distribution

Let X be a multivariate Gaussian, having a density function,

$$f(X) = \frac{1}{\sqrt{(2\pi)^n |\mathrm{CO}|}} exp^{(-\frac{1}{2}(x-\mu)^T \mathrm{CO}^{-1}(x-\mu))} \quad (2.87)$$

with mean μ and covariance matrix CO.

The entropy[2] of this distribution is [26]

$$H(X) = \frac{1}{2} \ln\{(2\pi e)^n |\mathrm{CO}|\} \quad (2.88)$$

[2]It is rather the differential entropy, see [26].

where |CO| is the determinant of the covariance matrix [26].

As a result, the mutual information between two normal distributions is given by [62]

$$I(X_i, X_j) = \frac{1}{2} \log \left(\frac{\sigma_{ii}\sigma_{jj}}{|\mathrm{CO}|} \right) \tag{2.89}$$

where σ_{ii} and σ_{jj} are the standard deviations of X_i and X_j respectively. Hence

$$I(X_i, X_j) = -\frac{1}{2} \log(1 - \rho^2) \tag{2.90}$$

with ρ being the Pearson's correlation [71] between X_i and X_j.

2.6.4.1 Squared Pearson's Correlation and Squared Spearman Rank Correlation Coefficient

Since only the ranking of entropies is required for variable selection, using directly the squared Pearson's correlation ρ^2 is more effective, with

$$\hat{\rho}^2 = \frac{(\sum_{i=1}^{m} x_i y_i - \alpha)^2}{(\sum_{i=1}^{m} x_i^2 - \alpha)(\sum_{i=1}^{m} y_i^2 - \alpha)} \tag{2.91}$$

where

$$\alpha = \frac{1}{m}(\sum_{i=1}^{m} x_i)(\sum_{i=1}^{m} y_i)$$

The complexity of $\hat{\rho}^2$ is in $O(m)$, where m is the number of samples.

The Spearman rank correlation coefficient [71] is a special case of the Pearson correlation in which the data are converted to ranks. Unlike the Pearson correlation coefficient, the Spearman rank correlation coefficient is able to detect any kind of monotone relation without making any assumptions about the frequency distribution of the variables.

2.7 Discretization Method

All the estimators discussed in the previous section, apart from the Gaussian one, have been designed for discrete variables. If the random variable X_i is continuous and can take real values lying between a and b, then it is required to partition the interval $[a, b]$ into $|\mathcal{X}_i|$ sub-intervals in order to adopt a discrete entropy estimator. The two most used discretizing algorithms are the equal width and the equal frequency quantization presented here. Other discretization methods can be found in [41, 81, 146].

2.7.1 Equal Width

The principle of the equal width discretization is to divide $[a, b]$ into $|\mathcal{X}_i|$ sub-intervals of equal size [146, 41, 81]:

$$[a,\ a + \frac{b-a}{|\mathcal{X}_i|}[, [a + \frac{b-a}{|\mathcal{X}_i|},\ a + 2\frac{b-a}{|\mathcal{X}_i|}[, ...[a + \frac{(|\mathcal{X}_i| - 1)(b-a)}{|\mathcal{X}_i|},\ b + \varepsilon[$$

Note that an $\varepsilon > 0$ is added in the last interval in order to include the maximal value in one of the $|\mathcal{X}_i|$ bins. This discretization scheme has a $O(m)$ complexity cost.

2.7.2 Equal Frequency

The equal frequency discretization scheme consists in partitioning the interval $[a, b]$ into $|\mathcal{X}_i|$ intervals, each having the same number, $m/|\mathcal{X}_i|$, of data points [41, 145, 81]. As a result, the intervals can have different sizes. If the $|\mathcal{X}_i|$ intervals have equal frequency, then the computation of entropy is straightforward: $\log \frac{1}{|\mathcal{X}_i|}$. However, if one of the bins is more dense than the others, then the resulting entropy needs to be estimated. This discretization is reported [146] as one of the most efficient method (combined with the naive Bayes classifier) .

2.7.3 The choice of $|\mathcal{X}_i|$

The value of the number of bins $|\mathcal{X}_i|$ controls the bias-variance trade-off. With a too high $|\mathcal{X}_i|$, each bin will contain a few number of points, hence the variance is increased, whereas a too low $|\mathcal{X}_i|$ will introduce a too high loss of information [26]. Two classical choices of $|\mathcal{X}_i|$ are discussed below.

2.7.3.1 The number of samples square root $\sqrt{m}$

In practice, $|\mathcal{X}_i| = \sqrt{m}$ is considered to be a fair trade-off between bias and variance [145]. One justification given for that choice is that the ratio $m/|\mathcal{X}_i|$ becomes $m/\sqrt{m} = \sqrt{m}$, hence there are as many bins as the average number of points per bin. Note also, that when estimating the entropy of a bivariate distribution where each variable has $\sqrt{m}$ bins, the number of bins of the joint distribution is upper-bounded by $|\mathcal{X}_i| \leq \sqrt{m} \times \sqrt{m} = m$. As a result, the empirical entropy estimator is not in the undersampled regime and should not have a too high bias when combined with this choice of $|\mathcal{X}_i|$.

2.7.3.2 Scott's criterion

A criterion based on the standard error estimate of a normal distribution has been proposed in [122]. It consists in choosing the number of intervals as a function of the

Name	Notation	Definition
Kullback-Leibler divergence	$KL(p;q)$	2.1
Entropy	$H(X)$	2.4
Conditional entropy	$H(Y\|X)$	2.6
Mutual information	$I(X;Y)$	2.7
Conditional Mutual information	$I(X;Y\|Z)$	2.8
Interaction information	$C(X;Y;Z)$	2.10
Multiinformation	$R(X;Y;Z)$	2.11

Table 2.3: Information-theoretic measures of the chapter

variance σ^2 of the distribution, the number of samples m and the range $(b-a)$ of X_i. More precisely, $|\mathcal{X}_i| = \left\lfloor \frac{(b-a)}{h} \right\rfloor$ where $h = 3,5 * \frac{\sigma}{\sqrt[3]{m}}$. The motivation for this criterion can be found in [122].

2.8 Conclusion

Information theory deals with various functions of probability distributions to measure predictability and dependency of random variables (see Tab. 2.3). The Kullback-Leibler divergence, entropy, mutual information, multiinformation and interaction information have been defined. In the chapter 3, these notions are used to define relevance, redundancy and complementarity of variables.

In a finite sample size setting, a probability distribution can be estimated by identifying the family, the number and the values of the parameters θ_m to describe a target distribution. In other words, a statistical model is built by carrying out two steps:

1. a structural identification which consists in finding the "best" class Λ of models for the target (the family and the number of parameters).

2. a parametric identification whose goal is to find the "best" model from the chosen class of models (the values of the parameters θ).

Hence, in this approach, not only the values of the parameters but also the number of the parameters θ depend on the data. As the number of parameters increases, the bias of a model is reduced but the variance is increased. A higher number of parameters may then reduce the estimation accuracy.

We have seen three machine learning algorithms (naive Bayes, nearest neighbors and SVM), five entropy estimators (empirical, Miller-Madow, shrink, Schurmann-Grassberger, Gaussian) and two discretization methods (equal frequency and equal width). In chapter 4, these algorithms are used in the experimental sessions. These methods all deal, explicitly or

not, with the bias-variance trade-off in order to extract information from data. In chapter 2.4, the bias-variance trade-off is used as a motivation to eliminate variables from models.

Chapter 3

Variable Selection and Network Inference: State-of-the-Art

This chapter presents the state-of-the-art of the two topics of this thesis: variable selection and network inference. Variable selection literature is reviewed in the first three sections and network inference is discussed in the last three ones. Section 3.1 deals with variable selection theory and the two following sections address the two main steps of a variable selection strategy: exploration of the space of subsets (Section 3.2) and evaluation (performance measure) of subsets (Section 3.3). The fourth section is devoted to the (undirected) network inference problem definition and the various assessment and validation tools used in the field. The fifth section focuses on mutual information networks, a category of (undirected) network inference strategies able to deal with large number of variables. Finally, the last section aims to connect Bayesian network inference (directed network inference), causality and information theory.

3.1 Part I: Variable Selection

Variable selection is a combinatorial optimization problem where a subset of variables is selected on the basis of statistical estimates of its performances.

We have seen in Chapter 2 that the generalization performance of a model is related to its structure and its number of parameters θ. The number of parameters is typically related to the number of inputs. For example, in a linear model, there is one parameter per variable. Non-linear models happen to require several parameters per variable. Hence, in general, adding variables boils down to adding parameters. As a result, "*feature* (variable) *selection is nothing else than a particular form of model selection*"[112].

The selection of variables is an important step in learning. It is often preferred to have rough estimations of the parameters of a model constituted by adequate inputs rather than

having good estimations of parameters with poorly relevant variables. As a result, one of the objectives of variable selection is to eliminate "useless" variables, which add variance without any bias reduction. Another positive side effect of eliminating variables is that it increases the intelligibility of a model, and at the same time, it decreases the measurements and storage requirements [56]. However, by eliminating a variable, its information is lost. Let $X = (X_S, X_R)$ be composed of two subsets of variables, X_S standing for the selected variables and X_R for the remaining or eliminated variables. By definition (Definition 2.8), we have,

$$H(Y|X) = H(Y|(X_S, X_R)) = H(Y|X_S) - I(X_R; Y|X_S) \tag{3.1}$$

As a result, if X_R possesses some information on Y given X_S, i.e., if

$$I(X_R; Y|X_S) > 0$$

then eliminating X_R increases the uncertainty on the output variable, i.e.

$$H(Y|(X_S, X_R)) \leq H(Y|X_S)$$

However, considering the bias-variance trade-off (2.61), eliminating information increases noise but improves the reliability (less variance) of the estimation. All variables are not equivalent regarding their noise and variance contributions.

3.1.1 Variable Selection Problem

For n input variables, the number of possible subsets is 2^n. Hence the space of solutions is exponential in the number of variables. Various results, like the following theorem by Cover and Van Campenhout [35], show that variable selection is a hard problem,

Theorem 3.1: *Let $X_{S_1}, X_{S_2}, ..., X_{S_{2^n}}$ be an ordering of the 2^n subsets of $A = \{1, 2, ..., n\}$, satisfying the consistency property $i < j$ if $S_i \subset S_j$ (therefore, $S_1 = \emptyset$ and $S_{2^n} = \{1, 2, ..., n\}$). Then there exists a distribution of the random variables (X, Y) such that*

$$L^*(X_{S_1}) > L^*(X_{S_2}) > ... > L^*(X_{S_{2^n}}) \geq 0 \tag{3.2}$$

Given an empty set of input variables $S_1 = \emptyset$, the Bayes error $L^*(X_{S_1})$ (2.44) is maximal. As long as we add input variables to the model, the probability of error decreases and the best probability of error $L^*(X_{S_{2^n}})$ could be reached when all the variables are selected.

In many problems, estimating a model with all the variables is impossible because of the variance. The problem of estimating the error and the conditional probability distribution

from a limited amount of samples is not considered by this theorem.

Example 3.1: *The following ordering (allowed by Theorem 3.1)*

$$L^*(X_3) > L^*(X_2) > L^*(X_1) > L^*(X_{1,2}) > L^*(X_{1,3}) > L^*(X_{2,3})$$

gives us X_1 as the best predictor and X_3 as the worst one (when taken alone) but the best subset of two variables $X_{2,3}$ is made of the two worst variables and the worst subset $X_{1,2}$ is made of the two best variables.

This example can be extended to any subset of k variables. As a consequence, any algorithm that aims at identifying the best subset of size k, out of n variables, must necessarily investigate all the $\begin{pmatrix} n \\ k \end{pmatrix}$ possible combinations [35].

The following theorem, shown in [31], is another result that expresses the difficulty of the variable selection problem.

Theorem 3.2: *[31]The search of a subset of discrete variables X_S of size $|S| = k$ such that there exists no two samples in a dataset D_m that have identical values for all the variables in the subset X_S but different values for the target, is NP-complete.*

3.1.2 Relevant and Redundant Variables

The preceding results are rather pessimistic since exhaustive search is intractable in the presence of a large number of variables. Although the optimal variable set is not reachable in a reasonable amount of time, a number of experimental studies [11, 74, 108] have shown that the removal of irrelevant and redundant variables can dramatically improve the predictive accuracy of models built from data. We introduce here notions, like redundancy, that can guide the search towards a good subset of variables.

Three degrees of relevance are defined in *[74]*:

Definition 3.1: *[Strong and weak relevance]A variable X_j is "strongly relevant" in X iff there exists some x_j, y and x_{-j} for which $p(x) > 0$, such that*

$$p(y|x) \neq p(y|x_{-j})$$

A variable X_j is "weakly relevant" iff it is not strongly relevant, but there exists a subset X_S of variables of X_{-j} for which there exists some x_j, y and x_S with $p(x_j, x_S) > 0$ such that

$$p(y|x_j, x_S) \neq p(y|x_S)$$

A variable is irrelevant iff it is not relevant (weakly or strongly).

In other words, an input variable X_j is strongly relevant if the removal of X_j alone will result in a change of the conditional probability distribution of Y. A variable is weakly relevant if it is not strongly relevant, but in some context X_S it may change the conditional probability distribution of Y.

For example, let us consider a trivial prediction problem with four variables: the distance between two cities in kilometers, the distance between two cities in miles, the flight time from one city to the other, the colour of the plane. Suppose we want to determine at which average speed the plane flies. The flight time is a strongly relevant variable because it is relevant whatever the selected subset of variables from the dataset. The two distances are weakly relevant variables because each one is relevant in the absence of the other. Finally, the colour of the plane is irrelevant whatever the subset. In practice, it is difficult to discover strong relevance from datasets with many variables, because it requires the estimation of the stochastic relationship between the target variable and all the inputs.

Another notion which is important in order to identify irrelevant or redundant variables is the "Markov blanket" [101, 76].

Definition 3.2: *Let $X_{M_j} \subset X_{-j}$. X_{M_j} is a Markov blanket for X_j if X_j is conditionally independent of $(Y, X_{-(j,M_j)})$ given X_{M_j}*

In other words, the Markov blanket X_{M_j} does not only subsume the information of X_j about Y but also the information of X_j about all the other variables $X_{-(j,M_j)}$ which are not in the blanket. Although the existence of a Markov blanket allows to drop a variable without any risk of information loss, it is not sufficient to distinguish between irrelevant and redundant variables. The additional condition to label a variable as redundant was introduced in [148]:

Definition 3.3: *$X_i \in X$ is redundant, iff it is weakly relevant and has a Markov blanket in the set X.*

The definitions of relevant and redundant variables are binary, that is, according to them a variable is either relevant or irrelevant. It would be useful for practical purposes to have a measure of relevance/redundancy in order to rank/select variables. In the following sections, we introduce existing measures of relevance and redundancy .

3.1.3 Relevance measure

The mutual information (introduced in Definition 2.7) is a natural measure of relevance since it quantifies the dependency level between random variables. The use of mutual information and conditional entropy as relevance measure traces back to [27]. Later, [130]

introduced a selection criterion called the *information bottleneck* which uses also mutual information as a relevance measure.

In [8], the following information-theoretic definition of variable relevance is given,

Definition 3.4: *The variable relevance $VR(X_i; Y|X_S)$ of X_i to a target Y given a context X_S is,*

$$VR(X_i; Y|X_S) = \begin{cases} \frac{I(X_i;Y|X_S)}{H(Y|X_S)} & \text{if } H(Y|X_S) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

This definition states that the variable relevance is a function $0 \leq VR(X_i; Y|X_S) \leq 1$ that indicates the relative reduction of uncertainty of Y knowing X_S once the information of X_i is given. This measure of variable relevance is a normalized conditional mutual information (constrained between zero and one according to (2.43)).

The mutual information $I(X_S; Y)$ as a measure of relevance has been proposed in [75].

Definition 3.5: *The relevance of a subset X_S to a target variable Y is $I(X_S; Y)$*

The relevance of a variable X_i, given a context X_S, is evaluated as the gain brought by the variable X_i to the context X_S.

$$I(X_i; Y|X_S) = I(X_{S,i}; Y) - I(X_S; Y) \tag{3.4}$$

In this work, we adopt the following definition of the relevance measure:

Definition 3.6: *Given three random variables X_i,Y and X_S, the relevance of the variable X_i to Y given the context X_S is $I(X_i; Y \mid X_S)$*

The measure $I(X_i; Y|X_S)$ is the non-normalized version of $VR(X_i; Y|X_S)$. One major aspect of variable selection is that relevance is *conditionally dependent* on the context X_S. This makes difficult to decide whether a variable should be labeled as *relevant* or not. In fact, a variable can have a significant relevance given a context and a null relevance given another one, see Example 2.1. This is in accordance with [74, 148], i.e., that a variable can be labeled as "relevant" only with respect to a given set of variables. In order to instantiate the notion of relevant variable in an information-theoretic framework, we can reformulate Definition 3.1 of [74, 148] from the following result (proof given in Appendix C),

Theorem 3.3: [Information-theoretic weak and strong relevance] *Given a set X of input variables*
A variable X_i is irrelevant to Y if:

$$\forall X_S \subseteq X_{-i} \ : \ I(X_i; Y|X_S) = 0 \tag{3.5}$$

A variable X_i is strongly relevant to Y if:

$$I(X_i; Y|X_{-i}) > 0 \tag{3.6}$$

A variable X_i is weakly relevant to Y if:

$$I(X_i; Y|X_{-i}) = 0 \quad AND \quad \exists X_S \subset X_{-i} \ : \ I(X_i; Y|X_S) > 0 \tag{3.7}$$

Note that the reformulation of Definition 3.1 in information-theoretic terms establishes a link between [74, 148] and the relevance measure (mutual information) used by [130, 8, 75, 27].

3.1.4 Redundancy measure

According to [141], a redundancy measure should be symmetric, non-negative and non-decreasing with the number of variables. The monotonicity is justified by the fact that, unlike relevance, the amount of redundancy of a variable can never decrease when more variables are added. As a result, [141] proposed to use multiinformation as redundancy measure, i.e.,

$$R(X_i; ...; X_n) = \sum_{i=1}^{n} H(X_i) - H(X_{1,..,n}) \tag{3.8}$$

This measure was also used in [104, 131] for the two-variables case, i.e.,

$$R(X_i; X_j) = I(X_i; X_j) \tag{3.9}$$

Note that while the relevance measure concerns the relation between inputs and output, the redundancy measure applies exclusively to input variables.

The simplest case of redundant variable is encountered when the measure of redundancy takes its maximum value, i.e., a variable X_i is redundant with respect to a set X_S if $R(X_i; X_S) = I(X_i; X_S) = H(X_i)$ or equivalently if $H(X_i|X_S) = 0$. This is due to the fact that $H(X_i|X_S) = 0$ is equivalent to $I(X_i; Y|X_S) = 0$ as shown by the following derivation

$$I(X_i; Y|X_S) = H(X_i|X_S) - H(X_i|Y, X_S) = -H(X_i|Y, X_S) \geq 0 \Leftrightarrow \tag{3.10}$$

$$H(X_i|Y, X_S) = 0 \Leftrightarrow I(X_i; Y|X_S) = 0 \tag{3.11}$$

which holds because of the non negativity of the entropy. In other terms, if $H(X_i|X_S) = 0$, we may drop the variable X_i since it is not able to bring any further conditional information to Y. Note that if the measure of redundancy is maximal, i.e., $I(X_i; X_S) = H(X_i)$, then X_S is a Markov blanket of X_i since in that case $I(X_i; Y, X_{-i,S}|X_S) = 0$.

Theorem 3.4: [Information-theoretic Markov blanket] *Let X be a set of variables containing the variable X_j, X_{M_j} some subset of X_{-j} and $X_{-(j,M_j)}$ the set of remaining variables. X_{M_j} is a Markov Blanket for X_j, iff $I(X_j; (Y, X_{-(j,M_j)})|X_{M_j}) = 0$.*

Using the information-theoretic Markov blanket theorem (whose proof is given in appendix C), the definitions of redundancy of [141, 104, 148], i.e., Definition 3.3, 3.8 and 3.9, are brought together.

3.1.5 Complementarity measure

Variable complementarity measures the increase of information coming from a combination of variables with respect to the information coming from sub-combinations of variables. Variable complementarity has been observed in several studies [56, 74]. However, this phenomenon was first formalized in the variable selection process by [67] using a three-way interaction gain defined as

$$IG = I(X_{i,j}; Y) - I(X_i; Y) - I(X_j; Y) \tag{3.12}$$

This quantity measures the gap between the joint mutual information of two variables X_i and X_j with Y and the sum of the mutual informations. The joint information of two random variables, i.e., $I(X_{i,j}; Y)$ can be higher than the sum of their individual information $I(X_i; Y)$ and $I(X_j; Y)$. In [68] the interaction information $C(X_i; X_j; Y)$ (Definition 2.10) is adopted as a measure of interacting variables since this measure reduces to the opposite interaction gain when used on two input variables.

Hence, in this work, we define complementary variables as following.

Definition 3.7: *$X_1, X_2, ..., X_n$ are said to be* complementary *if*

$$(-1)^n C(X_1; X_2; ...; X_n) > 0$$

The complementarity effect, also known as negative interaction in [68], has recently been explicitly used in variable selection algorithms [75, 86, 150]. Figure 3.1 illustrates a sampling of complementary variables given by the problem $Y = X_1 \oplus X_2$, where $\oplus$ is the exclusive disjunction function (XOR) (see Example 2.3).

Complementarity provides an alternative way to interpret Theorem 3.1. Since variables can have a null relevance taken separately and a maximal relevance once taken together, it follows that an optimal combination cannot be found on the basis of its subsets.

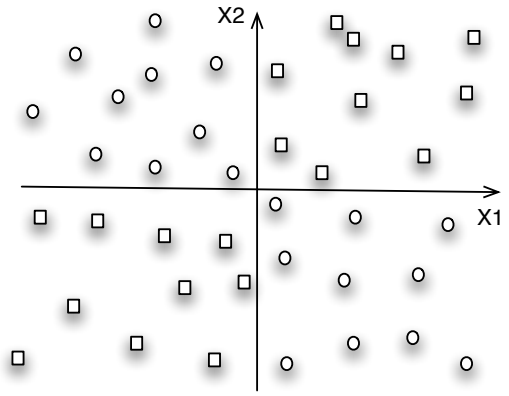


Figure 3.1: XOR sampling: there are two inputs, X_1 and X_2, and the output $Y = X_1 \oplus X_2$ can take two values: square or bullet.

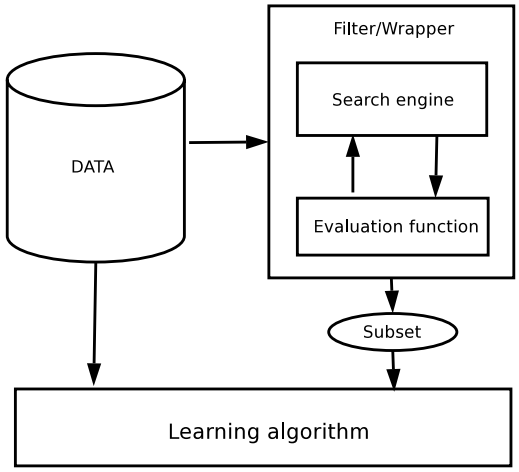


Figure 3.2: principle of a filter/wrapper approach

3.2 Variable Selection Exploration Strategies

According to [11, 74], there are three different approaches of variable selection: embedded methods, filter methods and wrappers methods. In the first category, the process of variable selection is embedded in the learning algorithms. Examples are the pruning of a decision tree [110], the progressive non-zero weight of a neural network [105], the ridge regression or the lasso [91, 60]. In the wrapper and in the filter approaches, variable selection is seen as a preprocessing step of learning (see Figure 3.2). This step consists in a search for a best subset of variables in the power set 2^X where X denotes the set of random variables. Hence, it is an example of combinatorial optimization problem [74] which depends on:

1. a method of exploring the space (including the starting point and stop criterion),
2. an evaluation function.

More formally the problem is, given n input variables X and a performance measure $\Phi : 2^X \rightarrow \mathbb{R}$, find the subset $X_S \subset X$ which maximizes the performance ,

$$X_S^{max} = \arg \max_{X_S \in 2^X} \Phi(X_S) \tag{3.13}$$

Exploration strategies can be classified into three main categories of combinatorial optimization algorithms namely *optimal search*, *stochastic search* and *sequential search* (see [57], chapter 4).

1. Optimal search strategies include exhaustive search and branch-and-bound methods [57]. Their high computational complexity makes them impracticable with a high number of inputs and, for this reason, they are not discussed in this work.
2. Stochastic search strategies are also called *randomized* or *non-deterministic* [107] because two runs of these methods (with the same inputs) will not necessarily bring the same result [42]. These methods explore a smaller portion of the search space 2^X by using rules often inspired by nature. Some examples are: simulated annealing [42], tabu search [42] and genetic algorithms [144, 42]. Although these methods can perform well in variable selection tasks [144], these are not explored in this work.
3. Sequential search strategies are also called *deterministic heuristics* [107]. These methods are widely used for variable selection [44, 103, 49, 20]. Most of them use a neighbor search (two subsets are said neighbors if they differ from one variable) to discover a local optimum. Some examples are: forward selection (see Section 3.2.1), backward elimination (see Section 3.2.2), bi-directional search (see Section 3.2.3). These sequential strategies are the topic of this section.

In the following, we denote by X the complete initial set of variables and by $X_i^{METH} \in X,\ i \in A = \{1, 2, ..., n\}$ the variable selected at each step by the method $METH$. X_S and X_R are the set of selected variables and the set of remaining variables respectively. X_i or X_j usually denote a variable in X_R or in X_S, respectively.

3.2.1 Forward Selection search

Forward Selection [20, 74] is a sequential search method that starts with an empty set of variables, $X_S = \phi$. At each step, it selects the variable X_t that brings the best improvement (in terms of a given evaluation criterion $\Phi(\cdot)$). A pseudo-code of the method

Algorithm 3.1 Pseudo-code of the forward selection search for variable selection

Inputs: input variables X, the output variable Y, a maximal subset size $d > 0$ and a performance measure $\Phi(\cdot)$ to maximize,
$X_S \leftarrow \phi$
$X_R \leftarrow X$
while $(|X_S| < d)$
  $maxscore \leftarrow -\infty$
  for all the inputs X_i in the search space X_R
    Evaluate $\Phi(X_{S,i})$ for the variable X_i with X_S the subset of selected variables.
    if ($\Phi(X_{S,i}) > maxscore$)
      $X_t \leftarrow X_i$
      $maxscore \leftarrow \Phi(X_{S,i})$
    end-if
  end-for
  $X_S \leftarrow X_{S,t}$
  $X_R \leftarrow X_{R-t}$
end-while
Output: the subset X_S

is given in Algorithm 3.1. As a consequence of the sequential process, each selected variable influences the evaluations of the following steps.

This search has been widely used in variable selection, (see [20, 11, 74]). The forward selection algorithm selects a subset of $d < n$ variables in d steps and explores only $\sum_{i=0}^{d-1}(n-i)$ subsets.

However, this search has some weaknesses:

1. two variables that are useful together (e.g. complementary, Definition 3.7) can appear as not relevant once taken individually, hence ignored by this procedure,

2. selecting the best variable at each step does not mean selecting the best subset (see Section 3.1.1).

3.2.2 Backward Elimination search

Backward elimination [20, 74, 91] is a search method that starts by evaluating a subset containing all the variables $X_S = X$ and progressively discards the least relevant variables. For instance, at the second step, the method compares n subsets of $n-1$ inputs. The variable X_t associated with the smallest change of the performances is eliminated. The process is repeated until it yields the chosen number of inputs d (see Algorithm 3.2). This method does not suffer from the risk of ignoring a pair of complementary variables as it is the case for forward selection.

Algorithm 3.2 pseudo code of backward elimination for variable selection

Inputs: input variables X (the input space), the output variable Y, a minimal subset size $d > 0$ and a performance measure $\Phi(\cdot)$ to maximize,
$X_S \leftarrow X$
while ($|X_S| > d$)
  $worstscore \leftarrow \infty$
  for all inputs X_j in the subset X_S
    Evaluate $\Phi(X_{S-j})$, with all inputs of the subset X_S without X_j
    if ($\Phi(X_{S-j})$<$worstscore$)
      $X_t \leftarrow X_i$
      $worstscore \leftarrow \Phi(X_{S-j})$
    end-if
  end-for
  $X_S \leftarrow X_{S-t}$
end-while
Output: the subset X_S

3.2.3 Bi-directional search

The strengths of the forward selection and of the backward elimination can be combined in different manners.

As an example, let 26 random variables constitute the search space and be denoted by letters of the alphabet. Let the best subset of four variables be denoted by the letters $\{B, E, S, T\}$. The forward and the backward approaches can be combined in different ways:

- by using a backward elimination on a subset selected with a forward search [20].
  If a forward selection has selected the subset $\{C, B, E, S, T, D, F, G\}$, then, we can use a backward elimination in order to keep the most important variables of the subset, and reach the subset $\{B, E, S, T\}$.

- by performing a stepwise approach [91, 20]: At each step, choose the best action between eliminating a variable or selecting one.
  In our example, we may at some stage have selected the subset $\{E, A, S, T\}$. The stepwise algorithm chooses between adding a variable that brings the best improvement $\{B, E, A, S, T\}$ or eliminating the less important variable $\{E, S, T\}$.

- by using sequential replacement [91, 20]: This procedure consists in replacing $k \geq 1$ variables at each step.
  In our example, we can imagine at some stage having the subset $\{P, E, S, T\}$ that becomes the subset $\{B, E, S, T\}$ after an iteration. The pseudo-code of the algorithm for $k = 1$ is described in Algorithm 3.3.

Algorithm 3.3 Pseudo-code of the sequential replacement algorithm (with replacement size $k = 1$)

Inputs: a selected subset of inputs X_S, the set of remaining variables X_R, the output variable Y and a performance measure $\Phi(\cdot)$ to maximize
do
  for all inputs X_i in the remaining variables X_R
    Evaluate $\Phi(X_{S,i})$
  end-for
  $X_{t1} \leftarrow \arg\max_{X_i} \Phi(X_{S,i})$
  for all for all inputs X_j in the subset X_S
    Evaluate the $\Phi(X_{S-j})$
  end-for
  $X_{t2} \leftarrow \arg\max_{X_j} \Phi(X_{S-j})$
  $X_S \leftarrow X_{(S,t1)-t2}$
  $X_R \leftarrow X_{(R,t2)-t1}$
end-do while $X_{t1} \neq X_{t2}$
Output: the subset X_S

3.3 Information-Theoretic Evaluation Functions

Two main categories of evaluation functions have been used in variable selection: the filter and the wrapper approaches.

The Wrapper Approach In the wrapper approach, the evaluation function is the validation outcome of a learning algorithm (see Section 2.5). These methods are typically too computationally costly to be used in a variable selection task dealing with several thousands of variables because of the validation cost. A k-cross-validation (see Section 2.4.1.2), for example, is computationally expensive because learning is repeated for each of the k learning sets. Although some learning algorithms such as the naive Bayes (Section 2.5.1) or the lazy learning (Section 2.5.2) have a small validation cost compared to other learning methods [92, 14], these methods cannot scale up to thousands variables. Hence, in this work, the wrapper approach is not treated.

The Filter Approach Filter approaches differentiate from wrappers since their evaluation function does not use a learning algorithm [11, 107, 76, 74]. Evaluation functions of filters are mainly: inter-class distance measures [77], information-theoretic measures [76] or probabilistic dependence measures [29]. These measures are considered as an intrinsic property of the raw data because they use rather "simple" models compared to functions returned by learning algorithms. As a consequence, evaluation functions of filters may be fast and adapted to large number of variables.

Some examples of filter algorithms are information-theoretic filters [38, 44, 49, 65, 148] that are detailed in the following.

3.3.1 Information-theoretic filters

Let us start by stating the objective of an information-theoretic filter:

Given a training dataset D_m of m samples, an output variable Y, n input variables X and an integer $d < n$, find the subset $X_S \subset X$ of size d that maximizes the mutual information $I(X_S; Y)$.

In other words, the objective of *filters variable selection* is to find the subset X_S, with $|X_S| = d$, such as:

$$X_S^{max} = \arg \max_{X_S \subset X : |X_S| = d} I(X_S; Y), \text{ d fixed} \tag{3.14}$$

This is a particular case of (3.13), where the evaluation function $\Phi(X_S)$ is the mutual information $I(X_S; Y)$ and where the subset size d is fixed. The quantity d influences the bias-variance trade-off (Theorem 2.5). We assume that the number of variables d has been determined by some a priori knowledge or by some cross-validation techniques. As filters often rank the variables according to their relevance measure, variables can be added one by one in a predictive model, until the cross-validated performances decrease. This procedure allows to reach an adequate number of variables for a given predictive model. Other strategies can be adopted such as the information bottleneck [130], Bayesian confidence on parametric estimations [72] or resampling techniques [50] .

In the following, we review the most important filter selection methods found in the literature which are based on information theory. We present the algorithms by stressing when and where the notion of relevance (Section 3.1.3), redundancy (Section 3.1.4) and complementarity (Section 3.1.5) are used.

3.3.2 Variable Ranking (RANK)

This method (RANK) returns a ranking of variables on the basis of their individual mutual informations with the output. This means that, given n input variables, the method first computes n times the quantity $I(X_i, Y)$, $i = 1, \ldots, n$, then ranks the variables according to this quantity and eventually discards the least relevant ones [44, 5].

The main advantage of this method is its low computational cost. Indeed, it requires only n computations of bivariate mutual information. The main drawback derives from the fact that possible redundancies between variables are not taken into account. Indeed, two redundant variables, yet highly relevant taken individually, will be both well-ranked. On the contrary, two variables could be complementary to the output (i.e., highly relevant together) while being poorly relevant once each taken individually (see Section 3.1.5). As

a consequence, these variables could be badly ranked, or even eliminated, by a ranking filter.

3.3.3 Fast Correlation Based Filter (FCBF)

The Fast Correlation Based Filter is a ranking method combined with a redundancy analysis which has been proposed in [148]. The FCBF starts by selecting the variable (in the remaining variables X_R) with the highest mutual information, denoted by X_i^{FCBF}. Then, all the variables which are less relevant to Y than redundant to X_i^{FCBF} are eliminated from the list. For example, X_i is removed from the remaining variable set X_R if

$$I(X_i; X_i^{FCBF}) > I(X_i; Y)$$

At the next step, the algorithm repeats the selection and the elimination steps. The procedure stops when no more variable remains to be taken into consideration.

In other words, at each step, the set of selected variables X_S is updated with the variable

$$X_i^{FCBF} = \arg \max_{X_i \in X_R} I(X_i; Y) \tag{3.15}$$

and the set of remaining variables X_R is updated by removing the set

$$\{X_i \in X_{R-i} : \ I(X_i; Y) < I(X_i; X_i^{FCBF})\} \tag{3.16}$$

This method is affordable because a few (less than n^2) evaluations of bivariate mutual information are computed. However, although the method addresses redundancy, it presents the risk of eliminating relevant variables. Indeed, it is possible that $I(X_i; X_i^{FCBF}) > I(X_i; Y)$ with X_i strongly relevant, as shown in Example 2.2. One drawback of this method is that it does not return a complete ranking of the variables of the dataset. In [148], this approach is shown competitive with two filters [77][1]. Note that in [148], a normalized measure of mutual information called the symmetrical uncertainty (2.40) is used, i.e. $SU(X, Y) = \frac{2I(X;Y)}{H(X)+H(Y)}$. This measure helps to improve the performances of the selection by penalizing inputs with large entropies.

3.3.4 Backward elimination and Relevance Criterion

Let $X_S^{max} \subset X$ be the target subset, i.e., the subset X_S of size d, that achieves the maximal mutual information with the output (3.14). By the chain rule for mutual information (Theorem 2.8),

$$I(X; Y) = I(X_S^{max}; Y) + I(X_R^{max}; Y | X_S^{max}) \tag{3.17}$$

where $X_R = X_{-S}$ is the set difference between the original set of inputs X and the set of variables X_S selected so far.

The backward elimination (using mutual information) [114] starts with $X_S = X$ and, at each step, eliminates from the set of selected variable X_S, the variable X_j^{back} having the lowest relevance on Y,

$$X_j^{back} = \arg \min_{X_j \in X_S} I(X_j; Y | X_{S-j}) \tag{3.18}$$

In other words, X_j^{back} is an approximation of X_R^{max} in (3.17). The approximation is exact for a subset size $d = n-1$ of one variable less than the complete set. The elimination process is then repeated until the desired size is reached. However, this approach is intractable for large variable sets since the beginning of the procedure requires the estimation of a multivariate density that includes the whole set of variables X.

3.3.5 Markov Blanket Elimination

The Markov blanket elimination [76] consists in approximating $I(X_j; Y | X_{S-j})$ in (3.18) by $I(X_j; Y | X_{M_j})$ with $X_{M_j} \subset X_{S-j}$ a subset of variables having limited fixed size k. The algorithm proceeds in two phases. First, for every variable X_j in the selected set X_S, k variables X_{M_j} are selected among the variables X_{S-j}. Second, the least relevant variable X_j^{MB} (conditioned on the selected subset $X_{M_j} \subseteq X_{S-j}$) is eliminated, i.e., $X_S = X_S \backslash X_j^{MB}$.

$$X_j^{MB} = \arg \min_{X_j \in X_S} I(X_j; Y | X_{M_j}) \tag{3.19}$$

The process is repeated until the selected variable set X_S contains no more irrelevant and redundant variables, or when the desired subset size is reached. The method is named from the fact that X_{M_j} is an approximate Markov blanket (see Definition 3.4). Also, if X_{M_j} is a Markov blanket then no relevant variables are eliminated from the candidate set. In [76], the Pearson's correlation coefficient [71] is used in order to find the k variables most correlated to the candidate X_j. These k variables are considered as the Markov blanket X_{M_j} of the candidate X_j. In this way, only linear dependencies between variables are considered. However, more complex functions can make the algorithm very slow. In fact, finding a Markov blanket is itself a variable selection task. As a result, this method is not adapted to large dimensionality problems.

3.3.6 Forward Selection and Relevance Criterion (REL)

A way to sequentially maximize the quantity $I(X_S; Y)$ in (3.14), is provided by the chain rule for mutual information (Theorem 2.8):

$$I(X_{S'}; Y) = I(X_S; Y) + I(X_i; Y|X_S) \tag{3.20}$$

where $X_{S'} = X_{S,i}$ is the updated set of variables. Rather than maximizing the left-hand side term directly, the idea of the forward selection combined with the relevance criterion consists in maximizing sequentially the second term of the right-hand term, $I(X_i; Y|X_S)$. In other words, the approach consists in updating a set of selected variables X_S with the variable X_i^{REL} featuring the maximum relevance (Section 3.1.3).

In analytical terms, the variable X_i^{REL} returned by the relevance criterion at each step is,

$$X_i^{REL} = \arg \max_{X_i \in X_R} \{I(X_i; Y|X_S)\} \tag{3.21}$$

where $X_R = X_{-S}$ is the difference between the original set of inputs X and the set of variables X_S selected so far. This strategy prevents from selecting a variable which, though relevant to Y, is redundant with respect to a previously selected one. This algorithm has been used in [8, 5, 13, 114]. In [8], the normalized version of relevance (2.43) is used.

Although this method is appealing, it presents some major drawbacks. The estimation of the relevance requires the estimation of large multivariate densities. For instance, at the d-th step of the forward search, the search algorithm requires $n - d$ evaluations, where each evaluation requires in turn the computation of a $(d + 1)$-variate density. It is known that for a large d, the estimations are poorly accurate and/or computationally expensive [104]. In particular in the small sample settings (around one hundred), having an accurate estimation of large $(d > 3)$ multivariate densities is difficult. For these reasons, the recent filter literature adopt selection criteria based on bi- and trivariate densities at most.

3.3.7 Forward Selection and Conditional Mutual Information Maximization criterion (CMIM)

The CMIM approach [49] proposes to select the variable $X_i \in X_R$ whose minimal relevance $I(X_i; Y|X_j)$ conditioned to each selected variable taken separately $X_j \in X_S$, is maximal. This requires the computation of the mutual information of X_i and the output Y, conditioned on each variable $X_j \in X_S$ previously selected.

Formally, the variable returned according to the CMIM is

$$X_i^{CMIM} = \arg \max_{X_i \in X_R} \{ \min_{X_j \in X_S} I(X_i; Y|X_j)\} \tag{3.22}$$

A variable X_i can be selected only if its information to the output Y has not been caught by an already selected variable X_j.

The CMIM criterion is an approximation of the relevance criterion,

$$X_i^{REL} = \arg \max_{X_i \in X_R} \{I(X_i; Y|X_S)\}$$

where $I(X_i; Y|X_S)$ is replaced by $\min_{X_j \in X_S}(I(X_i; Y|X_j))$.

[49] shows experiments where CMIM is competitive with FCBF [148] in selecting binary variables for a pattern recognition task. This criterion selects relevant variables, avoids redundancy, avoids estimating high dimensional multivariate densities and does not ignore complementarity two-by-two. However, it does not necessarily select a variable complementary to the already selected variables. Indeed, a variable that has a high negative interaction to the already selected variable will be characterized by a large conditional mutual information with that variable but not necessarily by a large minimal conditional information. In Examples 2.2 and 2.3, for instance, the complementary variable has a null relevance taken alone. In that case, $\min_{X_j \in X_S} I(X_i; Y|X_j) = 0$ and CMIM would not select that variable.

3.3.8 Forward Selection and Minimum Redundancy - Maximum Relevance criterion (MRMR)

The Minimum Redundancy-Maximum Relevance (MRMR) criterion has been proposed in [104, 131, 103] in combination with a forward selection search strategy. Given a set X_S of selected variables, the method updates X_S with the variable $X_i \in X_R$ that maximizes $v_i - z_i$, where v_i is a relevance term and z_i is a redundancy term. More precisely, v_i is the relevance of X_i to the output Y alone, and z_i is the average redundancy of X_i to each selected variables $X_j \in X_S$.

$$v_i = I(X_i; Y) \tag{3.23}$$

$$z_i = \frac{1}{|X_S|} \sum_{X_j \in X_S} I(X_i; X_j) \tag{3.24}$$

$$X_i^{MRMR} = \arg \max_{X_i \in X_R} \{v_i - z_i\} \tag{3.25}$$

At each step, this method selects the variable which has the best trade-off between

relevance and redundancy. This selection criterion is fast and efficient. At step d of the forward search, the search algorithm computes $n - d$ evaluations where each evaluation requires the estimation of $(d+1)$ bi-variate densities (one for each already selected variables plus one with the output). As a result, MRMR avoids the estimation of multivariate densities by using multiple bivariate densities.

A justification of MRMR given by the authors [104] is that

$$I(X;Y) = H(X) + H(Y) - H(X,Y) \tag{3.26}$$

with

$$R(X_1; X_2; ...; X_n) = \sum_{i=1}^{n} H(X_i) - H(X) \tag{3.27}$$

and

$$R(X_1; X_2; ...; X_n; Y) = \sum_{i=1}^{n} H(X_i) + H(Y) - H(X,Y) \tag{3.28}$$

hence

$$I(X;Y) = R(X_1; X_2; ...; X_n; Y) - R(X_1; X_2; ...; X_n) \tag{3.29}$$

where,

- the minimum of the second term $R(X_1; X_2; ...; X_n)$ is reached for independent variables since, in that case, $H(X) = \sum_i H(X_i)$ and $R(X_1; X_2; ...; X_n) = \sum_i H(X_i) - H(X) = 0$. Hence, if a subset of variables X_S is already selected, a variable X_i should have a minimal redundancy $I(X_i; X_S)$ with the subset. Pairwise independency does not guarantee independency. However, the authors approximate $I(X_i; X_S)$ with $\frac{1}{|S|} \sum_{j \in S} I(X_i; X_j)$.
- the maximum of the first term $R(X_1; X_2; ...; X_n; Y)$ is attained for maximally dependent variables.

Qualitatively, in a sequential setting where a selected subset X_S is given, independence between the variables in X is reached by minimizing $\frac{1}{|X_S|} \sum_{X_j \in X_S} I(X_i; X_j) \simeq I(X_i; X_S)$ and maximizing dependency between the variables of X and of Y, i.e., by maximizing $I(X_i; Y)$.

Although the method addresses the issue of bivariate redundancy through the term z_i, it does not capture complementarity between variables. This can be ineffective in situations like Example 2.2 where, although the set $\{X_i, X_S\}$ is very relevant, we have that

1. the redundancy term z_i is large due to the redundancy of X_i and X_S,

2. the relevance term v_i is small since X_i "alone" is not relevant to Y.

3.3.9 k-Additive Truncation of Mutual Information

In [75], it is proposed to use the k first terms, denoted by $I^{(k)}(X;Y)$, of (3.30) (Section 2.2.2):

$$I(X;Y) = \sum_{X_i \in X} I(X_i;Y) - \sum_{X_{i,j} \subset X} C(X_i;X_j;Y) + \sum_{X_{i,j,l} \subset X} ... + (-1)^{n+1} C(X_1;X_2;...;X_n;Y) \tag{3.30}$$

As a result, $I^{(k)}(X;Y)$ can deal with complementarity up to order k.

The second and third orders, $I^{(2)}(X;Y)$ and $I^{(3)}(X;Y)$ are of practical interest for selecting variables [75].

$$I^{(2)}(X;Y) = \sum_{X_i \in X} I(X_i;Y) - \sum_{X_{i,j} \subset X} C(X_i;X_j;Y) \tag{3.31}$$

$$= \sum_{X_{i,j} \subset X} I(X_{i,j};Y) - (\mid \mathcal{X} \mid -2) \sum_{X_i \subset X} I(X_i;Y) \tag{3.32}$$

$$I^{(3)}(X;Y) = \sum_{X_i \subset X} I(X_i;Y) - \sum_{X_{i,j} \subset X} C(X_i;X_j;Y) + \sum_{X_{i,j,l} \subset X} C(X_i;X_j;X_l;Y) \tag{3.33}$$

$$= \sum_{X_{i,j,l} \subset X} I(X_{i,j,l};Y) - (\mid \mathcal{X} \mid -3) \sum_{X_{i,j} \subset X} I(X_{i,j};Y) + \left[\begin{pmatrix} \mid \mathcal{X} \mid -1 \\ 2 \end{pmatrix} - \mid \mathcal{X} \mid +2 \right] \sum_{X_i \subset X} I(X_i;Y) \tag{3.34}$$

However, the number of required mutual informations to compute $I^{(2)}(X;Y)$ and $I^{(3)}(X;Y)$ is higher than for CMIM or MRMR. In [75], the method has reached successful results on five artificial regression problems. A similar selection criterion will be detailed in the contributions (Section 4.1).

3.4 Part II: Network Inference

In chapter 4, network inference is seen as an extension of variable selection. Before connecting information-theoretic methods of network inference and feature selection, we present a state-of-the-art of network inference methods. Network inference consists in representing the dependencies between the variables of a dataset by a graph [140]. The exact semantics of an arc in the graph may differ from one inference method to another. However, when network inference is applied to microarray data, arcs are usually meant to represent a regulator/regulated gene interaction where the genes are represented by nodes in the graph. The latter graph is called a transcriptional regulatory network [135].

Various network inference methods have been proposed in the literature such as,

- differential equation network where expression data are explained by systems of ordinary differential equations (ODEs) or stochastic ODEs [51].
- Boolean network where the state of a gene (on or off) is expressed as a Boolean function of the input genes [51].
- correlation and partial correlation network that sets an arc between two genes if it exhibits a high score based on correlation measures [119, 123].
- mutual information network that sets an arc between two genes if it exhibits a high score based on pairwise mutual information [46, 83].
- Bayesian network that represents the probabilistic dependencies between variables [102, 95, 128].

In the following sections, mutual information networks and Bayesian networks are investigated. This choice is partly motivated by our desiderata of dealing with non-linear (and probabilistic) dependencies.

First, we describe in generic terms the principles of (undirected) network inference.

Let us consider n (discrete) input variables $X = (X_1, X_2, ..., X_n)$ and let us assume that each variable X_j ($j \in A = \{1, 2, ..., n\}$) is a function of a causal subset $X_{S_j} \subset X$,

$$x_j = f_j(x_{S_j}) + \varepsilon,\ j \in A,\ S_j \subset A \tag{3.35}$$

where causal is used here in the sense (of *direct causality)* used in Bayesian network literature [102, 95, 128]. The following definition will be detailed in Section 3.6.1.

Definition 3.8: *[95] X_{S_j} is a direct cause of X_j if a manipulation of X_{S_j} changes the distribution of X_j and there is no other variable or set of variables X_W (that does not*

contain X_j or variables of X_{S_j}) such that once we know the value of X_W, a manipulation of X_{S_j} no longer changes the probability distribution of X_j.

In other words, we assume an implicit underlying network T of relationships between the variables, with elements

$$t_{ij} = \begin{cases} 1 & \text{if } X_i \in X_{S_j} \text{ or } X_j \in X_{S_i} \\ 0 & \text{else} \end{cases} \tag{3.36}$$

Hence, the objective of a network inference algorithm $\mathcal{N}$ consists in producing a network $\hat{T}$ from a dataset D_m,

$$\hat{T} = \mathcal{N}(D_m) \tag{3.37}$$

that is as "close" (using performance measure Φ) as possible to the unknown (or partially known) true network T.

$$\hat{T}^{max} = \arg\max_{\hat{T}} \Phi(\hat{T}, T) \tag{3.38}$$

Qualitatively, we consider the following chain,

$$T \underbrace{\longrightarrow}_{explains} X \underbrace{\longrightarrow}_{measurements} D_m \underbrace{\longrightarrow}_{inference} \hat{T} \tag{3.39}$$

where $\hat{T}$ is inferred from a dataset D_m that is generated according to the probability distribution $p(X)$ which is itself governed by T.

A network inference problem can be seen as a binary decision problem where the algorithm $\mathcal{N}$ plays the role of a classifier. Each pair of nodes $\hat{t}_{ij}$ is thus assigned a positive label (1, i.e., an edge) or a null one (0, i.e., no edge). A reference network T is then required to assess the performances of the inferred network $\hat{T}$. Two strategies are usually adopted to define a reference network:

1. The dataset is artificially generated from a known network which allows to compare performances of competing methods.

2. The inference concerns microarray measurements from a cell where a small amount of genetic interactions have already been discovered by researchers. This list of known interactions can then be used as a reference network. Since discovering (biologically) a new gene-gene interaction is expensive, network inference techniques can support the discovery process.

A positive label (an edge) predicted by the algorithm is considered as a true positive (tp) or as a false positive (fp) depending on the presence or not of the corresponding edge in

Inferred \Truth	True edge (1)	No edge (0)
Inferred edge (1)	tp= $\#(\hat{t}_{ij} = t_{ij} = 1)$	fp= $\#(\hat{t}_{ij} = 1 \neq t_{ij} = 0)$
Deleted edge (0)	fn= $\#(\hat{t}_{ij} = 0 \neq t_{ij} = 1)$	tn= $\#(\hat{t}_{ij} = t_{ij} = 0)$

Table 3.1: Confusion matrix CM

the reference network. Analogously, a null label is considered as a true negative (tn) or a false negative (fn) depending on whether the corresponding edge is absent or not in the underlying true network T, respectively (Table 3.1).

Many inference methods return a weighted adjacency matrix W of the network. Hence, a threshold value θ is used in order to delete the arcs of the network that have a too low score [51, 46, 18].

$$\hat{t}_{ij} = \begin{cases} 1 & \text{if } w_{ij} \geq \theta \\ 0 & \text{otherwise} \end{cases} \tag{3.40}$$

For each threshold value θ, a different inferred network $\hat{T}(\theta, W)$ coming from the weighted adjacency matrix W can be computed. As a result, a specific confusion matrix CM (Section 2.4.1.2) is obtained for each θ.

In the following sections, we introduce three validation tools, namely ROC curves, PR curves and F-scores, that will be used as performance measures Φ (3.38) of network inference techniques.

3.4.1 ROC curves

A Receiver Operating Characteristic (ROC) curve is a graphical plot of the *tpr* (true positive rate) vs. fpr (false positive rate) for a binary classifier system when the threshold is varied [32], where the false positive rate is

$$fpr = \frac{fp}{tn + fp} = \frac{fp}{\#(t_{ij} = 0)} \tag{3.41}$$

and the true positive rate is

$$tpr = \frac{tp}{tp + fn} = \frac{tp}{\#(t_{ij} = 1)} \tag{3.42}$$

The *tpr* is also known as *recall* or *sensitivity*. The objective is to maximize *tpr* while minimizing fpr.

By setting the threshold θ to its extreme values, it is possible to minimize the number of false negative or false positive. In network inference, the threshold applied to the weighted adjacency matrix plays that role. If θ is set to its minimum value, no arc is eliminated

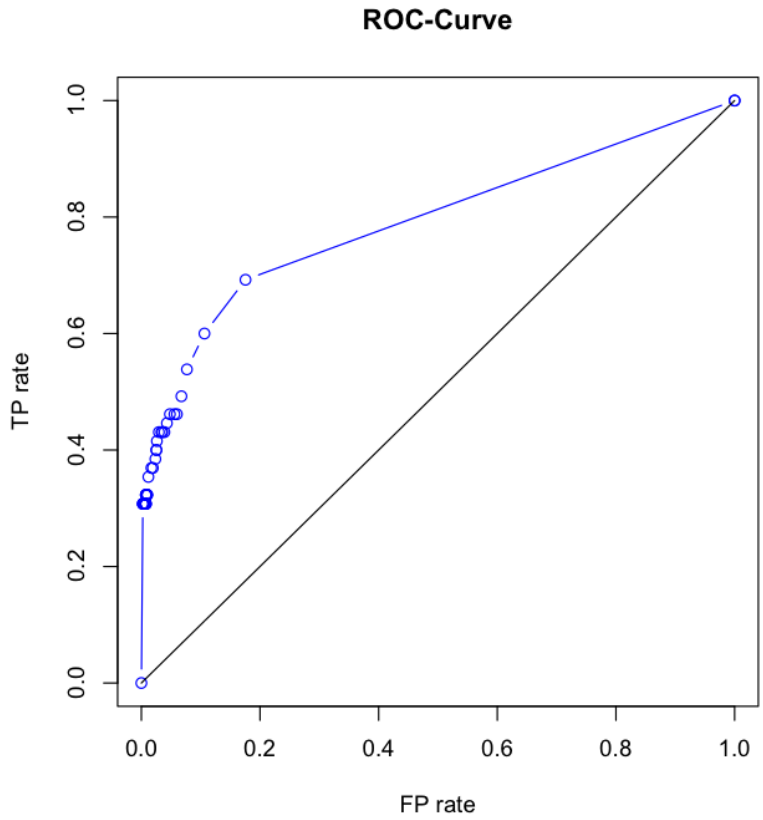


Figure 3.3: Example of ROC curves (generated with the R package 'minet' [89])

from the network. Hence, the tpr and fpr are both maximal ($tn = fn = 0$, at point $(1, 1)$ on Figure 3.3). On the other side, if θ is maximal, all the arcs are eliminated, resulting in a minimal tpr and a minimal fpr ($tp = fp = 0$, at point $(0, 0)$). A perfect classifier would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing a 100% tpr (all true positives are detected) and a 0% fpr (no false positives are found). A random guess gives a point along the diagonal line (the so-called line of no-discrimination) that goes from the left bottom to the top right corners (see Figure 3.3). Points above the diagonal line indicate good classification results, while points below the line indicate wrong results.

It is recommended [109] to use receiver operator characteristic (ROC) curves when evaluating binary decision problems in order to avoid effects related to the chosen threshold θ. However, ROC curves can present an overly optimistic view of an algorithm's performance if there is a large skew in the class distribution, as typically encountered in transcriptional network inference because of sparseness. To tackle this problem, precision-recall (PR) curves have been proposed as an alternative to ROC curves [12].

3.4.2 PR curves

The *precision* quantity is given by

$$pre = \frac{tp}{tp + fp} = \frac{tp}{\#(\hat{t}_{ij} = 1)} \tag{3.43}$$

It measures the fraction of real edges among the ones classified as positive.

The *recall* quantity (also called true positive rate (tpr)) is given by

$$rec = tpr = \frac{tp}{tp + fn} = \frac{tp}{\#(t_{ij} = 1)} \tag{3.44}$$

It denotes the fraction of real edges that are correctly inferred. The objective of an inference method is to maximize both precision and recall. These quantities depend on the threshold chosen.

The PR curve is a diagram which plots precision (pre) against recall (rec) for different values of the threshold [126](see Figure 3.4 for a PR curve corresponding to the ROC curve in Figure 3.3). In a network inference setting, this diagram illustrates the trade-off between eliminating many arcs (low recall due to high threshold) with confidence on the remaining arcs (high precision) and keeping many arcs (high recall) with doubt on their significance (low precision).

3.4.3 F-Scores

The F-score is a weighted harmonic average of precision and recall [126]:

$$F_\beta(\hat{T},\, T) = \frac{(1 + \beta)(pre)(rec)}{\beta pre + rec} \tag{3.45}$$

where β is a non-negative real parameter denoting the weight of the recall w.r.t. precision. This quantity lies between 0 and 1.

The three commonly used F-scores are $\beta = 1$,

$$F(\hat{T},\, T) = \frac{2(pre)(rec)}{pre + rec} \tag{3.46}$$

$\beta = 2$ (the F_2-measure), which weighs recall twice as much as precision, and $\beta = 0.5$ (the $F_{0.5}$-measure), which weighs precision twice as much as recall.

A compact representation of the PR diagram can be returned by the maximum and/or the average (avg) F_β-score.

$$\begin{aligned} F_\beta^{max}(W,\ T) &= \max_\theta F_\beta(\hat{T}(\theta, W),\, T) \\ F_\beta^{avg}(W,\ T) &= avg_\theta[F_\beta(\hat{T}(\theta, W),\, T)] \end{aligned} \tag{3.47}$$

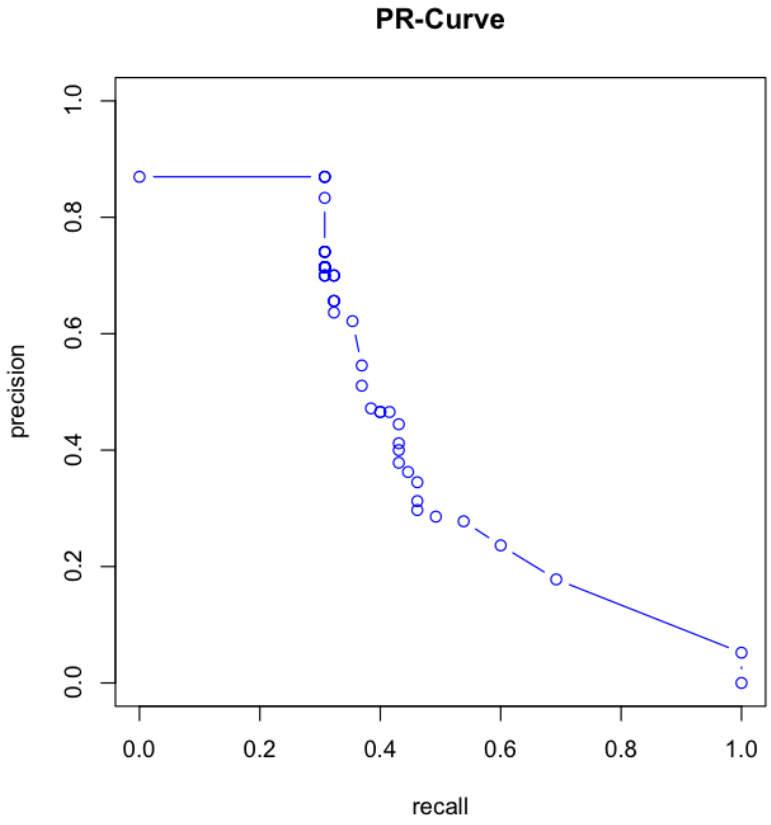


Figure 3.4: Example of PR curves (generated with the R package 'minet' [89])

where θ is the threshold parameter.

These measures are used, in Chapter 4, as performance measure Φ of undirected network inference methods (3.38).

3.5 Mutual Information Networks

Mutual information networks are a subcategory of inference methods. In these methods, a link between two nodes is set if it exhibits a high score based on pairwise mutual information. The adoption of mutual information in the network inference task traces back to Chow and Liu's tree algorithm [23]. In Bayesian networks, mutual information has been used by [101, 22, 80, 149] and later [120, 96] suggested the use of multiinformation (Definition 2.11). In this section, a current state-of-the-art of network inference methods, based on pairwise mutual information, is formulated.

Mutual information networks require the computation of the mutual information matrix (MIM), a square matrix whose mim_{ij} element is given by,

$$\text{mim}_{ij} = I(X_i; X_j) \tag{3.48}$$

It is the mutual information between X_i and X_j, where X_i is a discrete random variable denoting the expression level of the ith gene in a transcriptional regulatory network inference. An advantage of these methods lies in their affordable computational complexity. This results from the fact that $\frac{n(n-1)}{2}$ calls of mutual information, based on bivariate probability distributions, are required to compute the MIM. Since each estimation of a bivariate distribution can be computed quickly and does not require a large amount of samples, these methods are particularly adapted to dealing with microarray data. Mutual information is a symmetric measure, hence it is not possible to derive the direction of an edge. This limitation is common to all the following methods. However, this information could be provided by edge orientation algorithms commonly used in Bayesian networks (Section 3.6).

3.5.1 Chow-Liu Tree

The Chow and Liu approach consists in finding the maximum spanning tree on the complete graph whose edge weights are the mutual information between two nodes [23].

In graph theory, a tree is a graph in which any two vertexes are connected by exactly one path. A spanning tree is a tree that connects all the vertexes of the graph. The maximum spanning tree is the spanning tree with the sum of edge weights more than or equal to that of every other spanning tree.

A maximum spanning tree can be done in $O(n^2 \log n)$ using, for example, Kruskal's algorithm [94]. The drawback of this method lies in the fact that the resulting network has a low number of edges. Precision and recall cannot be studied as a function of the parameter. Chow-Liu tree is proved [83] to be a subnetwork of the network reconstructed by the ARACNE algorithm described in Section 3.5.4.

3.5.2 Relevance Network (RELNET)

The relevance network approach [19] has been introduced for gene clustering and successfully applied to infer relationships between RNA expression and chemotherapeutic susceptibility [18]. It consists in inferring a network in which a pair of genes $\{X_i, X_j\}$ is linked by an edge if the mutual information $I(X_i; X_j)$ is larger than a given threshold θ. The complexity of the method is $O(n^2)$ since all pairwise interactions are considered.
This method does not eliminate all indirect interactions between genes. For example, if gene X_1 regulates both gene X_2 and gene X_3, this would cause a high mutual information between the pairs $\{X_1,X_2\}$, $\{X_1,X_3\}$ and $\{X_2,X_3\}$. As a consequence, the algorithm will set an edge between X_2 and X_3 although these two genes interact only through gene X_1.

Combining RELNET with the Gaussian estimator of mutual information (Section 2.6) boils down to building a correlation network [70].

3.5.3 CLR Algorithm

The CLR algorithm [46] is an extension of the RELNET algorithm. This algorithm derives a score from the empirical distribution of the mutual information for each pair of genes. In particular, instead of considering the information $I(X_i; X_j)$ between genes X_i and X_j, it takes into account the score $w_{ij} = \sqrt{z_i^2 + z_j^2}$ where

$$z_i = \max\left(0, \frac{I(X_i; X_j) - \mu_i}{\sigma_i}\right) \tag{3.49}$$

where μ_i and σ_i are respectively the mean and standard deviation of the empirical distribution of the mutual information values $I(X_i, X_k)$, $k = 1, \ldots, n$. The pseudo code of CLR is given in Algorithm 3.4. The CLR algorithm was successfully applied to decipher the *E. Coli* transcriptional regulatory network [46]. CLR has a complexity in $O(n^2)$ once the MIM is computed.

3.5.4 ARACNE

The Algorithm for the Reconstruction of Accurate Cellular Networks (ARACNE) [83] is based on the Data Processing Inequality (Theorem 2.9) stating that, if gene X_1 interacts

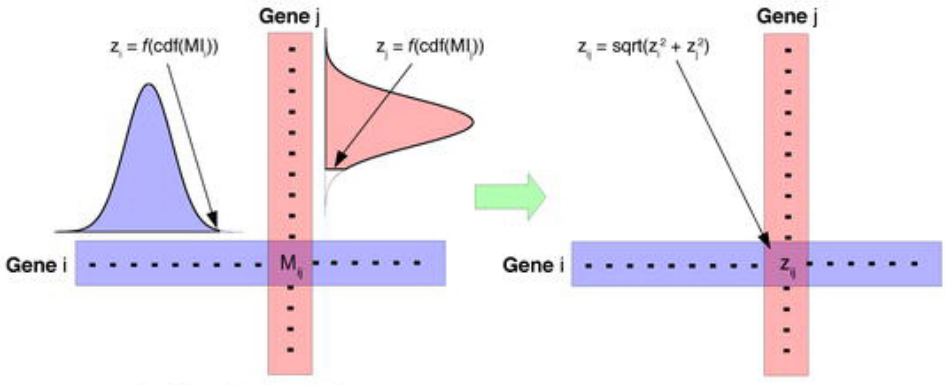


Figure 3.5: Figure from [46] illustrating the principle of CLR.

Algorithm 3.4 pseudo code of the normal version of CLR algorithm

Inputs: $I(X_i; X_j),\ \forall i, j \in A = \{1, 2, ..., n\}$
for all inputs X_i in the input space X
$\mu_i \leftarrow mean(I(X_i; X_j), j \in \{1, 2, ..., n\})$
$\sigma_i \leftarrow variance(I(X_i; X_j),\ j \in \{1, 2, ..., n\})$
end-for
for each pair of variables $X_{i,j}$ in the input space X
$w_{ij} \leftarrow \max\left(0, \frac{1}{\sqrt{2}} * \left\{\frac{I(X_i;X_j)-\mu_i}{\sigma_i} + \frac{I(X_i;X_j)-\mu_j}{\sigma_j}\right\}\right)$
end-for
Output: the weighted adjacency matrix W (having elements w_{ij})

Algorithm 3.5 pseudo code of ARACNE algorithm

Inputs: the MIM, i.e., $I(X_i; X_j), \forall i, j \in \{1, 2, ..., n\}$
for each pair of variables $X_{i,j}$ in the input space X
  for each variable X_k in the space $X_{-(i,j)}$
    if$(I(X_i; X_j) < I(X_i; X_k) \quad and \quad I(X_i; X_j) < I(X_j; X_k))$
      $w_{ij} \leftarrow 0$
    else
      $w_{ij} \leftarrow I(X_i; X_j)$
  end-for
end-for
Output: the weighted adjacency matrix W (having w_{ij} as elements)

with gene X_3 through gene X_2, then

$$I(X_1; X_3) \leq \min\left(I(X_1; X_2), I(X_2; X_3)\right)$$

ARACNE begins by assigning to each pair of nodes a weight equal to their mutual information. Then, as in RELNET, all edges for which $I(X_i; X_j) < \theta$ are removed, with θ a given threshold. Eventually, the weakest edge of each triplet is interpreted as an indirect interaction and is removed (see pseudo code Algorithm 3.5).

An extension of ARACNE removes the weakest edge only if the difference between the two lowest weights lies above a threshold η. Hence, increasing θ lowers the number of inferred edges while the opposite happens when increasing η.

If the network is a tree including only pairwise interactions, the method guarantees the reconstruction of the original network, once it is provided with the exact MIM (see [83]). ARACNE's complexity is $O(n^3)$ since the algorithm considers all triplets of genes. In [83] the method has been able to recover components of the transcriptional regulatory network in mammalian cells and has outperformed Bayesian networks and relevance networks on several inference tasks [83].

3.6 Bayesian networks and information theory

Bayesian network theory connects causality and probability theory. This connection will give us a better understanding of mutual information networks and of complementary variables. Therefore, this link is discussed here. However, most Bayesian network inference algorithms (such as IC [102]) are too computationally costly for dealing with thousands of variables such as microarray datasets and, for this reason, are not treated in this thesis.

A Bayesian network is a combination of a directed acyclic graph (DAG) G and a joint probability distribution p. Bayesian networks use the graph G in order to encode the

joint probability distribution p of a set of n variables. In particular, the DAG G can be interpreted by saying that each variable X_i is independent, given its parents (variables with arrows to X_i), of its non-descendants (variables to which there is no directed path from X_i) [95].

$$p(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} p(X_i|parents(X_i)) \tag{3.50}$$

When a DAG G and a probability distribution p satisfy (3.50), the graph G is said to satisfy the *Markov condition,* and is called a *Bayesian network*.

A word of caution is necessary here about the capacity of Bayesian networks to model loops. Although cycles appear to have good possibilities for modelling feed-back and feed-forward effects in transcriptional networks, there is no suitable joint probability to model these situations. A distribution such as $p(A|B)p(B|C)p(C|A)$ is not a well defined probability distribution, apart from very special cases [140]. Furthermore, the notion of loop is related to dynamics whereas we are modelling from i.i.d. samples (with no temporal dependencies). Therefore, cycles are not allowed in Bayesian networks.

DAGs are imperfect representations of joint probability distributions. Indeed, not all sets of conditional independence relations that might be satisfied by a probability distribution can be represented by these graphical models and two or more graphical models can often represent the same conditional independence relations [123]. Hence, an additional condition is required in order for G and p to match with (3.50):

Definition 3.9: *[95] A DAG G and a probability distribution p satisfy the faithfulness condition if the Markov condition entails all and only the conditional independencies in p.*

A lot of definitions, properties and discussions have been made in order to assess or assume that a graph G and a probability distribution p are faithful to each other, some of these are discussed in Section 3.6.1. More information on that topic, can be found in [102, 95, 128].

An interesting property of faithful Bayesian networks is the following: *the minimum set that achieves optimal classification accuracy is the Markov blanket of a target variable* [132]. In a Bayesian network, the Markov blanket of a target variable is composed of the *variable's parents, the variable's children, and the variable's children's parents* [101]. As a consequence, the optimal set of variables to predict a target is directly identified in a Bayesian network. In the next chapter, experiments that rely on this property are conducted.

3.6.1 Causality

Bayesian networks not only offer a compressed view of a joint probability distribution, but can bring a causal interpretation of the underlying process generating the distribution. Let us stress this interesting feature, in information-theoretic terms, since it is related to variable selection concepts such as variable complementarity.

Definition 3.10: *[95] X is a* cause *of Y if there exists a value $x \in \mathcal{X}$ such that setting $X = x$ leads to a change in the probability distribution of Y.*

The difference between seeing and acting is very important in the causal model literature. [102] introduces the $do(\cdot)$ operator in order to formally make the difference.

Example 3.2: *The probability that the pavement gets wet when we observe that the sprinkler is on is the same than the probability that the pavement gets wet when we set the sprinkler on $p(pavement = wet|sprinkler = on) = p(pavement = wet|do(sprinkler = on))$. However, the probability to observe that the sprinkler is on given that we observe the pavement wet is different than this probability when we set the pavement wet (by other means than the sprinkler). When the pavement is wet, the probability that the sprinkler was on, is increased, $p(sprinkler = on|pavement = wet) > p(sprinkler = on)$. However, setting the pavement wet does not increase the probability that the sprinkler was on. $p(sprinkler = on|do(pavement = wet)) = p(sprinkler = on)$. An asymmetry appears when using the $do(\cdot)$ operator. This is because the $do(\cdot)$ operator does not only set the value of the conditioning variable to $X = x$, but also renders the conditioning variable X independent of all its causes [102].*

In the following, we denote "X causes Y" using an arrow from X to Y, i.e., $X \rightarrow Y$. A *causal link* (undirected) between X and Y, i.e., $X \rightarrow Y$ or/and $X \leftarrow Y$, is denoted by $X \leftrightarrow Y$.

The definition of causality states that a causal relation between two variables creates a stochastic dependency between the probability distributions of causes and effects. Taking mutual information as dependency measure, we obtain the following assertion:

$$X_i \leftrightarrow Y \Rightarrow I(X_i; Y) > 0 \tag{3.51}$$

However, (3.51) is not always true. A known example is the cancellation of two causal pathways [95]. In most cases, a causality relationship implies a stochastic dependency between the variables. As a result, in the following, we assume that *causality implies dependency*.

However, the converse is not true, dependency does not imply causality,

$$X_i \leftrightarrow Y \not\Leftarrow I(X_i; Y) > 0$$

An objection to *dependency implies causality* relies on the common-cause effect. Indeed, there is another possibility that can create a dependency between two variables: having a common cause, i.e., $X_i \leftarrow X_k \rightarrow X_j$. Two variables that have a common cause can be dependent while manipulating one of them does not influence the other.

This is where the direct causality definition can help us. Let us repeat Definition 3.8 here,

X is a direct cause of Y if X is a cause of Y and there is no other variable W such that once we know the value of W, a manipulation of X no longer changes the probability distribution of Y.

This definition states that two dependent variables are no longer dependent once given the direct cause. Or analogously, if there are no set of variables that cancel the dependency between two other variables, then one of the variables is a direct cause of the other. This remains true for the common-cause case since the dependency between the two consequences should be cancelled given the true cause. In other words, we can state that if *two variables are dependent in every context*, then these variables have a causal relationship. Formulated in information-theoretic terms [22] it gives the following:

$$\forall X_S \subset X_{-(i,j)} : \; I(X_i; X_j | X_S) > 0 \Rightarrow X_i \leftrightarrow X_j \tag{3.52}$$

There is an implicit strong assumption behind (3.52): no missing causes, that is all the variables that cause at least two effects (two variables in the dataset) should also be present in the dataset. Indeed, if there is a common cause to two observable effects, the two effects are dependent in every context except once we condition on the common cause. If the common cause is hidden then (3.52) can lead to false causality relationships.

Bayesian networks assume *causal sufficiency*, that is, *there is no hidden common cause (to two effects)* in the dataset, i.e.,

$$\forall (X_i, X_j) \in X, \; \nexists (X_k \notin X) : \; X_i \leftarrow X_k \rightarrow X_j \tag{3.53}$$

It should be noted that we can usually conceive intermediate unidentified variables along each edge *in the causal direction* $X_i \rightarrow X_k \rightarrow X_j$, see the following example from [128].

Example 3.3: *If a is the event of striking a match and c is the event of the match catching on fire and no other events are considered, then A causes C. However, if we add b which*

stands for the sulfur on the match tip that achieved sufficient heat to combine with oxygen, then A no longer causes C directly. Rather A causes B and B causes C, i.e., $A \rightarrow B \rightarrow C$.

When we consider an arc between a variable and its grand parent, the arc remains causal since acting on the grand parent should change the distribution of the descendant variables. However, if we miss a common parent and set a link between a variable and its brother, then the link inferred with our rules is no longer causal. Indeed acting on the brother does not change the distribution of the target variable. In the latter case, the semantic of the link is lost. There are methods that try to remove the causal sufficiency assumption by detecting some hidden variables but this goes beyond the scope of this work; for more information on that topic, see for example [45].

At this stage, we do not know how to infer the directionality of a causality relationship. If we would have some temporal information such as event x happens before event y, then we could draw an arc from X to Y (assuming causality follows the arrow of time). If we could manipulate the variables, we would see that the dependency between a cause and a consequence remains when acting on the cause whereas it disappears when acting on the consequence. In other words, acting on a consequence renders the consequence independent from its cause(s), as seen in Example 3.2.

So the question that arises at this point is: *can we infer the direction of an arc just by making observations*? Surprisingly the answer to that question is positive. This can be done in some cases thanks to the explaining away effect.

Definition 3.11: *[The explaining away effect] [95] Once the value of a common effect is given, it creates a dependency between its causes because each cause explains away the occurrence of the effect, thereby making the other cause less likely.*

This is a common mechanism used by medical doctors when doing their diagnoses.

Example 3.4: *Some cancer can cause headache but a lot of more probable diseases, such as a cold, can also cause headache. Once a doctor has evidence that headaches are caused by a cold, he stops searching for a cancer although having cold and having a cancer are two independent events (see Figure 3.6).*

Note that the explaining away effect is characterized by a negative interaction information [67]. Hence, in information-theoretic terms [22] it can be expressed as follows:

$$X_i \rightarrow Y \leftarrow X_j \Rightarrow I(X_i; X_j|Y) > I(X_i; X_j) \tag{3.54}$$

where $I(X_i; X_j|Y) > I(X_i; X_j) \Leftrightarrow I(X_i; X_j|Y) - I(X_i; X_j) > 0 \Leftrightarrow C(X_i; X_j; Y) < 0$. Hence, the variables are complementary (see Section 3.1.5)

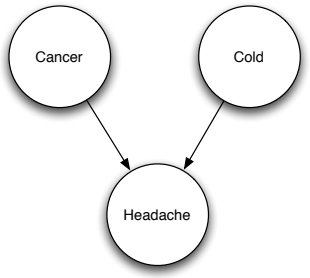


Figure 3.6: Illustration of the explaining away effect

We have seen in Section 2.2.2 that interaction information is symmetric for the three variables,

$$C(X_i;X_j;Y) = I(X_i;X_j) - I(X_i;X_j|Y) = I(X_i;Y) - I(X_i;Y|X_j) = I(X_j;Y) - I(X_j;Y|X_i) \tag{3.55}$$

As a consequence, the reversal statement of (3.54) is not true,

$$X_i \rightarrow Y \leftarrow X_j \not\Leftarrow I(X_i;X_j|Y) > I(X_i;X_j)$$

However, given undirected causal links, we can orient them thanks to the interaction information. This can be done in two ways:

First, if we have one variable Y, connected with two others, X_i and X_j, and there is a complementarity between them, then the variable Y is a common effect of the two other variables.

$$\left.\begin{array}{c} X_i \leftrightarrow Y \leftrightarrow X_j \\ I(X_i;X_j|Y) > I(X_i;X_j) \end{array}\right\} \Rightarrow X_i \rightarrow Y \leftarrow X_j \tag{3.56}$$

Second, if we have one variable Y that is a consequence of another variable X_i while also connected to a third variable X_j and that there is no complementarity between them, then the third variable is not another cause of the variable Y, hence it is a consequence.

$$\left.\begin{array}{c} X_i \rightarrow Y \leftrightarrow X_j \\ I(X_i;X_j|Y) \leq I(X_i;X_j) \end{array}\right\} \Rightarrow X_i \rightarrow Y \rightarrow X_j \tag{3.57}$$

Note that these two last rules are an information-theoretic translation of the edge orientation algorithm used in [102, 128]. Although edge orientation algorithms are not used in this work, we have identified how complementary variables can be created artificially.

3.7 Conclusion

With the increasing number of variables, exploration strategies of variable selection algorithms have moved from an exhaustive search [1] to stochastic and sequential searches [11, 74, 20]. The evaluation functions have also moved from wrapper approaches using a learning algorithm as evaluation function [11, 74, 20] to the filter approach using simpler evaluation functions such as mutual information [148, 49]. In a small sample setting, approximations of mutual information based on bi- and trivariate densities are often preferred to the multivariate estimation of mutual information.

Network inference allows to model dependencies between variables using a graph. An inference algorithm fed with microarray data returns a graph that can be interpreted as a transcriptional regulatory network. Mutual information networks rely on the computation of a matrix of pairwise mutual informations. Since bivariate mutual information are fast-to-compute, these techniques allow the inference of large networks, such as complete transcriptional regulatory networks.

Bayesian networks infer directed acyclic graphs. While computationally too expensive, these networks link causality, information theory and complementarity. This feature is used in experiments depicted in the next chapter.

Information-theoretic variable selection and information-theoretic network inference appear, at first sight, as two separate fields. However, the next chapter proposes a new method that connects them.

Chapter 4

Original Contributions

This chapter contains the main original contributions of the thesis:

1. A new criterion for variable selection (kASSI, Section 4.1) and its implementations in
   (a) a new forward selection algorithm [87](PREEST, Section 4.2) and its validation (in Section 4.2.1).
   (b) a new subset evaluation function [86] (DISR, Section 4.3) that is combined with an effective implementation of the backward elimination [90] (MASSIVE, Section 4.4). The experimental comparison between our new method and the state-of-the-art methods is given in Section 4.5.1.
2. A new network inference method called MRNET (Section 4.6) that has lead to
   (a) a new open-source R and Bioconductor package of network inference [89] (Section 4.7).
   (b) experimental studies on the impact of the intensity of noise, the number of samples, the number of variables, the estimators on information-theoretic network inference [88, 99] (Section 4.8).

4.1 The k-Average Sub-Subset Information criterion (kASSI)

This section presents an original criterion for variable selection [87, 90] that maximizes the mutual information with the output without requiring the estimation of a large multivariate density. The variable selection criterion is,

$$X_S^{kASSI} = \arg\max_{X_S \subseteq X} \{ \sum_{X_V \subseteq X_S : |V| = k} I(X_V; Y) \} \tag{4.1}$$

where X_V denotes subsets of variables of X_S (which is itself a subset of X). The rationale behind the k-Average Sub-Subset Information (kASSI) criterion is that an approximation of the information of a set of variables is related to the sum of the information of its subsets. For instance, to estimate the mutual information $I(X_{1,2,3,4};Y)$ between the set of variables $\{X_1, X_2, X_3, X_4\}$ and the output Y, the criterion uses the sum of the mutual informations of all the trivariate subsets: $I(X_{1,2,3};Y)$, $I(X_{2,3,4};Y)$, $I(X_{1,3,4};Y)$ and $I(X_{1,2,4};Y)$ (bivariate subsets can also be used).

A theoretical justification for the criterion can be found in the following theorem stating that the sum maximized in (4.1) appears in the lower bound of the quantity $I(X_S;Y)$.

Theorem 4.1: *[Lower bound on mutual information]*

$$\frac{1}{\left(\begin{array}{c} d \\ k \end{array}\right)} \sum_{X_V \subseteq X_S : |V|=k} I(X_V;Y) \leq I(X_S;Y) \tag{4.2}$$

PROOF: Since mutual information can only increase by adding variables (Section 2.1), we have

$$\max_i I(X_{S-i};Y) \leq I(X_S;Y) \tag{4.3}$$

By definition of the average, one has:

$$\frac{1}{d}\sum_{i=1}^{d} I(X_{S-i};Y) \leq \max_i I(X_{S-i};Y) \leq I(X_S;Y) \tag{4.4}$$

■

Applying Theorem 4.1 recursively, we obtain

$$\frac{1}{d}\sum_{i\in S} I(X_{S-i};Y) \geq \frac{1}{d(d-1)} \sum_{i\in S}\sum_{j\in S-i} I(X_{S-(i,j)};Y)$$

$$\Rightarrow I(X_S;Y) \geq \frac{1}{\left(\begin{array}{c} d \\ k \end{array}\right)} \sum_{X_V \subseteq X_S : |V|=k} I(X_V;Y)$$

Another interpretation of the selection criterion (4.1) is provided by the *averaging estimators* theory [60]. Suppose we aim to estimate the information per variable of a subset X_S, i.e., the quantity $Iv(X_S;Y) = \frac{I(X_S;Y)}{d}$. In this case, the kASSI criterion boils down to an estimator $\hat{Iv}$ of Iv which is the average of estimates of the target quantity

computed for all the subsets of S of size k , i.e

$$\hat{Iv} = \frac{1}{\left(\begin{array}{c} d \\ k \end{array} \right)} \sum_{V \subseteq S : |V| = k} \frac{I(X_V; Y)}{k} \tag{4.5}$$

The criterion (4.1) can be interpreted as an averaging approach to the estimation of the unknown quantity Iv which relies on the average of existing estimates of lower order.

Because of the combinatorial nature of the kASSI criterion, two particular values of k are computationally interesting: the case $k = d - 1$ and the case $k = 2$.

The kASSI criterion is similar to the k-additive truncation of mutual information of [75] (Section 3.3.9). We have developped the kASSI [87] without knowing [75].

Although both approaches are similar, the kASSI requires a smaller amount of computation of mutual information than the k-additive truncation.

4.2 The case $k = d - 1$: PREEST

The theoretical results described in the previous section have been implemented in a new search algorithm for variable selection, hereafter denoted by PREEST. In this method the $(d-1)$ASSI becomes the estimator of the mutual information of a given subset.

The PREEST algorithm [87] is a variant of the forward selection (Section 3.2.1) combined with the relevance criterion (Section 3.3.6). It relies on two different ways of assessing the relevance of a subset of variables:

1. a classical estimation of the mutual information of $I(X_S; Y)$ using the empirical entropy estimator and,

2. a *pre-estimation* of $I(X_S; Y)$ returned by the $(d-1)$ASSI.

More precisely, we define the preestimation measure of a subset of variables as,

$$PREEST(X_S) = \sum_{i \in S} I(X_{S-i}; Y) \tag{4.6}$$

The pre-estimation computation is fast since it only requires the computation of a sum of $d - 1$ mutual informations (that have been computed in the previous step of the forward selection). It follows that for a set of size d the pre-estimation has a computational complexity of order $O(d)$ whereas the evaluation of mutual information demands a cost of order $O(m \times d)$ with m the number of training samples.

The rationale for the PREEST algorithm (see detailed pseudo-code in Alg. 4.1) is that the evaluation assessment of a subset is carried out only if its pre-estimated value is

sufficiently high. Two parameters are required: the number of the p best subsets returned by the previous step, which has to be considered for further exploration and the number e of the most promising subsets to be evaluated among the pre-estimated ones.

At each step, the algorithm first selects the p best subsets assessed so far and obtains $p \times (n-d)$ new candidates by combining the p best subsets with the $n-d$ remaining input variables. The $p \times (n-d)$ sets are then pre-estimated by computing the kASSI. Eventually, the e best subsets according to the pre-estimation assessments are evaluated by directly computing the mutual information.

It follows that the algorithm carries out $p \times (n-d)$ pre-estimations and e evaluations at each step. As a result, the complexity of each step has order $O(\max(p \times n \times d,\ e \times m \times d))$. In such situation, a conventional greedy strategy using mutual information would require evaluating about n subsets per step, thus featuring a $O(n \times m \times d)$ complexity per step.

Choosing $p = m$ and $e = n$, we can shape the algorithm to have the same complexity order as a classical greedy search (e.g. forward selection, Section 3.2.1). However, in this case, our algorithm, besides evaluating $e = n$ subsets, has the advantage of pre-estimating, for an equal computational complexity, m times more subsets than the forward search. It follows that the region of the variable space which is explored by the PREEST algorithm is larger than the one considered by the greedy search.

A drawback of the algorithm resides in how to pre-estimate a variable set when its subsets of size $d-1$ have not been previously evaluated. This is expected to be a rare event for small d but becomes exponentially more frequent for large d. Our solution is to assign, by default, to these subsets the value of the worst evaluation of the preceding step. This amounts to assume that the sets which have not been evaluated are presumably non relevant subsets. This leads to a progressive bias of the pre-estimation assessment, which for increasing d, tends to weigh more the best subsets of the previous evaluation. Consequently, the method starts by exploring larger regions than the forward search for small d but ends up converging to a classical forward search for increasing d.

A second version of the method consists in setting $e = \frac{n}{d}$, $p = \frac{m}{d}$ in order to obtain a computational cost equal to $O(n \times m)$.

4.2.1 Experiments on PREEST

The experimental validation of PREEST was done in collaboration with Olivier Caelen (see [87]). We carried out an experimental session based on seven datasets provided by the UCI ML repository. The name of the datasets together with the number of variables and samples are reported in Table 4.1. Each dataset is divided into a training set, used to perform the variable selection, and an equal-sized test set. The continuous attributes are discretized by first partitioning their domain into seven equal-sized intervals, and then

Algorithm 4.1 Pseudo-code of the PREEST search algorithm

Inputs: the input variables X, the output variable Y, a maximal subset size d, two numbers e and p
L_{ES} is a list of pairs: teach variable of X together with its mutual information $I(X_i;Y)$
$X_S \leftarrow \phi$
while $(|X_S| < d)$
  for all the p best subsets X_{S_j} in the list of subsets L_{ES}
    for all the inputs X_i
      $score \leftarrow PREEST(X_{S_j,i})$
      if ($score$ is among the e best scores)
        add the subset $X_{S_j,i}$ in the list L_{PS} of the e best preestimations
    end-for
  end-for
  for all the e best subsets X_{S_j} in the list of preestimated subsets L_{PS}
    $score \leftarrow I(X_{S_j};Y)$
    if ($score$ is among the p best scores)
      add $(X_{S_j}, score)$ in the list L_{ES} of the p best estimations
    end-if
  end-for
end-while
Output: the best subset X_S of the list L_{ES}

associating a different class to each interval (see Section 2.7).

We compare the accuracy of the classical forward search and of two versions of the PREEST algorithm. The two versions are obtained by setting two pairs of values for the parameters p and e in order to shape properly their computational cost. In the first version (PREEST1), we set $e = n$ and $p = m$ in order to impose the same complexity $O(n \times m \times d)$ as the forward search. In the second version (PREEST2), we set $e = \frac{n}{d}$, $p = \frac{m}{d}$ in order to obtain a complexity equal to $O(n \times m)$.

The accuracy of the three algorithms is assessed by performing ten selection tasks for $d = 1, \ldots, 10$ and measuring the mutual information (the higher, the better) in the test sets. The value of the normalized mutual information $0 \leq \frac{I(S;Y)}{H(Y)} \leq 1$ averaged over the ten test sets for the three algorithms is reported in Table 4.1. We have chosen this quantity instead of the usual classification accuracy in order to avoid the bias deriving from the adoption of a specific learning machine.

The column PREEST1 in Table 4.1 shows that the performance of the improved technique is equivalent to that of a classical forward search when we set the parameters in order to obtain the same computational complexity.

The column PREEST2 shows that a drastic reduction of the complexity of the search can be obtained without any major deterioration in the mutual information.

dataset	n	m	forward	PREEST1	PREEST2
Wisconsin	32	97	0.79	0.79	0.77
Covertype3	54	5000	0.68	0.68	0.68
Lung-Cancer	57	16	0.83	0.84	0.84
Musk2	169	3299	0.78	0.75	0.72
Arrhythmia	280	226	0.73	0.74	0.71
Isolet	618	780	0.78	0.79	0.75
Multi-features	649	1000	0.88	0.88	0.88

Table 4.1: The first column indicates the name of the dataset coming from the UCI ML repository, the two following columns indicate, respectively, the number n of variables and the number m of samples of the selection set and test set. The three last columns report the percentage of mutual information (averaged over the 10 test sets for $d = 1, \ldots, 10$) between the target and the subsets returned by the three algorithms.

4.3 The case $k = 2$: DISR

The kASSI criterion combined with a forward search for $k = d$ is equivalent to the REL approach (Section 3.3.6) and, for $k = 1$, it boils down to the ranking algorithm (Section 3.3.2). The case $k = d - 1$, considered in the previous section, can improve the forward selection when there are enough samples to estimate a $d - 1$ variate density. However, in microarray data, this is rarely the case.

Consider the case $k = 2$.

$$X_S^{2ASSI} = \arg\max_{X_S \in X} \{ \sum_{X_i \in X_S} \sum_{X_j \in X_S} I(X_{i,j}; Y) \} \tag{4.7}$$

This formulation is particularly interesting since it can deal with complementarities up to order 2 (like in Examples 2.2 and 2.3) while preserving the same computational complexity as the MRMR and CMIM criteria (see Table 4.4).

A variant of criterion (4.7) is obtained by replacing the mutual information with a normalized relevance measure defined by [147] (Section 2.3) as

$$\mathrm{SR}(X_S; Y) = \frac{I(X_S; Y)}{H(X_S, Y)} \tag{4.8}$$

This normalization aims to improve the selection strategy by penalizing inputs with large entropies. We name the selection criterion *double input symmetrical relevance* (DISR).

$$X_S^{DISR} = \arg\max_{X_S \in X} \{ \sum_{X_i \in X_S} \sum_{X_j \in X_S} SR(X_{i,j}; Y) \} \tag{4.9}$$

When DISR is combined with a forward selection (like the criteria MRMR, CMIM and

REL, Section 3.3), the variable selected at each step is given by,

$$X_i^{DISR} = \arg \max_{X_i \in X_R} \{ \sum_{X_j \in X_S} SR(X_{i,j}; Y) \} \tag{4.10}$$

where $X_R = X_{-S}$.

4.3.1 A theoretical comparison of DISR with other criteria

It is interesting to compare the proposed criterion with respect to the state-of-the-art methods of Section 3.3. The presented criteria can be analyzed under different perspectives. We stress in Table 4.2,

1. which issues, among relevance, redundancy and complementarity, are taken into account, (based on our analysis of Section 3.3)
2. the ability of a criterion to avoid the estimation of large multivariate densities and
3. whether it returns a ranking of variables.

Table 4.4 reports a comparative analysis of the different techniques in terms of computational complexity of the evaluation step.

methods:	RANK	FCBF	REL	CMIM	MRMR	DISR
Select Relevance	Yes	Yes	Yes	Yes	Yes	Yes
Eliminate Redundancy	No	Yes	Yes	Yes	Yes	Yes
2th order Complementarity	No	No	Yes	No	No	Yes
Avoid Multivariate Density	Yes	Yes	No	Yes	Yes	Yes
Return Ranking	Yes	No	Yes	Yes	Yes	Yes

Table 4.2: Comparison of the properties (relevance, redundancy and complementarity, ability to avoid estimation of large multivariate densities, ability to rank the variables) that are taken into account in each selection criterion (based on our analysis of Section 3.3).

methods:	REL	CMIM	MRMR	DISR
calls of mutual information	1	$d-1$	d	$d-1$
k-variate density	$d+1$	3	2	3
computational cost	$O(d \times m)$	$O(d \times m)$	$O(d \times m)$	$O(d \times m)$

Table 4.3: The computational cost of a variable evaluation using REL, CMIM, MRMR, DISR with the empirical entropy estimator.

We observe from Tables 4.4 and 4.2 that the DISR criterion avoids redundant variables, multivariate density estimation, but selects complementary variables (up to the second order), at the same computational cost than CMIM and MRMR.

4.4 MASSIVE algorithm

The DISR maximization step can be expressed as a weighted Dense Subgraph Problem (DSP) [3] or equivalently as a Dispersion Sum Problem [106]. The dense subgraph problem is defined for a complete undirected graph on the node set $A = \{1, ..., n\}$ where each edge (i, j) takes a weight $w_{ij} \geq 0$, with $w_{ii} = 0$. The goal is

$$maximize \sum_{i \in A} \sum_{j \in A} w_{ij} v_i v_j \tag{4.11}$$

$$subject\ to \sum_{i \in A} v_i = d \tag{4.12}$$

$$v_i \in \{0, 1\}, \quad i \in A \tag{4.13}$$

which means to select a node subset $S \subseteq A$ of fixed size $|S| = d$, such that the total edge weight in the induced subgraph is maximal.

In the dispersion sum problem, n locations are given, where location i is distant from location j by $w_{ij} = \frac{I(X_{i,j};Y)}{H(X_{i,j},Y)}$, and the objective is to establish d facilities among the n locations, as distant as each other (having the maximum average distance between facilities).

The DISR optimization problem can be put in a DSP framework by setting:

1. the ith node represents the variable X_i,
2. the binary variable $v_i, i = 1, \ldots, n$ takes the value 1 if the ith variable is selected and 0 otherwise
3. the weight $w_{ij} = \frac{I(X_{i,j};Y)}{H(X_{i,j},Y)}$ is the symmetrical relevance of the two variables linked by the edge.

The DSP is a NP-hard problem since it can be reduced to the CLIQUE problem (see [106]). However, there exists a branch-and-bound algorithm able to deal with up to 90 variables [106], and several promising results on the performance of greedy searches [21, 111]. [10] indicates that the backward elimination combined with a sequential search (BESR) performs well on binary quadratic problems such as the DSP ([10] uses BESR before a linear programming optimization). The BESR method starts with a set containing

all the variables (i.e. $v_j = 1$ for all $j \in A$) and then selects the variable i whose removal (i.e. $v_i = 1 \leftarrow 0$) induces the lowest decrease of the objective function and so on, till the adequate number of variable is reached (i.e. $\sum_{i \in A} v_i = d$). The procedure is enhanced by an iterative sequential replacement which, at each step, swaps the status of a selected and a non-selected variable (i.e. swapping $v_i = 1$ and $v_j = 0$) such that the largest increase in the objective function is achieved. The sequential replacement is stopped when no further improvement is possible.

The combination of backward elimination and sequential search is a bidirectional search (Section 3.2.3). The backward elimination strategy can be adopted here since considering a n-variables-problem does not mean estimating a n-variate probability distribution as is usually the case in variable selection (Section 3.2); instead, it increases the number of elements w_{ij} that have to be summed in (4.9). The DISR criterion requires only symetrical relevance of pairwise combinations of inputs, that are all computed and stored in the matrix W.

The proposed method combines an evaluation function (DISR) able to select complementary variables and a search algorithm (the backward elimination) also able to select complementary variables. We call this combination Matrix of Average Sub-Subset Information for Variable Elimination (MASSIVE).

4.4.1 Computational Complexity

The proposed implementation works as follows:

1. the DISR-matrix is computed. This step demands $\frac{n(n-1)}{2}$ evaluations $w_{ij} = \frac{I(X_{i,j};Y)}{H(X_{i,j},Y)}$ since the average sub-subset information is symmetric.

2. a backward elimination is applied to the DISR-matrix. This computation has a $O(n^2)$ complexity if the implementation for binary quadratic problems of [85] is adopted (see Algorithm 4.2 for a detailed pseudo-code).

3. a sequential replacement is performed. This has also a $O(n^2)$ complexity [85].

Table 4.4 compares the computational complexity of the evaluation step of different techniques. Note that the table reports a naive implementation of CMIM. A more efficient implementation of CMIM is given in [49]. The total complexity of MASSIVE is in $O(F \times n^2)$ where F is the cost of an estimation of the mutual information involving m samples and three variables (two inputs and one output). For instance, if the empirical entropy is used (Section 2.6), MASSIVE has a cost $O(m \times n^2)$. A complete ranking of variables can be returned by MASSIVE, by selecting the subset composed of the n variables (with no increase in the asymptotic computational cost). In that case, the ranking is given by

Algorithm 4.2 Detailed pseudo-code of the backward elimination for quadratic optimization problem (given a matrix of weights W). The C++ code for this method is freely available on the Internet http://www.ulb.ac.be/di/map/pmeyer/links.html.

Inputs: number d of variables to select, matrix of weights W (with elements w_{ij}) of size $n \times n$
$S \leftarrow \{1, 2, ..., n\}$
Initialize *score* vector: **for** $k \in S$: $score_k \leftarrow \sum_j w_{jk}$
Select minimal score: $b \leftarrow \arg\min_{k \in S}(score_k)$
while $|S| > d$
  Update subset by eliminating worst variable: $S \leftarrow S \setminus b$
  Update *score* vector:$score_k \leftarrow score_k - w_{kb}$, $k \in S$
  Select minimal score: $b \leftarrow \arg\min_{k \in S}(score_k)$
end-while
Output: subset S

the backward elimination since there are no remaining variables to be used for sequential replacement. Note that a conventional forward selection (up to d variables) based on an information criterion (e.g., MRMR) demands $O(n \times d)$ evaluations, each having a complexity depending on d and on the number of samples m (i.e. $O(m \times n \times d^2)$ with the empirical estimator). A conventional backward selection, where the evaluation is performed inside the loop and not precomputed as in MASSIVE, demands $O(n^2)$ evaluations (i.e. $O(m \times n^3)$ with the empirical estimator). Hence, the MASSIVE implementation makes possible the adoption of a BESR strategy at a cost lying between the conventional forward and backward approaches.

methods:	REL	CMIM	MRMR	DISR	MASS
calls of evaluation function	$\frac{(n \times d)}{2}$	$\frac{(n \times d)}{2}$	$\frac{(n \times d)}{2}$	$\frac{(n \times d)}{2}$	$\frac{(n \times n)}{2}$
calls of MI by evaluation	1	$d-1$	d	$d-1$	1
k-variate density	$d+1$	3	2	3	3

Table 4.4: The number of calls of the evaluation function is $n \times d$ in a forward selection strategy. Note that $d = n$ for a backward elimination or for a complete ranking of the n variables. The computational cost of the criteria REL, CMIM, MRMR, DISR and MASSIVE is the number of calls of mutual information (MI) multiplied by the cost of an estimation of the mutual information involving a k-variate density and m samples.

4.5 Experiments with MASSIVE

4.5.1 Experiments on real data

The experimental validation of MASSIVE has been performed in collaboration with Colas Schretter [90]. The validation consists mainly of four steps.

1. We select (or rank) a set of variables using all considered methods.
2. We choose the adequate number of variables to be selected by using a wrapper approach on the ranking made by each method.
3. We assess each selected subset by counting the number of adequate classifications using different learning algorithms. Since the same learning algorithm is used for all the selection methods, the best selection technique is considered as the one leading to the best classification accuracy.
4. We use a statistical test in order to assess when our method significantly outperforms the others.

4.5.1.1 Datasets

The first experiment uses eleven publicly available datasets. The inputs are the gene expressions of a tumor cell (from a patient) and the target variable is the type of cancer. The dataset characteristics are detailed in Table 4.5 where n is the number of variables corresponding to the number of gene expressions, m is the number of tumor samples, and $|\mathcal{Y}|$ is the number of cancer classes.

Each continuous variable is discretized according to two different methods: equal width and equal frequency (described in Section 2.7). The number of intervals of each input is chosen with the Scott criterion (described in Section 2.7). The entropy estimation method used on the discretized dataset is the empirical entropy (described in Section 2.6). This estimator is known to be biased but fast to compute, i.e., in $O(m \times d)$. Note also that all the variable selection algorithms are compared with the same entropy estimation method (on the same number m of samples) and at each step of a forward/backward search, $n - d$ subsets of variables of exactly the same size are compared. As a result, the impact of the bias of the estimator on the ranking of subsets (in each step of each algorithm) is assumed to be weak (see Section 2.6).

MASSIVE has been compared with five state-of-the-art approaches discussed in Section 3.3: the Ranking algorithm, the FCBF and three filters based on the Relevance criterion, the MRMR criterion, and the CMIM criterion.

	datasets	n	m	$\|\mathcal{Y}\|$	ts (min)
1	SRBCT	2308	83	4	0,7
2	Leukemia1	5327	72	3	3
3	DLBCL	7129	77	2	5,5
4	9_Tumors	7129	60	9	4,6
5	Brain_Tumor1	7129	60	2	4,5
6	Brain_Tumor2	12625	50	2	11,4
7	Prostate_Tumor	12600	102	2	23
8	Leukemia2	12582	72	3	16,5
9	11_Tumors	12533	174	11	43
10	Lung_Cancer	12600	203	5	46,4
11	14_Tumors	15009	308	26	198

Table 4.5: The 11 datasets of microarray cancer from http://www.tech.plym.ac.uk/spmc/. The column n contains the number of different sequences measured in the microarray, m the number of clinical samples and $|\mathcal{Y}|$ the number of cancer classes. The column *ts* reports the time needed to select 15 variables with the C++ MASSIVE toolbox on a 3.4GHz Intel Pentium4 processor with 2GB RAM.

A two-fold cross-validation is used to partition each dataset into separate datasets for variable selection and validation. Each selection method selects $d = 15$ variables. Then a wrapper approach (see Section 3.1) selects its optimal subset size among the 15 selected variables. It occured in the experimental session that the classifiers required less than ten variables in order to reach their optimal accuracy (for most datasets).

The learning procedure uses a 3-nearest neighbor classifier and a SVM learning algorithm (with a Gaussian kernel) (Section 2.5). The assessment of each selection criterion goes as follows:

1. The dataset is split into two equal parts: the selection set and the validation set.
2. The ranking of $d = 15$ variables is performed on the selection set,
3. a 10-fold cross-validation (for a number of variables ranging from 1 to 15, according to the ranking returned by the filter) is performed on the validation set. In other words, the validation set is split into ten parts in order to select the best number of variables.
4. The best classification accuracy, i.e., $\frac{\#\, correct\ classifications}{\#\ samples}$, is computed (on the validation set) and reported in Section 4.5.1.2.
5. The selection set and the validation set are swapped and steps 1 to 4 are repeated.

6. The adjusted p-value (using the method available in [9]) of a statistical paired permutation test (with MASSIVE), is computed from the vectors of (classification) error, and a bold-faced notation is used when the p-value is lower than 0.05.

As far as the implementation of the learning methods and of the statistical test is concerned, we used the algorithms made available by the R statistical language [54].A word of caution is necessary here concerning the classification accuracy measured using a 10-fold cross validation. In theory, it would have been better to split the dataset into three parts (selection, learning and validation). However, there are too few samples in microarray datasets to do that. Since we are assessing the selection of variable, we choose to avoid the selection bias by using a selection set and to accomodate with the learning bias introduced by using the learning set as validation set.

4.5.1.2 Results

	MASS	CMIM	MRMR	FCBF	RANK	REL
1	83.13	**42.17**	78.31	**57.83**	77.11	**53.01**
2	87.5	80.56	91.67	**70.83**	86.11	**68.06**
3	81.82	80.52	90.91	76.62	84.42	75.32
4	6.67	5	10	0	8.33	6.67
5	63.33	60	63.33	55	65	61.67
6	74	58	64	58	72	**54**
7	91.18	91.18	90.2	83.33	90.2	**78.43**
8	88.89	79.17	87.5	**43.06**	80.56	**54.17**
9	60.92	**42.53**	54.02	**22.99**	**49.43**	**34.48**
10	79.8	80.3	81.28	74.88	77.83	80.3
11	19.48	**12.99**	18.51	**12.01**	19.48	**12.99**
Avg	67	**57**	66	**50**	**65**	**53**

Table 4.6: **SVM classifier and equal width quantization**: Accuracy with 10-fold cross-validation on the test set. Bold notation is used to identify which techniques are significantly different (pval $< .05$) from MASSIVE in terms of accuracy.

	MASS	CMIM	MRMR	FCBF	RANK	REL
1	89.16	**49.4**	90.36	**49.4**	79.52	**49.4**
2	91.67	81.94	97.22	**75**	90.28	**69.44**
3	85.71	85.71	87.01	**71.43**	87.01	**72.73**
4	16.67	16.67	16.67	6.67	10	6.67
5	60	56.67	61.67	55	58.33	65
6	64	62	66	58	66	58
7	85.29	81.37	88.24	80.39	89.22	83.33
8	87.5	81.94	83.33	**50**	73.61	**58.33**
9	48.85	**36.78**	50.57	**33.33**	40.8-0.225	**27.59**
10	83.74	79.8	80.3	**77.34**	81.77	**77.34**
11	16.56	13.31	15.26	**11.04**	16.56	15.58
Avg	66	**59**	67	**52**	**63**	**53**

Table 4.7: **3NN classifier and equal width quantization**: Accuracy with 10-fold cross-validation on the test set. Bold notation is used to identify which techniques are significantly different (pval $<$ .05) from MASSIVE in terms of accuracy.

	MASS	CMIM	MRMR	FCBF	RANK	REL
1	79.52	**38.55**	79.52	**49.4**	72.29	**43.37**
2	87.5	81.94	88.89	81.94	83.33	81.94
3	94.81	**80.52**	92.2	**77.92**	**81.82**	**80.52**
4	15	8.33	8.33	**5**	8.33	6.67
5	65	65	65	63.33	65	65
6	76	**54**	68	60	60	**52**
7	91.18	84.31	94.12	**81.37**	92.16	84.31
8	93.06	**61.11**	90.28	**51.39**	90.28	**63.89**
9	46.55	**33.33**	53.45	**19.54**	52.3	**34.48**
10	82.27	**74.88**	83.25	**74.88**	78.33	**71.43**
11	17.53	15.58	22.4	**8.44**	19.16	**11.04**
Avg	68	**54**	68	**52**	**64**	**54**

Table 4.8: **SVM classifier and equal frequency quantization**: Accuracy with 10-fold cross-validation on the test set. Bold notation is used to identify which techniques are significantly different (pval $<$ .05) from MASSIVE in terms of accuracy.

	MASS	CMIM	MRMR	FCBF	RANK	REL
1	79.52	**50.6**	84.34	**46.99**	72.29	**55.42**
2	88.89	77.78	90.28	**76.39**	86.11	77.78
3	93.51	83.12	92.21	**83.12**	92.21	**81.82**
4	23.33	18.33	21.67	11.67	21.67	13.33
5	58.33	66.67	66.67	60	60	60
6	62	68	68	60	68	68
7	89.22	81.37	90.2	**75.49**	90.2	84.31
8	83.33	72.22	81.94	**62.5**	90.28	72.22
9	50.57	**38.51**	55.17	**17.24**	51.15	**32.18**
10	82.76	78.33	82.76	**74.88**	78.82	**71.43**
11	18.83	15.26	**29.87**	**10.06**	19.81	**12.01**
Avg	66	**59**	**69**	**53**	66	**57**

Table 4.9: **3NN classifier and equal frequency quantization**: Accuracy with 10-fold cross-validation on the test set. Bold notation is used to identify which techniques are significantly different (pval $< .05$) from MASSIVE in terms of accuracy.

The experimental results show that the MASSIVE method is competitive with the state-of the-art information-theoretic filters. In particular, MASSIVE is significantly better than all other information-theoretic criteria, except MRMR. The results for the FCBF are poor on some datasets because of its internal stop criterion (Section 3.3). Indeed, on several datasets, the FCBF stopped after having selected one variable, only.

4.5.2 Experiments with synthetic datasets

The previous results show that MASSIVE is competitive with state-of-the-art approaches although it is not able to outperform the MRMR approach. A more detailed comparison between MASSIVE and MRMR is then worthy to be done. This section considers two synthetic experiments designed in order to

1. show a situation where the notion of complementarity may be advantageous, and to
2. assess the capability of the MASSIVE algorithm to select the optimal variable set for a large number of samples.

4.5.2.1 Classification accuracy with complementary variables

This experiment shows that the notion of complementarity used by the DISR brings a significant improvement with respect to MRMR (in terms of accuracy) if variable depen-

dency follows specific causal patterns.

Let us consider a set of random variables A, B, Y, S, H whose statistical dependencies are described by the Bayesian network (Section 3.6) of Figure 4.1 and the conditional probability of Table 4.10. The interpretation of the Bayesian network is the following: an abnormal cellular activity (A) can cause a cancer (Y) and activate a blood marker (B). A possible symptom of that cancer is a headache (H). Luckily, headaches may have also innocuous causes, like sinus infection (S).

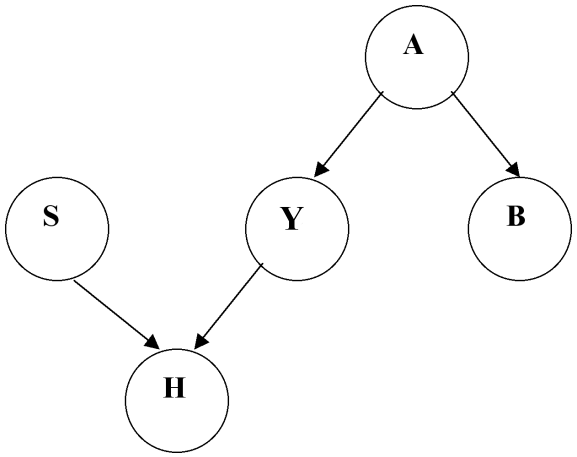


Figure 4.1: Artificial Bayesian network describing the interaction of variables : abnormal cellular activity (A), cancer (Y), blood marker (B), headache (H) and sinus infection (S).

$p(A)$	$p(B\|A=1)$	$p(Y\|A=1)$	$p(H\|S=1,Y=0)$	$p(H\|S=1,Y=1)$
(0.8,0.2)	(0.3, 0.7)	(0.3, 0.7)	(0.2, 0.8)	(0.1, 0.9)
$p(S)$	$p(B\|A=0)$	$p(Y\|A=0)$	$p(H\|S=0,Y=1)$	$p(H\|S=0,Y=0)$
(0.3,0.7)	(0.7, 0.3)	(0.7, 0.3)	(0.2, 0.8)	(0.9, 0.1)

Table 4.10: All variables are binary, p(X)=(P(X=0),P(X=1)). For example, the first column indicates that 20% patients coming to the consultation have an abnormal cellular activity and 70% have a sinus infection; the 4th column indicates that there is 80% chance having some headache either if you have a sinus infection or cancer.

Let us consider a classification task where the goal is to predict the variable Y using both the set $\{A, B, H, S\}$ of variables and a set of ten irrelevant variables. The training set is composed of 300 i.i.d. samples and the test set is composed of 700 i.i.d. samples. A 3-Nearest Neighbor (Section 2.5.2) and a Support Vector Machine algorithm (Section 2.5.3) have been used to assess the performances of both filters using the five first variables selected.

In this example only the three variables $\{A, S, H\}$ are relevant for predicting Y. Indeed,

	MASSIVE	MRMR
3NN	62	**56**
SVM	75	**70**

Table 4.11: Accuracy percentage with 10-fold cross-validation on the test set, boldfaced if p-value by a paired permutation test <0.05. The learning algorithm consider the five first variables selected by both filters.

they form the Markov blanket of Y (see Section 3.6.1) composed of the target's parents, the target's children, and the target's children's parents. The Markov blanket of a variable has been shown to be the minimum set that achieves optimal classification accuracy (see Section 3.6.1).

We show in theoretical and experimental terms that MASSIVE is able to select $\{A, S, H\}$. Indeed, variable S and H are complementary in order to predict Y since

$$C(S; H; Y) = I(S; Y|H) - I(S; Y) > 0$$

MRMR, instead, selects only the variables A and H which are relevant to the output and discards all the others since they are either irrelevant or are redundant with A or H. Indeed, in a pairwise setting S is independent of Y (null relevance) and redundant with H. The different selections of MASSIVE and MRMR have a significant impact on the classification accuracy (Table 4.11). The dataset and the R code used to generate it are available with the MASSIVE C++ code.

4.5.2.2 Optimal variable set selection

Although the goal of these experiments is to assess the potentiality of the complementarity notion in small sample and real variable selection problem, it is worthy investigating how the selection accuracy scales up with the number of samples. In order to carry out this analysis we have used the *SynTReN* generator, a simulator designed to generate microarray data by selecting subnetworks from *E. coli* and *S. cerevisiae* source networks [33]. The mRNA expression levels for the genes in the network are obtained by simulating equations based on the Michaelis-Menten and Hill kinetics under different conditions (see [33]).

The SynTRen simulator has been used to generate 14 different classification tasks where the target is the simulated expression of a target gene and the set of relevant inputs is made of seven genes which are connected to the target in the network. For each of the simulated tasks 500 irrelevant variables were added to the seven genes that regulate the target and training sets with increasing numbers of generated samples ($m = 100, 1000, 2000$). The resulting dataset is then discretized using the equal frequency binning algorithm (see Section 2.7). A MRMR and a MASSIVE variable selection ($d = 7$) was performed. For each

MRMR	MASS	MRMR	MASS	MRMR	MASS
2	3	3	4	3	6
5	6	6	7	6	7
5	7	6	7	6	7
2	2	3	2	3	6
2	3	3	3	3	6
2	2	3	3	3	6
1	1	2	1	2	6
2	2	3	3	3	6
3	4	3	3	3	7
2	2	3	3	3	2
2	3	3	6	3	7
1	1	2	1	2	6
2	3	3	4	3	7
3	3	4	5	4	7
35%	43%	48%	53%	48%	86%

Table 4.12: Table reporting the number of relevant variables selected by MRMR and MASSIVE on 14 datasets of 507 input variables and 100 (column 2,3), 1000 (column 4,5) and 2000 (column 5,6) samples, respectively. The expression of the target is generated by a synthetic microarray data simulator based on Michaelis-Menten and Hill kinetics equations. The last row reports the average covering of the selection.

dataset, the number of relevant variables selected by MRMR and MASSIVE is reported in the table 4.12. We observe that

1. for both algorithms the quality of the selection increases with the number of samples coherently with the conclusion reached in [66],

2. the quality of the MASSIVE selection converges more rapidly toward the optimum.

4.5.3 Conclusion

As shown by the experiments, MASSIVE is a competitive information-theoretic variable selection algorithm. It outperforms four out of five state-of-the-art approaches tested on eleven real datasets. Although MRMR is an effective variable selection technique on these datasets, it is easy to build examples where MASSIVE outperforms MRMR. This is because the combination of DISR and backward elimination allows MASSIVE to select complementary variables. MRMR typically discards indirect interaction characterized by complementarity variables such as the variable sinus infection (S) in the example given in Figure 4.1. However, in the next section (Section 4.6) we introduce a new method of network inference that benefits from this weakness of MRMR.

4.6 Minimum Redundancy Networks (MRNET)

This section introduces our original approach to network inference based on variable selection technique. Using variable selection strategies for network inference has many advantages [65, 133]. For instance:

1. variable selection algorithms can often deal with thousands of variables in a reasonable amount of time. This makes inference scalable to large networks.

2. variable selection algorithms may be easily made parallel, since each of the n selections tasks is independent.

3. variable selection algorithms can use a priori knowledge. For example, knowing the list of regulator genes of an organism improves the selection speed and the inference quality by limiting the search space of the variable selection step to this small list of genes. The knowledge of existing edges can also improve the inference. For example, in a sequential selection process, as used in the forward selection, the next variable is selected given the already selected variables. As a result, the performance of the selection can be strongly improved by conditioning on known relationships.

However, there is a major disadvantage in using a variable selection technique for network inference: it selects indirect interactions (such as variable's children's parents). Indeed, we have seen that the objective of variable selection is selecting, among a set of input variables, those that lead to the best predictive model. We have also seen that the minimum set that achieves optimal classification accuracy in a faithful Bayesian network is the Markov blanket of a target variable, composed of the variable's parents, the variable's children, and the variable's children's parents. The problem is the following: the variable's children's parents are indirect relationships. Hence, drawing a link between the selected variables and the target leads to false conclusions. In order to avoid the problem, one should determine which of the selected variables are indirect links (such as the variable's children's parents) and which are direct. The latter feature render these techniques much less attractive. However, since MRMR relies only on pairwise interactions, it does not take into account the information gain due to conditioning. As a result, the MRMR criterion is less exposed to the inconvenient of most variable selection techniques while sharing their interesting properties. Paradoxically, the weakness of MRMR in the variable selection setting renders that criterion attractive for a network inference task.

The matrix $w_{ij} = I(X_{i,j}; Y)/H(X_{i,j}, Y)$, computed in MASSIVE, allows a fast approximation of the mutual information between any subset and the output Y. However, if we change the target variable Y', another matrix $w'_{ij} = I(X_{i,j}; Y')/H(X_{i,j}; Y')$ has to be computed. In the case of MRNET, the matrix $\text{mim}_{ij} = I(X_i; X_j)$ is the same for every

target variable $Y = X_i,\ i \in A = \{1, 2, ..., n\}$ of the dataset. Hence, a series of supervised MRMR gene selection procedures, where each gene in turn plays the role of the target output Y, can be performed from a single matrix.

4.6.1 MRNet Algorithm

The MRMR method has been introduced together with a forward selection for performing filter selection in supervised learning problems (Section 3.3.8). Let us consider a supervised learning task where the output is denoted by Y and X is the set of input variables. The method ranks the set X of inputs according to a score that is the difference between the mutual information with the output variable Y (maximum relevance) and the average mutual information with the previously ranked variables (minimum redundancy). The rationale is that direct interactions should be well ranked whereas indirect interactions (i.e. the ones with redundant information with the direct ones) should be badly ranked by the method. The forward selection starts by selecting the variable X_i having the highest mutual information to the target Y. The second selected variable X_j will be the one with a high information $I(X_j; Y)$ to the target and at the same time a low information $I(X_j; X_i)$ to the previously selected variable. In the following steps, given a set X_S of selected variables, the criterion updates X_S by choosing the variable which maximizes the MRMR score (see Section 3.3.8). At each step of the algorithm, the selected variable is expected to allow an efficient trade-off between relevance and redundancy.

The network inference approach, that we call MRNET, consists in repeating this selection procedure for each target gene by putting $Y = X_i$ and inputs $X_{-i} = X \setminus \{X_i\}$, $i = 1, \ldots, n$, where X is the set of the expression levels of all genes. For each pair $\{X_i, X_j\}$, MRMR returns two (not necessarily equal) scores s_i and s_j according to (3.25). The score of the pair $\{X_i, X_j\}$ is then computed by taking the maximum between s_i and s_j. A specific network can then be inferred by deleting all the edges whose score lies below a given threshold θ (as in RELNET, CLR and ARACNE). Thus, the algorithm infers an edge between X_i and X_j either when X_i is a well-ranked predictor of X_j ($s_i > \theta$), or when X_j is a well-ranked predictor of X_i ($s_j > \theta$).

An effective implementation of the forward selection using a similarity matrix is given in [85] (see Algorithm 4.2 for backward implementation and Algorithm 4.3 for forward implementation in network inference). This implementation demands an $O(f \times n)$ complexity for selecting f variables. It follows that MRNET has an $O(f \times n^2)$ complexity since the variable selection step is repeated for each of the n genes. In other terms, the complexity ranges between $O(n^2)$ and $O(n^3)$ according to the value of f. Note that the lower the f value, the lower the number of incoming edges per node to infer and consequently the lower the resulting complexity. In practice, we stop the selection of variables when the average

Algorithm 4.3 Detailed pseudo-code of the MRNET algorithm (given the MIM).

Inputs: a matrix of weights MIM (with elements mim_{ij}) of size n
Initialize W matrix to $n \times n$ zeros
for each variable $i \in A = \{1, 2, ..., n\}$
  $S \leftarrow b$ with b the index of the maximum value in the column i of the MIM
  $w_{bi} \leftarrow mim_{bi}$
  $R \leftarrow A \setminus \{i, b\}$
  Initialize $relevance$ vector: **for** $k \in R$: $relevance_k \leftarrow mim_{ik}$
  Initialize $redundancy$ vector: **for** $k \in R$: $redundancy_k \leftarrow mim_{bk}$
  while $w_{bi} > 0$ and $|S| < n$
    Select best variable:$b \leftarrow \arg\max_{k \in R}(relevance_k - redundancy_k/|S|)$
    $w_{bi} \leftarrow relevance_b - redundancy_b/|S|$
    Update subset by selecting best variable: $S \leftarrow \{S, b\}$
    Update search space by removing best variable: $R \leftarrow R \setminus b$
    Update $redundancy$ vector:$redundancy_k \leftarrow redundancy_k + mim_{bk}$, $k \in R$
  end-while
end-for
Output: the weighted adjacency matrix W

redundancy exceeds the relevance. The backward elimination could also be adopted with this implementation. However, this would lead to a computational complexity of $O(n^3)$ even if the inferred network is sparse. Hence, adopting a forward selection accelerates the inference.

Mutual information is a symmetric measure. As a result, it is not possible to derive the direction of the edge from its weight. This limitation is common to all the mutual information network inference methods presented so far. However, this information could be provided by edge orientation algorithms commonly used in Bayesian networks (see 3.6.1).

4.7 The R/Bioconductor package MINET

R is a widely used open source language and environment for statistical computing and graphics [54]. It is a GNU version of S-Plus that has become a reference in statistical analysis [137]. A particular strength of R lies in the ability to write packages containing specific methods that can interact with existing generic tools such as plotting functions for graphs and curves. Bioconductor is an open source and open development software project for the analysis and comprehension of genomic data [53]. The latter is mainly based on the R programming language.

In this work, we have designed the R/Bioconductor package MINET, which stands for Mutual Information NETwork inference. This package has been written in collaboration

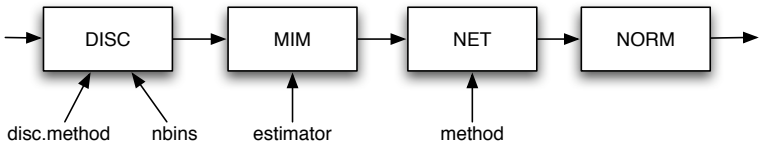


Figure 4.2: The four steps in the MINET function (discretization DISC, mutual information matrix BUILD.MIM, inference MR.NET, ARACNE.NET, CLR.NET and normalization.

with Frédéric Lafitte [89] and is freely available on the R CRAN package resource [54] as well as on the Bioconductor website [53].

4.7.1 Network inference

Once the R platform is launched, the package and its description can be loaded using the following command:

```
library(minet)
library(help=minet)
```

The main function of the package is

MINET(DATASET, METHOD, ESTIMATOR, DISC, NBINS)

where DATASET is a matrix or a dataframe containing the data, METHOD is the inference algorithm (such as ARACNE, CLR or MRNET) and ESTIMATOR is the entropy estimator (such as Miller-Madow, shrink, Dirichlet or empirical).

All the estimators require discretized data. The DISC argument allows the user to choose between two binning algorithms (i.e. equal frequency or equal width) and the parameter NBINS sets the number of bins. This function returns the inferred network as a weighted adjacency matrix with values ranging from 0 to 1. The higher a weight, the higher the evidence that a gene-gene interaction exists.

The function MINET sequentially executes the four subroutines mentioned in Figure 4.2.

4.7.1.1 Discretization

The discretization step uses the function

DISCRETIZE(DATA, DISC= "EQUALFREQ", NBINS=SQRT(NROW(DATA)))

where DATA is the dataset to be discretized and DISC can take two values: "EQUALFREQ" and "EQUALWIDTH" (default is "EQUALFREQ"). NBINS, the number of bins to be used for discretization, is by default set to $\sqrt{m}$, where m is the number of samples, as recommended in [146]. Other choices are possible through internal R functions such as NCLASS.SCOTT(DATA), NCLASS.FD(DATA) or NCLASS.STURGES(DATA).

4.7.1.2 MIM computation

The computation of the mutual information matrix happens with the function

BUILD.MIM(DATA, ESTIMATOR="MI.EMPIRICAL")

The arguments are DATA, the gene expression dataset (or any dataset where columns contain variables/features and rows contain outcomes/samples) and ESTIMATOR, that is the mutual information estimator. The package implements four estimators : "MI.EMPIRICAL", "MI.SHRINK", "MI.SG","MI.MM" (default:"MI.EMPIRICAL") - refering to the estimators explained above.

4.7.1.3 Network inference

Three network inference methods are available in the package. They receive as argument the MIM matrix and return a matrix which is the weighted adjacency matrix of the network. ARACNE(MIM), CLR(MIM), MRNET(MIM) implement respectively CLR, ARACNE and MRNET.

It should be noted, that the modularity of the MINET package allows to test network inference methods on other distance matrices than the MIM or for example to use other mutual information estimators to build the MIM. A correlation matrix can be computed using the R function COR(DATA,MEHOD="PEARSON").

4.7.1.4 Normalization

In the MINET function, a final normalization step sets all weights of the inferred adjancy matrix between 0 and 1 by subtracting the minimum value and dividing by the maximum value of the adjacency matrix.

4.7.1.5 Visualization

In order to display the inferred network, the RGRAPHVIZ package can be used with the following commands (see Figure 4.3):

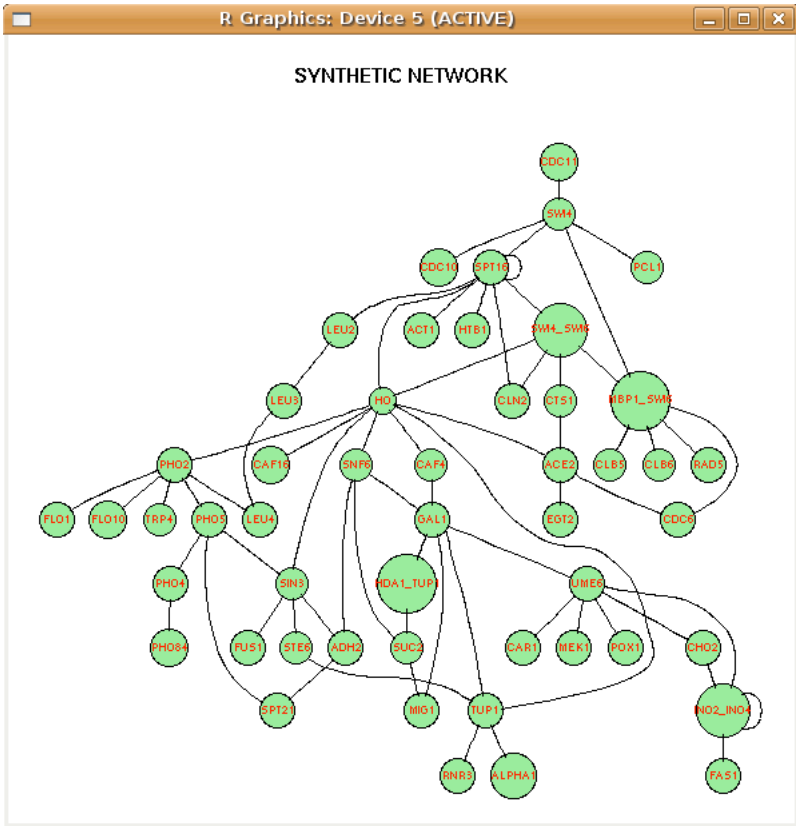


Figure 4.3: Graph generated by MINET and plotted with RGRAPHVIZ

```
library(Rgraphviz)
graph <- as(returned.matrix, "graphNEL")
plot(graph)
```

4.7.2 Validation

In order to assess the methods, the package is equipped with different validation tools. The VALIDATE(NET, SYN.NET,STEPS=50) function allows to compare an inferred network NET to a reference network SYN.NET, described by a Boolean adjacency matrix. The validation process consists in removing the inferred edges having a score below a given threshold and computing the related confusion matrix, for STEPS thresholds ranging from the minimum to the maximum value (in the matrix). A resulting dataframe TABLE containing the list of all the confusion matrices is returned and can be analyzed.

In particular, the function PR(TABLE) returns the related precisions and recalls, RATES(TABLE) computes true positive and false positive rates while the function FSCORES(TABLE, BETA) returns the $F_\beta - scores$. The functions SHOW.PR(TABLE) and SHOW.ROC(TABLE) allow the

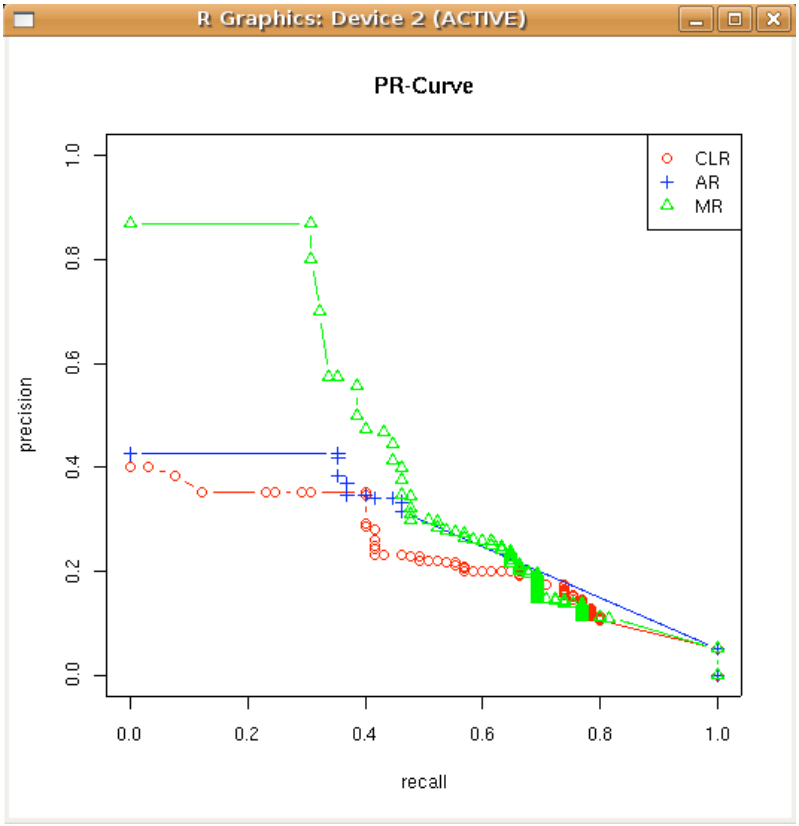


Figure 4.4: Precision-Recall curves plotted with SHOW.PR(TABLE)

user to plot PR-curves and ROC-curves (Figure 4.4), respectively, from a list of confusion matrices.

```
data(syn.net)
net <- minet(syn.data)
table <- validate(net, syn.net)
show.pr(table)
```

4.7.3 Additional Material

In addition to all the available functions implemented in the MINET package (Table 4.13), there is a dataset SYN.DATA of 50 genes and 100 samples that has been generated from the network SYN.NET using the microarray data generator Syntren [33]. A demo script (DEMO(DEMO)) shows the main functionalities of the package using this dataset.

For the sake of computational efficiency, all the inference functions as well as the discretization are implemented in C++. As a reference, a network of five hundred variables may be inferred in less than one minute on an Intel Pentium 4 with 2Ghz and 512 DDR SDRAM.

Function	Action
DISCRETIZE(DATA, DISC, NBINS)	Unsupervised discretization
BUILD.MIM(DATA, ESTIMATOR)	Mutual information matrix estimation
MRNET(MIM)	MRNET algorithm
ARACNE(MIM)	ARACNE algorithm
CLR(MIM)	CLR algorithm
VALIDATE(NET1,NET2)	Computes confusion matrices
PR(TABLE)	Computes precisions and recalls from confusion matrices
RATES(TABLE)	Computes true positive rates and false positive rates from confusion matrices
SHOW.PR(TABLE)	Displays precision-recall curves from confusion matrices
SHOW.ROC(TABLE)	Displays receiver operator characteristic curves from confusion matrices
FSCORES(TABLE)	Returns a vector of F_β-scores from confusion matrices

Table 4.13: Functions in the package MINET

4.8 Experiments on MRNET

The experimental framework consists of four steps (see Figure 4.5): the artificial network and data generation, the computation of the mutual information matrix, the inference of the network, and the validation phase.

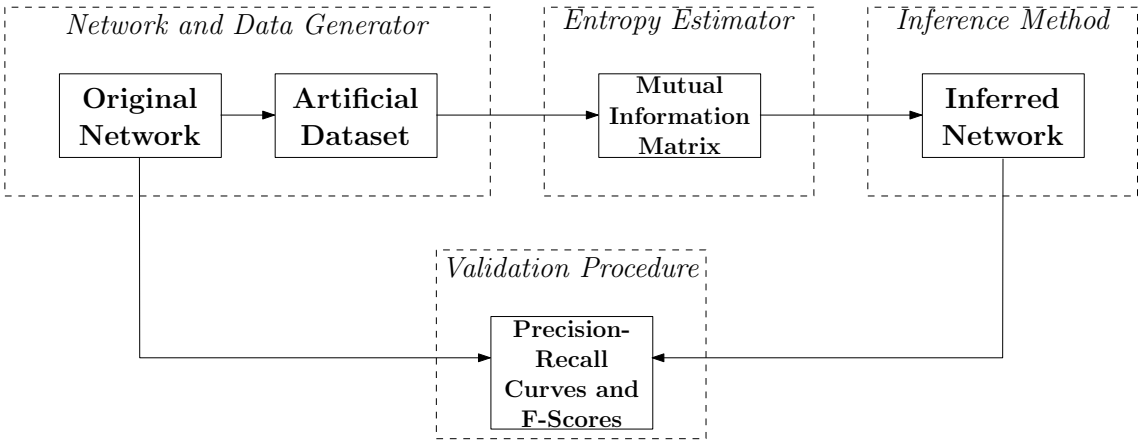


Figure 4.5: An artificial microarray dataset is generated from an original network. The inferred network can then be compared to this *true* network.

4.8.1 Comparing network inference methods

This section details each of the four steps of Figure 4.5. These experiments have been conducted with Frédéric Lafitte [88].

4.8.1.1 Syntactic Microarray Data Generators

In order to assess our algorithm and compare it to other methods, we created a set of benchmarks on the basis of artificially generated microarray datasets. In spite of the limitations of synthetic data, this makes a quantitative assessment of the accuracy possible thanks to the availability of the *true* network underlying the microarray dataset (see Figure 4.5).

We have used two different generators of artificial gene expression data: the data generator described in [113] (hereafter referred to as the *sRogers* generator) and the *SynTReN* generator [33]. The two generators, whose implementations are freely available on the World Wide Web, are sketched in the following paragraphs.

sRogers generator. The *sRogers* generator produces the topology of the genetic network according to an approximate power-law distribution on the number of regulatory

connections out of each gene. The steady-state of the system is provided by integrating a system of differential equations. This generator offers the possibility to obtain $2k$ different measures (k wild-type and k knock-out experiments). These measures can be replicated R times, yielding a total of $N = 2kR$ samples. After the optional addition of noise, a dataset containing normalized microarray measurements is returned.

SynTReN generator. The *SynTReN* generator generates a network topology by selecting subnetworks from *E. coli* and *S. cerevisiae* source networks. Then, transition functions and their parameters are assigned to the edges in the network. Eventually, mRNA expression levels for the genes in the network are obtained by simulating equations based on Michaelis-Menten and Hill kinetics under different conditions. As for the previous generator, after the optional addition of noise, a dataset containing normalized and scaled microarray measurements is returned.

Generation. The two generators have been used to synthesize thirty datasets. Table 4.14 reports for each dataset the number n of genes, the number N of samples and the Gaussian noise intensity (expressed as a percentage of the signal variance).

4.8.1.2 Mutual Information Matrix Estimation

In order to benchmark MRNET versus RELNET, CLR and ARACNE, the same MIM is used for the four inference approaches. Several estimators of mutual information have been proposed in literature [100, 83, 7, 46]. Here we test the Miller-Madow entropy estimator and a parametric Gaussian density estimator (see Sec. 2.6). The data were discretized using the equal width algorithm (see Sec. 2.7) with $|\mathcal{X}_i| = \sqrt{m}$. Note that the complexity of both estimators is $O(m)$, where m is the number of samples. Since the whole MIM cost is $O(m \times n^2)$, the MIM computation could be the bottleneck of the whole network inference procedure for a large number of samples ($m >> n$). We deem, however, that at the current state of the technology this should not be considered as a major issue since the number of samples is typically much smaller than the number of measured variables.

4.8.1.3 Results and Discussion

A thorough comparison would require the display of the PR-curves (Figure 4.6) for each dataset. However, the PR-curve information can often be replaced by the maximum F-score (Section 3.4.3) as done in Table 4.15.

Also, in order to asses the significance of the results, a Macnemar's test (Appendix D.1) is performed .

We may summarize the results as follows:

Dataset	Generator	topology	n	N	noise
RN1	*sRogers*	power-law tail	700	700	0%
RN2	*sRogers*	power-law tail	700	700	5%
RN3	*sRogers*	power-law tail	700	700	10%
RN4	*sRogers*	power-law tail	700	700	20%
RN5	*sRogers*	power-law tail	700	700	30%
RS1	*sRogers*	power-law tail	700	100	0%
RS2	*sRogers*	power-law tail	700	300	0%
RS3	*sRogers*	power-law tail	700	500	0%
RS4	*sRogers*	power-law tail	700	800	0%
RS5	*sRogers*	power-law tail	700	1000	0%
RV1	*sRogers*	power-law tail	100	700	0%
RV2	*sRogers*	power-law tail	300	700	0%
RV3	*sRogers*	power-law tail	500	700	0%
RV4	*sRogers*	power-law tail	700	700	0%
RV5	*sRogers*	power-law tail	1000	700	0%
SN1	*SynTReN*	*S. Cerevisae*	400	400	0%
SN2	*SynTReN*	*S. Cerevisae*	400	400	5%
SN3	*SynTReN*	*S. Cerevisae*	400	400	10%
SN4	*SynTReN*	*S. Cerevisae*	400	400	20%
SN5	*SynTReN*	*S. Cerevisae*	400	400	30%
SS1	*SynTReN*	*S.Cerevisae*	400	100	0%
SS2	*SynTReN*	*S. Cerevisae*	400	200	0%
SS3	*SynTReN*	*S. Cerevisae*	400	300	0%
SS4	*SynTReN*	*S. Cerevisae*	400	400	0%
SS5	*SynTReN*	*S. Cerevisae*	400	500	0%
SV1	*SynTReN*	*S. Cerevisae*	100	400	0%
SV2	*SynTReN*	*S. Cerevisae*	200	400	0%
SV3	*SynTReN*	*S. Cerevisae*	300	400	0%
SV4	*SynTReN*	*S. Cerevisae*	400	400	0%
SV5	*SynTReN*	*S. Cerevisae*	500	400	0%

Table 4.14: Datasets with n the number of genes and m the number of samples.

	Miller − Madow				*Gaussian*			
	RE	CLR	AR	MR	RE	CLR	AR	MR
SN1	0.22	0.24	**0.27**	**0.27**	**0.21**	0.24	**0.3**	0.26
SN2	**0.23**	0.26	**0.29**	**0.29**	0.21	0.25	**0.31**	0.25
SN3	0.23	**0.25**	0.24	**0.26**	0.21	0.25	**0.31**	0.26
SN4	0.22	0.24	**0.26**	**0.26**	0.21	0.25	**0.28**	0.26
SN5	**0.21**	0.23	**0.24**	**0.24**	**0.2**	0.25	**0.27**	0.24
SS1	0.21	0.22	0.22	**0.23**	0.19	**0.24**	**0.24**	0.23
SS2	**0.21**	0.24	0.28	**0.29**	**0.2**	0.24	**0.27**	0.25
SS3	0.21	0.24	0.27	**0.28**	**0.2**	0.24	**0.28**	0.25
SS4	**0.22**	0.24	**0.27**	**0.27**	**0.21**	0.24	**0.3**	0.26
SS5	**0.22**	**0.24**	0.28	**0.29**	**0.21**	0.24	**0.3**	0.26
SV1	**0.32**	0.36	**0.41**	**0.39**	0.3	0.4	**0.44**	0.38
SV2	0.25	0.28	**0.35**	**0.33**	0.25	0.35	**0.36**	0.32
SV3	0.21	0.24	**0.3**	0.28	0.21	0.28	**0.3**	0.27
SV4	0.22	0.24	**0.27**	**0.27**	**0.21**	0.24	**0.3**	0.26
SV5	**0.24**	0.23	**0.29**	**0.29**	0.22	0.24	**0.31**	0.26
S-AVG	0.23	0.25	0.28	0.28	0.21	0.26	0.30	0.27
RN1	0.59	**0.65**	**0.6**	0.61	0.89	0.87	0.92	**0.93**
RN2	**0.5**	**0.57**	**0.5**	**0.49**	0.89	0.87	**0.92**	**0.92**
RN3	0.5	**0.55**	0.5	0.52	0.89	0.87	**0.92**	**0.92**
RN4	**0.46**	**0.51**	**0.47**	**0.47**	0.89	0.87	**0.92**	**0.91**
RN5	**0.42**	**0.46**	0.41	0.4	0.88	0.86	**0.91**	**0.91**
RS1	0.1	**0.11**	0.09	0.1	**0.19**	**0.19**	**0.19**	0.18
RS2	**0.35**	0.32	0.31	0.31	0.45	0.44	**0.47**	**0.46**
RS3	**0.38**	0.32	0.36	**0.38**	0.58	0.56	**0.6**	**0.6**
RS4	0.47	**0.54**	0.47	0.5	0.75	0.75	**0.8**	0.79
RS5	0.58	**0.68**	0.6	0.64	0.9	0.86	**0.93**	**0.93**
RV1	**0.52**	**0.38**	0.46	**0.46**	**0.72**	**0.75**	**0.72**	**0.72**
RV2	0.49	**0.53**	0.49	**0.53**	**0.71**	**0.71**	**0.71**	**0.71**
RV3	0.45	**0.5**	**0.45**	**0.48**	0.69	0.69	**0.71**	**0.71**
RV4	0.47	**0.51**	0.48	0.48	0.69	0.7	**0.74**	0.72
RV5	0.47	**0.52**	**0.47**	0.48	0.7	0.68	**0.74**	0.73
R-AVG	0.45	0.48	0.44	0.46	0.72	0.71	0.74	0.74
Tot-AVG	0.34	0.36	0.36	0.37	0.47	0.49	0.52	0.51

Table 4.15: Maximum F-scores for each inference method using two different mutual information estimators. The best methods (those having a score not significantly weaker than the best score, i.e. p-value < 0.05) are typed in boldface. Average performances on *SynTReN* and *sRogers* datasets are reported respectively in the S-AVG, R-AVG lines.

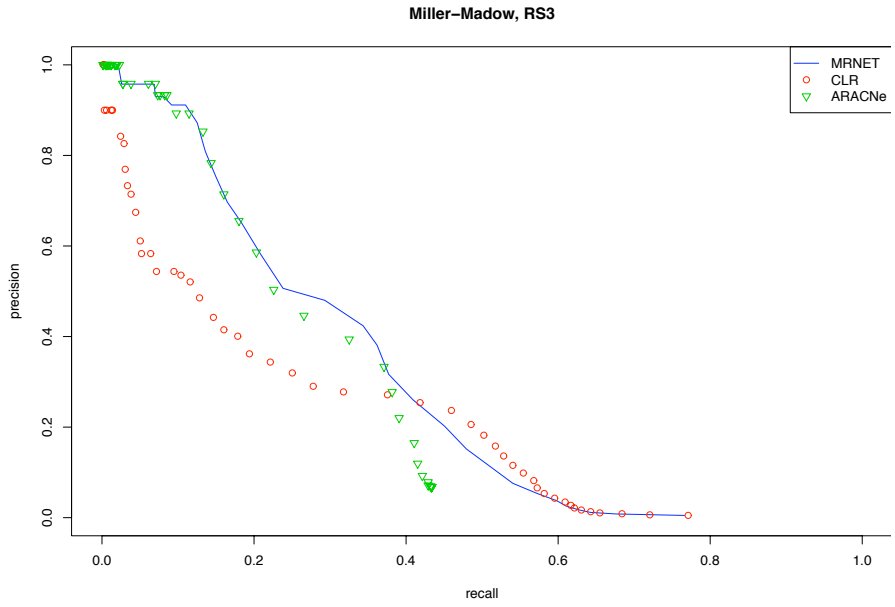


Figure 4.6: PR-curves for the RS3 dataset using Miller-Madow estimator. The curves are obtained by varying the rejection/acceptation threshold.

1. Sensitivity to the number of variables: The number of variables ranges from 100 to 1000 for the datasets RV1, RV2, RV3, RV4 and RV5, and from 100 to 500 for the datasets SV1, SV2, SV3, SV4 and SV5. Figure 4.8 shows that the accuracy and the number of variables of the network are weakly negatively correlated. This appears to be true independently of the inference method and of the MI estimator.

2. Sensitivity to the number of samples: The number of samples ranges from 100 to 1000 for the datasets RS1, RV2, RS3, RS4 and RS5, and from 100 to 500 for the datasets SS1, SS2, SS3, SS4 and SS5. Figure 4.7 shows how the accuracy is strongly and positively correlated to the number of samples.

3. Sensitivity to the noise intensity: The intensity of noise ranges from 0% to 30% for the datasets RN1, RN2, RN3, RN4 and RN5, and for the datasets SN1, SN2, SN3, SN4 and SN5. The performance of the methods using the Miller-Madow entropy estimator decreases significantly with increasing noise, whereas the Gaussian estimator appears to be more robust (see Figure 4.9).

4. Sensitivity to the mutual information estimator: Figure 4.10 exhibits that the Gaussian parametric estimator gives better results than the Miller-Madow estimator. This

is particularly evident with the *sRogers* datasets.

5. Sensitivity to the data generator: The *SynTReN* generator produces datasets for which the inference task appears to be harder, as shown by Table 4.15.

6. Accuracy of the inference methods: Table 4.15 supports that MRNET is competitive with the other approaches.

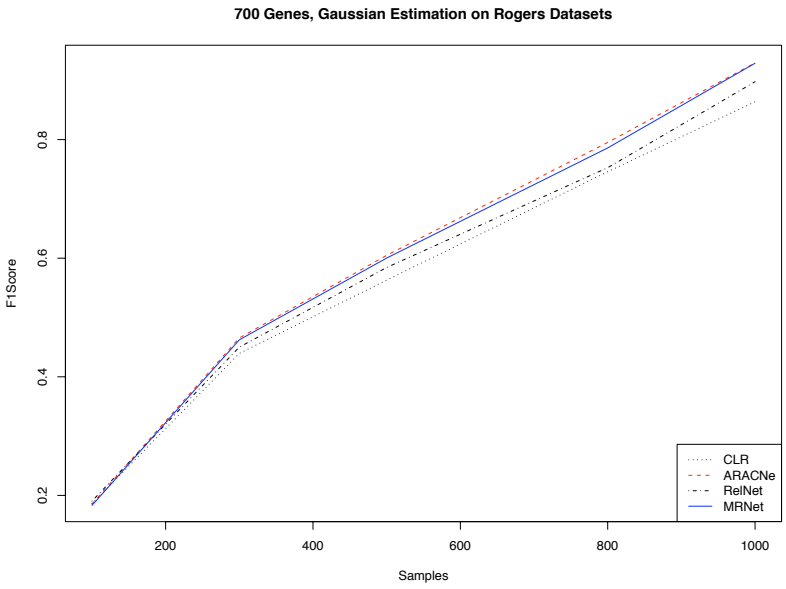


Figure 4.7: Impact of the number of samples on accuracy (*sRogers* RS datasets, Gaussian estimator).

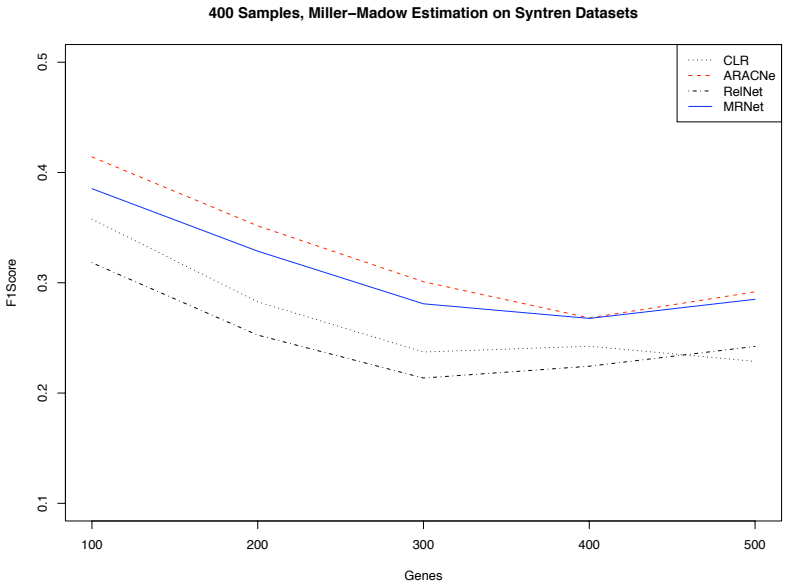


Figure 4.8: Influence of the number of variables on accuracy (*SynTReN* SV datasets, Miller-Madow estimator).

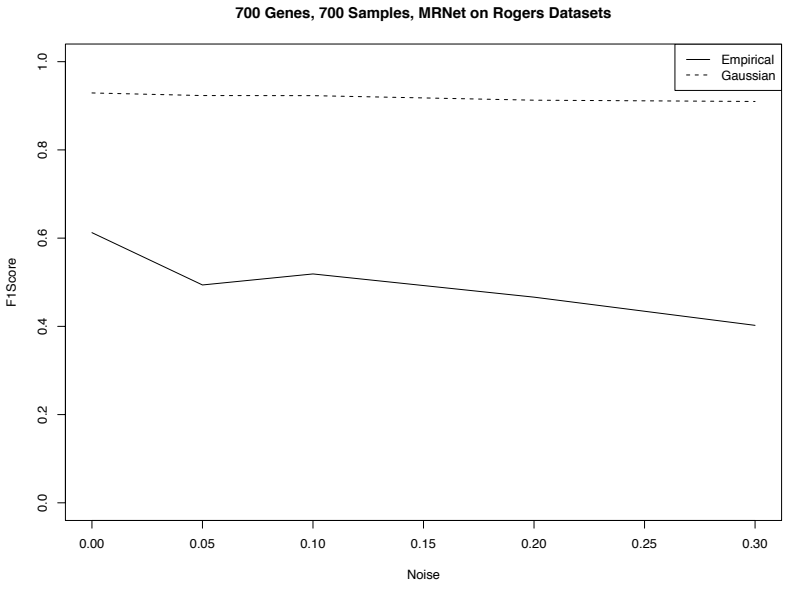


Figure 4.9: Influence of the noise on MRNET accuracy for the two MIM estimators (*sRogers* RN datasets).

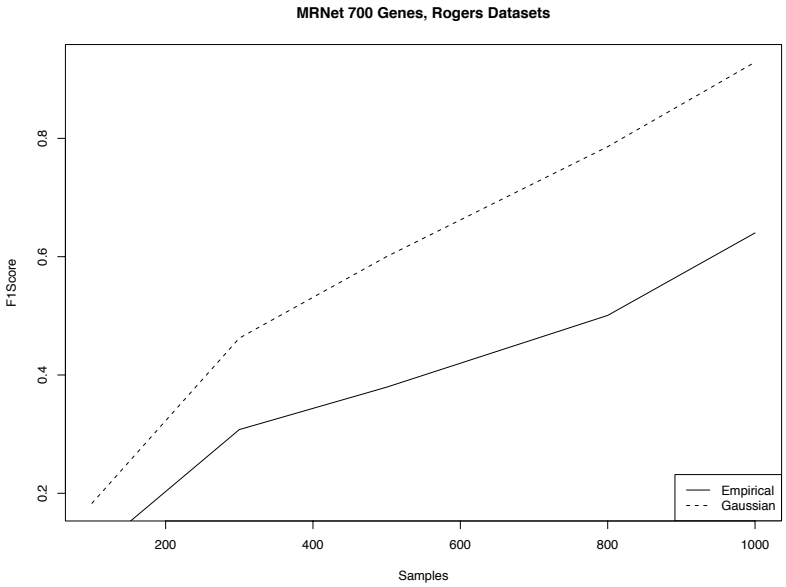


Figure 4.10: Influence of mutual information estimator on MRNET accuracy for the two MIM estimators (*sRogers* RS datasets).

4.8.2 Study of the impact of the estimator

In the previous experiment, two simple estimators have been used for variable selection and network inference. This results from the assumption we made, that the bias of an entropy estimation should penalize equally (with the same number of samples and the same number of variables) all subsets for the task at hand. In the following experiments, we study the impact of the combination estimator-discretization on network inference task containing noise and missing values in the dataset. These experiments have been performed with Catharina Olsen [99].

4.8.2.1 Network generation

The synthetic benchmark relies on twelve artificial microarray datasets generated by the SynTReN generator [33]. This simulator emulates the gene expression process by adopting topologies derived from subnetworks of E.coli and S.cerevisiae networks. Interaction kinetics are modeled by non-linear differential equations based on Michaelis-Menten and Hill kinetics.

The datasets are described in Table 4.16 with respect to the number m of samples and the number n of genes.

No.	Dataset	source net	n	m
1	ecoli_300_300	E.coli	300	300
2	ecoli_300_200	E.coli	300	200
3	ecoli_300_100	E.coli	300	100
4	ecoli_300_50	E.coli	300	50
5	ecoli_200_300	E.coli	200	300
6	ecoli_200_200	E.coli	200	200
7	ecoli_200_100	E.coli	200	100
8	ecoli_200_50	E.coli	200	50
9	ecoli_100_300	E.coli	100	300
10	ecoli_100_200	E.coli	100	200
11	ecoli_100_100	E.coli	100	100
12	ecoli_100_50	E.coli	100	50

Table 4.16: Generated datasets. Number of genes n, number of samples m.

4.8.2.2 Introducing missing values

In order to study the impact of missing values, expression values have been removed from the generated datasets. The number of missing values is distributed according to the $\beta(a, b)$ distribution with parameters $a = 2$ and $b = 5$. The maximal allowed number of missing values is a third of the entire dataset. This distribution was used, instead of the uniform distribution, in order to have input variables with different probabilities of missing values.

4.8.2.3 Setup

For each experiment, ten repetitions were carried out. Each dataset was analyzed using three inference methods (i.e. MRNET, ARACNE and CLR, Sections 3.5 and 4.6) and the following estimators: Pearson correlation, empirical, Miller-Madow, shrink and the Spearman correlation coefficient (Section 2.6). The empirical, the Miller-Madow and the shrink estimator were computed applying the equal width and the equal frequency discretization approaches. Furthermore, the computation was carried out with and without additive Gaussian noise (having 50% variance of the observed values). Each of these setups was also assessed with introduced missing values.

4.8.2.4 Validation

The maximal F-score has been computed for each experiment (see Section 3.4.3). Using a paired t-test, the maximal F-scores were then compared and statistically validated.

4.8.2.5 Results

The results of the synthetic benchmark are collected in Table 4.17 which returns the F-score for each combination of inference method, mutual information estimator and nature of the dataset (noisy vs. not noisy, complete vs. missing data). Note that the maximal F-score is highlighted, together with the F-scores which are not significantly different from the best.

We analyse the results according to four different aspects: the impact of the estimator, the impact of the discretization, the impact of the inference algorithm and the influence of sample and network size. The section concludes with the identification of the best combination of inference algorithm and estimator.

Impact of the estimator:

1. No NA and no noise: the empirical and the Miller-Madow estimator with equal frequency binning lead to the highest F-scores for the MRNET and the ARACNE inference methods. The Spearman correlation is not significantly different from the best, in case of ARACNE, and close to the best in case of MRNET. The CLR method is less sensitive to the estimator and the best result is obtained with the Pearson correlation.

2. Noisy data or missing value (NA): the Pearson correlation and the Spearman correlation lead to the highest F-score for all inference methods. A slight better accuracy of the Pearson correlation can be observed in presence of missing values. The Spearman correlation outperforms the other estimators in MRNET and ARACNE when complete yet noisy datasets are considered. In CLR, Pearson and Spearman lead the ranking without being significantly different.

Impact of the discretization:

1. No NA and no noise: the equal frequency binning approach outperforms the equal width binning approach for all discrete estimators. The gap between the two discretization methods is clearly evident in MRNET and less striking in ARACNE and CLR.

2. NA or noise: the differences between binning algorithms are attenuated.

Impact of the inference algorithm:

1. No NA and no noise: the MRNET inference technique outperforms the other algorithms.

Estimator		MRNET				
		no noise no NA	noise no NA	no noise NA	noise NA	avg
Pearson		0.2006	0.1691	**0.1790**	**0.1611**	0.1775
Spearman		0.3230	**0.1771**	0.1464	0.1333	**0.1950**
Emp	eqf	**0.3420**	0.1551	0.1136	0.0868	0.1744
Emp	eqw	0.2028	0.1650	0.1036	0.0822	0.1384
MM	eqf	**0.3396**	0.1524	0.1140	0.0924	0.1746
MM	eqw	0.1909	0.1592	0.1068	0.0883	0.1363
Shr	eqf	0.3306	0.1506	0.1150	0.0788	0.1688
Shr	eqw	0.1935	0.1574	0.1090	0.0839	0.1360
		ARACNE				
Pearson		0.1117	0.1082	0.1054	**0.1069**	0.1081
Spearman		**0.1767**	**0.1156**	**0.1167**	**0.1074**	**0.1285**
Emp	eqf	**0.1781**	0.1042	0.0993	0.0765	0.1145
Emp	eqw	0.1287	0.1082	0.0892	0.0727	0.0997
MM	eqf	**0.1786**	0.1032	0.0985	0.0783	0.1147
MM	eqw	0.1217	0.1049	0.0931	0.0767	0.0881
Shr	eqf	0.1736	0.1000	0.1009	0.0697	0.1111
Shr	eqw	0.1152	0.1045	0.0898	0.0717	0.0953
		CLR				
Pearson		**0.2242**	**0.1941**	**0.2231**	**0.1911**	**0.2081**
Spearman		**0.2197**	**0.1915**	0.1806	0.1582	0.1863
Emp	eqf	0.2123	0.1729	0.1847	0.1397	0.1774
Emp	eqw	0.2098	0.1724	0.1799	0.1327	0.1737
MM	eqf	0.2128	0.1729	0.1860	0.1427	0.1786
MM	eqw	0.2083	0.1723	0.1845	0.1384	0.1759
Shr	eqf	0.2096	0.1670	0.1864	0.1311	0.1735
Shr	eqw	0.2030	0.1659	0.1822	0.1333	0.1711

Table 4.17: Results using MINET with inference methods MRNET, ARACNE and CLR; noise 50% of the signal variance ("noise"), number of missing values maximal one third of the dataset ("NA"); Estimators: Pearson, Spearman, empirical, Miller-Madow and shrink, the last three with equal frequency ("eqf") and equal width ("eqw") binning approaches; in bold: maximum F-scores and not significantly different values.

2. The situation changes in presence of noisy or missing values. Here CLR appears to be the most robust by returning the highest F-scores for all combinations of noise and missing values.

Impact of the number of sample and network sizes:

The role of network size is illustrated in Figure 4.11, which shows how the F-score decreases as long as the network size increases. This behavior can be explained by the increasing difficulty of recovering a larger underlying network in front of an increasing dimensionality of the modelling task.

In Figure 4.12, the values of the F-score are not clearly in favour of the larger sample sizes. It can be observed that starting from the case $m = 200$ the three curves increase.

Conclusion: It emerges from Table 4.15 that the most promising combinations are represented by the MRNET algorithm with the Spearman estimator and the CLR algorithm with the Pearson correlation. The former seems to be less biased because of its good performance in front of non-noisy datasets while the latter seems to be more robust since less variant in front of additive noise.

4.8.3 Comparison on Biological Data

We proceeded by i) setting up a dataset which combines several public domain microarray datasets about the yeast transcriptome activity, ii) carrying out the inference with the two selected techniques, and iii) assessing the quality of the inferred network with respect to two independent sources of information: the list of interactions measured by means of an alternative genomic technology and a list of biologically known gene interactions derived from the TRANSFAC database.

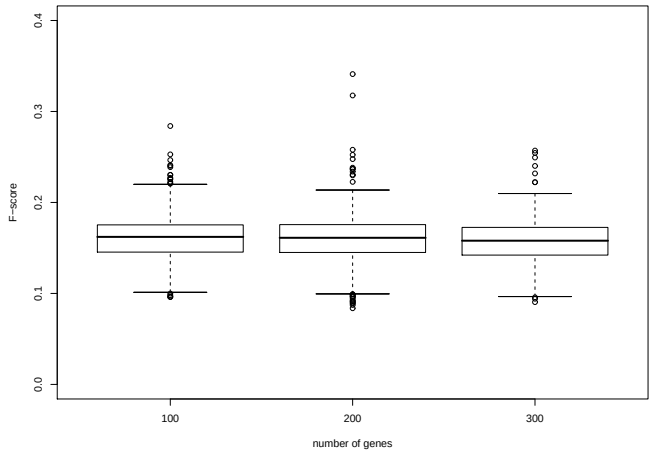
4.8.3.1 The dataset

The dataset has been built by first normalizing and then joining ten public domain yeast microarray datasets, whose number of samples and origin is detailed in Table 4.18. The resulting dataset contains the expression of 6352 yeast genes in 711 experimental conditions.

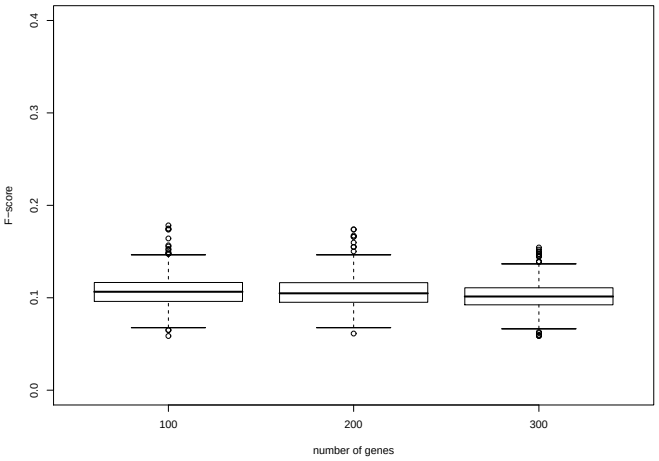
4.8.3.2 Assessment by ChIP-chip technology

The first validation of the network inference outcome is obtained by comparing the inferred interactions with the outcome of a set of ChIP-chip experiments. The ChIP-chip technology, detailed in [16], measures the interactions between proteins and DNA by

a)



b)



c)

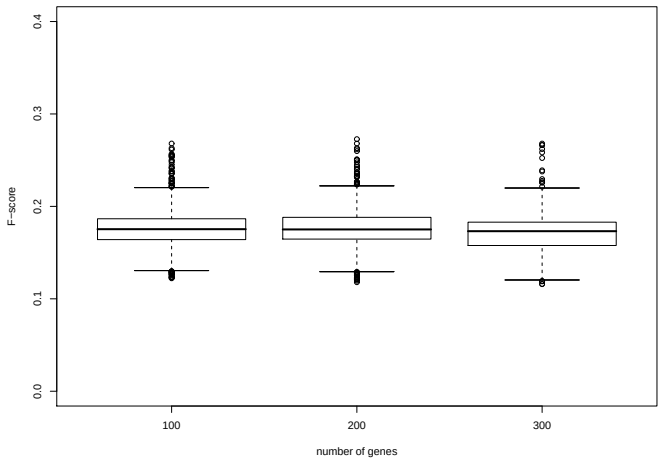
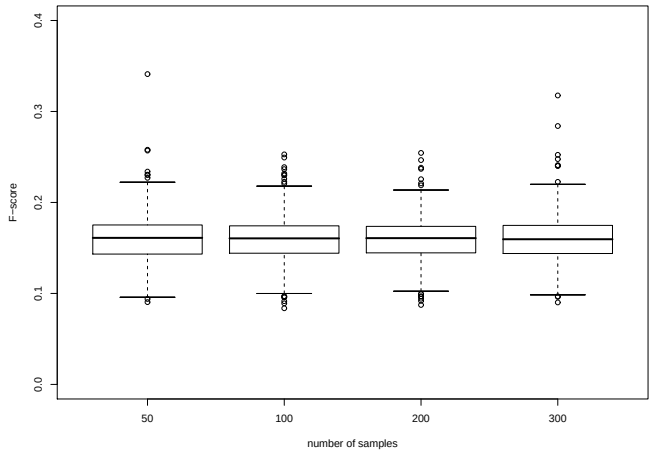
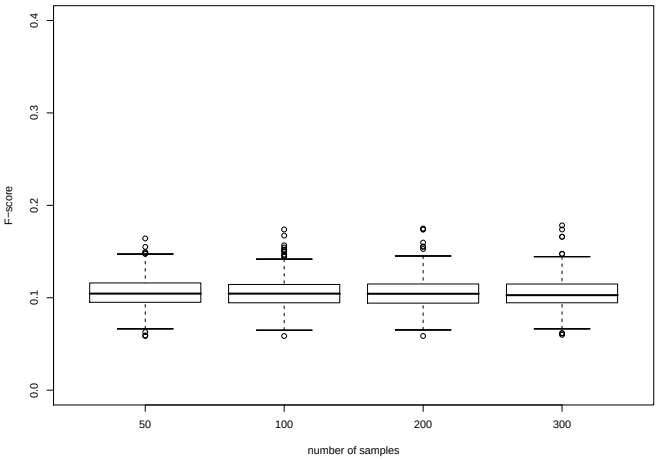


Figure 4.11: Mean F-scores and standard deviation with respect to number of genes, for all 10 repetitions with additive Gaussian noise and no missing values. a) MRNET, b) ARACNE and c) CLR

a)



b)



c)

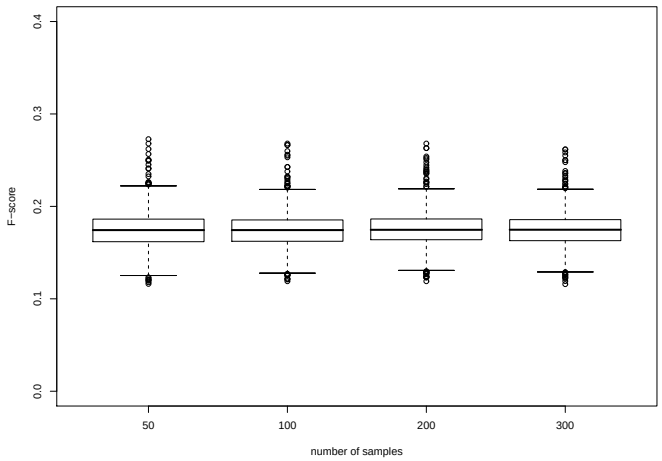


Figure 4.12: Mean F-scores and standard deviation with respect to number of samples, for all 10 repetitions with additive Gaussian noise and no missing values. a) MRNET, b) ARACNE and c) CLR

Datasets	Samples	Origin
1	7	[34]
2	7	[24]
3	77	[127]
4	4	[48]
5	173	[52]
6	52	[63]
7	63	[64]
8	300	[64]
9	8	[98]
10	20	[55]
SUM	711	[99, 79]

Table 4.18: datasets

identifying the binding sites of DNA-binding proteins. The procedure can be summarized as follows. First, the protein of interest is cross-linked with the DNA site it binds to, then double-stranded parts of DNA fragments are extracted. The ones which were cross-linked to the protein of interest are filtered out from this set, reverse cross-linked and their DNA are purified. In the last step, the fragments are analyzed using a DNA microarray in order to identify gene-gene connections. For our purposes it is interesting to remark that the ChIp-chip technology returns for each pair of genes a probability of interaction. In particular we use, for the validation of our inference procedures, the ChIp-chip measures of the yeast transcriptome provided in [58].

4.8.3.3 Assessment by biological knowledge

The second validation of the network inference outcome relies on existing biological knowledge and in particular on the list of putative interactions in Saccaromyces Cerevisiae published in [125].

This list contains 1222 interactions involving 725 genes and in the following we will refer to this as the Simonis list.

4.8.3.4 Results

In order to make a comparison with the Simonis list of known interactions, we limited our inference procedure to the 725 genes contained in the list.

The quantitative assessment of the final results is displayed by means of a receiver operating characteristics (ROC) and the associated area under the curve (AUC). This curve compares the true positive rate (tpr) to the false positive rate (fpr) (see Section 3.4.1)

method	AUC
Harbison	0.66
CLR-Pearson	0.55
MRNET-Spearman	0.54
MRNET-Miller-Madow	0.53
CLR-Miller-Madow	0.52
random	0.5

Table 4.19: AUC for: Harbinson, CLR with Gaussian, MRNET with Spearman, CLR with Miller-Madow, MRNET with Miller-Madow.

Figure 4.13 displays the ROC curves and Table 4.19 reports the associated AUC for the following techniques: the ChIP-chip technique, the MRNET-Spearman correlation combination, the CLR-Gaussian combination, the CLR-Miller-Madow combination, the MRNET-Miller-Madow combination and the random guess.

A first consideration to be made about these results is that network inference methods are able to be significantly better than a random guess also in real biological settings. Also the two combinations which appeared to be the best in synthetic datasets confirmed their supremacy over the Miller-Madow based techniques also in real data.

However the weak, though significative, performance of the networks inferred from microarray data requires some specific considerations.

1. It is worth mentioning that the information coming from microarray datasets is known to be less informative than the one coming from the ChIP-chip technology. Microarray datasets remain nowadays however more easily accessible to the experimental community and techniques able to extract complex information from them are still essential for system biology purposes.

2. Both the microarray dataset we set up for our experiment and the list of known interactions we used for assessment are strongly heterogeneous and concern different functionalities in yeast. We are confident that more specific analyses on specific functionalities could increase the final accuracy.

3. Like in any biological validation of bioinformatics methods, the final assessment is done with respect to a list of putative interactions. It is likely that some of our false positives could be potentially true interactions, or at least deserve additional investigation.

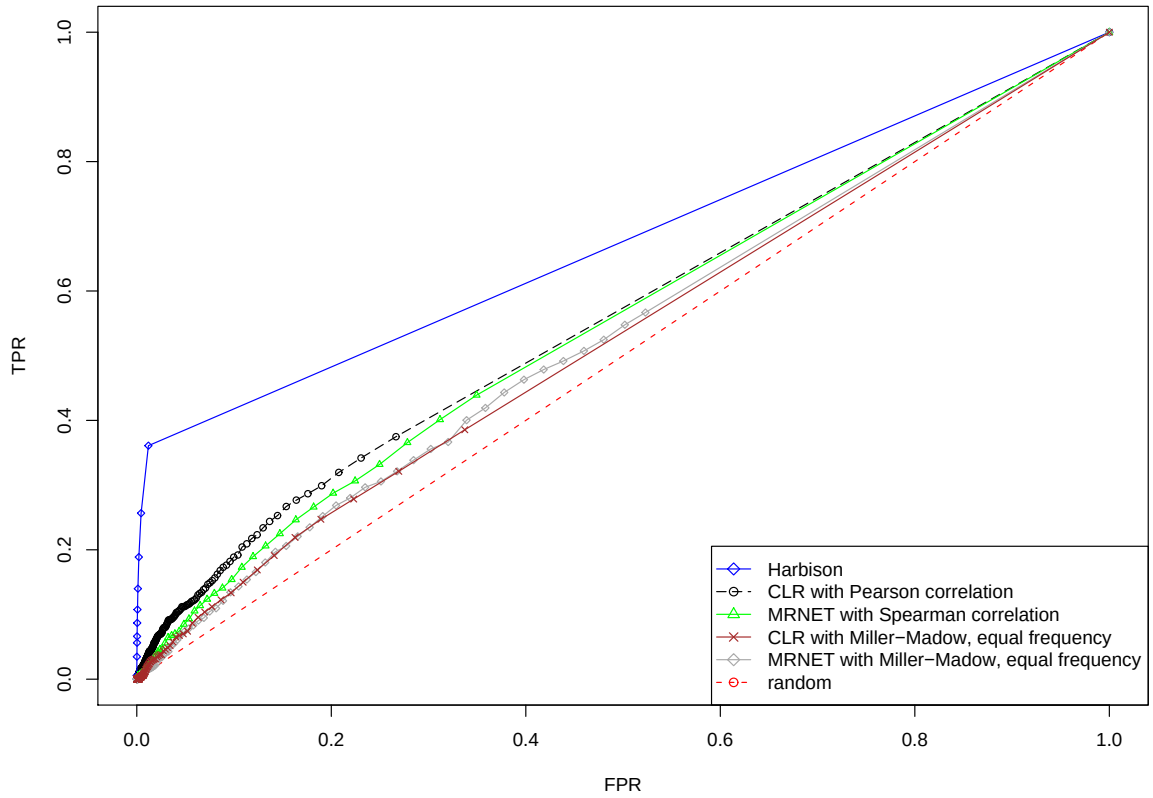


Figure 4.13: ROC curves: Harbison network, CLR combined with Pearson correlation, MRNET with Spearman correlation, CLR combined with the Miller-Madow estimator using the equal frequency discretization method, MRNET with Miller-Madow using equal frequency discretization and random decision.

4.9 Conclusion

Variable selection algorithms are composed of two parts: a search strategy and an evaluation function. We have proposed at first to improve the classical search strategy called the forward selection, by using a faster evaluation function ((d-1)ASSI), that is an approximation of the mutual information. Although our new search method performs as well as the forward selection with a smaller computational cost, it is not adapted to the low number of samples typically encountered in microarray dataset. However, the approximation (DISR) using combinations of only tri-variate (two inputs, one output) probability distributions is well suited for microarray data. This new criterion (DISR) has the following strengths:

1. it copes with complementarities up to order two, while having the same complexity as state-of-the-art methods,

2. it is proved to be in the lower bound of the quantity approximated (mutual information),

3. finding the best subset in terms of DISR evaluation reduces to a well-known quadratic optimization problem (the DSP). This consideration has lead us to bring back the backward elimination strategy in variable selection algorithms. Actually, backward elimination is more adapted to the selection of complementary variables, than forward selection.

Although MRMR is an effective variable selection technique, it omits complementary variables from the selection. This drawback in variable selection becomes a strength for network inference. Indeed, network inference can be performed through successive variable selection procedures if only direct interactions are selected. Hence, algorithms that select complementary variables are less adapted to network inference. MRMR also relies on pairwise mutual informations. Hence, it can be called from a matrix of mutual information (MIM) such as ARACNE or CLR. Our new method, MRNET, has been shown competitive with the state-of-the-art techniques on many datasets. In particular, the combination of MRNET with the Spearman rank correlation outperforms nearly all other information-theoretic inference methods on many synthetic and real datasets. Finally, an open-source package that implements MRNET, CLR and ARACNE methods is now available on the CRAN repository and on the Bioconductor repository.

Chapter 5

Conclusion

This thesis has provided variable selection and network inference methods for datasets having *thousands of variables, tens of noisy samples, non-linear multivariate dependencies between variables* and *little a priori knowledge*. Our objective has been motivated by the analysis of microarray data whose applications can lead to new diagnosis tools (using variable selection) and new target of treatments of various diseases (using network inference).

5.1 Variable Selection

Variable subset selection in the filter/wrapper approach for datasets with many variables can be seen as a combinatorial optimization problem in a very large dimensional space. The evaluation function and the search engine have to be improved consequently in order to deal with problems of high dimensionality. A large number of variables requires

1. a fast evaluation function
2. a search algorithm able to converge towards a "good" solution with as few evaluations as possible.

Furthermore, the low number of samples and the noise, typical in microarray data, render the use of conventional approaches difficult. Finally, the lack of assumptions and the poor knowledge of the *"non-linear combination of features"* existing in genomics make the situation even worse.

We have introduced a new information-theoretic filter for mining microarray data and we compared it with state-of-the-art approaches. Theoretically, our new selection criterion (kASSI) is justified by maximizing a term appearing in the lower bound of the mutual information of a subset. The normalized second-order approximation of our criterion (DISR) is well suited for microarray data because

1. its low computational cost in function of n allows to deal with a large number of variables,

2. large multivariate mutual informations are approximated by combining trivariate mutual informations. As a result, the method can deal with few samples

3. complementarity two-by-two influence the selection (at the same computational cost as state-of-the-art methods). The latter feature is interesting in biology where gene or protein combinations have to be detected.

An appealing aspect of our method lies in the fact that it forms a well-known quadratic optimization problem: the DSP. This observation has led towards a search strategy: backward elimination combined with sequential replacement. This expensive method (in terms of the number of subset evaluations) was made possible through the computation and memorization of the DISR-matrix. The experimental results of that method are on par or better than the other state-of-the-art approaches reviewed here.

5.2 Network Inference

Network inference can be seen as an extension of variable selection. Inferring a network can be done by selecting relevant variables for each variable of the dataset. Since datasets are large, variable selection methods should be fast. In this work, we have introduced a new network inference method, called MRNET. This method relies on an effective method of information-theoretic variable selection called MRMR. Similarly to other network inference methods, MRNET relies on pairwise interactions between genes, making the inference of very large networks (up to several thousands of genes) possible.

MRNET has been experimentally compared to three state-of-the-art information-theoretic network inference methods, namely RELNET, CLR and ARACNE, on various tasks. Microarray datasets have been generated artificially with two different generators, in order to effectively assess their inference power. Also, different mutual information estimation methods have been used. The experimental results showed that MRNET is competitive with the benchmarked information-theoretic methods. This conclusions remained valid when methods have been compared on real data.

Finally, an open-source R package, where MRNET, RELNET, ARACNE and CLR are implemented, has been accepted as an official package of the R CRAN software and of the open library of bioinformatics software: Bioconductor.

5.3 Discussion

This thesis relied on the following claims:

1. Microarray data analysis is a major biomedical issue.
2. Capturing non-linear dependencies leads to better results than assumptions of linearity.
3. The study of variable interactions is relevant to data analysis.

The work developped in the thesis makes now possible a critical discussion of these claims.

Microarray data analysis is a major biomedical issue Microarray experiments allow for the observations of a patient transcriptional response to a disease or to a treatment. We can expect, that with the maturation of microarray technology, the datasets will become bigger (more samples and more variables) and less noisy. Hence variable selection techniques, based on mutual information (non-linear, quite sensitive to noise compared to correlation, fast w.r.t. number of samples and of variables) would become more and more attractive.

Though microarrays have an important future, other new techniques are coming to light, such as proteomic measurements or ChIP-chip experiments, we are convinced that variable selection and network inference algorithms will play a major role in the analysis of these data, too.

Capturing non-linear dependencies leads to better results than the assumption of linearity Our last experiments clearly show that estimators based on linear correlations are more robust to noise than the other estimators. Can we conclude that the data exhibits a linear behavior?

As you assume linearity you are reducing the variance of the estimators because the number of parameters is reduced. However, it may also increases the bias if the dependencies are strongly non-linear. This is the bias-variance trade-off introduced in Section 2.4.1.2. The variance reduction issue might be more important than the bias reduction issue. However, microarray technology is improving, we can expect to have datasets with lower noise and more samples from which non-linear relationships could be easily extracted. For this reason, we believe that information-theoretic methods should become more popular in the future.

Variable interactions are relevant to data analysis The thesis introduced two main variable interactions: redundancy and complementarity. From the experiments, it clearly appears that dealing with redundancy is essential. Our method of variable selection focuses on complementarity by using second order interactions. However, other applications with a lower number of variables and a higher number of samples would allow to use a higher order of complementarity. The kASSI criterion expresses a kind of bias-variance trade-off: as the dimensionality of the multivariate density used for computing mutual information is increased, higher order interactions can be captured. However, doing so can decrease the accuracy of the estimation (and increases the computational cost).

An additional role of complementarity is causality detection via the explaining away effect. Hence, the study of complementarity should remain a promising field.

5.4 Future Direction

This section aims to sketch some future directions to improve this work. We mention three main research axis:

1. Algorithmic improvements:

   (a) Discretization methods: indeed, the last experimental study (Section 4.8) has shown an important impact of discretization on network inference. Hence, new discretization techniques are required to improve the inference.

   (b) Entropy estimators: Spearman and Pearson based estimators appear to be the best ones (Section 4.8), but other estimators based on continuous data should be investigated.

   (c) Search engines for variable selection: many heuristics and meta-heuristics have been omitted from the field of investigation of this work. However, combining them with our evaluation function (DISR) could give rise to new results.

   (d) Subset evaluation function: information-theoretic methods form a subfield of variable selection strategies. There are other non-linear information measures that should be tested in the future.

   (e) Network inference: it appears from our experiments that methods based on linear assumptions are competitive with non-linear ones. As a result, partial correlation and differential equation networks, that have been partly discarded because of their linearity, should be studied.

   (f) Causal inference: in this work, causality is studied under the framework of Bayesian networks. However, causality is also studied in philosophy, statistics or economics. These fields should be investigated and related to this work.

2. Biological validation: our work has focused on theoretical and experimental validation of our original methods. We have shown that they are competitive with state-of-the-art approaches. However, applications of these methods to biological data have to be performed in view of discovering new gene interactions. For instance, the inference of the human transcriptional regulatory networks is a promising field.

3. Extension of techniques: the tools of variable selection and network inference developed in our thesis can be considered as starting points for further developments.

   (a) MASSIVE and MRNET could benefit from the ability to deal with time series datasets. The dynamics of genetic interactions have been omitted in this work but it is becoming more and more important in bioinformatics.

   (b) The ability of detecting hidden variables should significantly improve the accuracy of our methods, because MASSIVE and MRNET implicitly assume that all the relevant variables are in the dataset.

   (c) The arcs produced by MRNET are not oriented, arcs orientation algorithms could improve network inference.

   (d) The trade-off between relevance and redundancy adressed by MRMR weights arbitrarily redundancy and relevance equally. Furthermore, the forward selection might not be the most adequate algorithm to efficiently adress a trade-off as evaluation function. New search methods and new weighting strategies could also lead to better minimum redundancy networks.

Bibliography

[nhg] National human genome project glossary. xi, xii, 4, 150

[1] Almuallim, H. and Dietterich, T. G. (1991). Learning with many irrelevant features. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI91)*, pages 547–552. AAAI Press. 62, 83

[2] Alon, U. (2006). *An Introduction to Systems Biology.* Chapman and Hall. 1, 4, 151

[3] Asahiro, Y., Iwama, K., Tamaki, H., and Tokuyama, T. (2000). Greedily finding a dense subgraph. *Journal of Algorithms*, 34(1):203–221. 92

[4] Basso, K., Margolin, A., Stolovitzky, G., Klein, U., Dalla-Favera, R., and Califano, A. (2005). Reverse engineering of regulatory networks in human b cells. *Nature Genetics*, 37. 1, 2, 7

[5] Battiti, R. (1994). Using mutual information for selecting features in supervised neural net learning. In *IEEE Transactions on Neural Networks.* 61, 64

[6] Beerenwinkel, N., Schmidt, B., Walter, H., Kaiser, R., Lengauer, T., Hoffmann, D., Korn, K., and Selbig, J. (1999). Diversity and complexity of hiv-1 drug resistance: A bioinformatics approach to predicting phenotype from genotype. *PNAS.* 44

[7] Beirlant, J., Dudewica, E. J., Gyofi, L., and van der Meulen, E. (1997). Nonparametric entropy estimation: An overview. *Journal of Statistics.* 41, 112

[8] Bell, D. A. and Wang, H. (2000). A formalism for relevance and its application in feature subset selection. *Machine Learning*, 41(2):175–195. 8, 53, 54, 64

[9] Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society*, Series B(57):289–300. 97

[10] Billionnet, A. and Calmels, F. (1996). Linear programming for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, 92:310–325. 92

[11] Blum, A. and Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245–271. 5, 51, 56, 58, 60, 83

[12] Bockhorst, J. and Craven, M. (2005). Markov networks for detecting overlapping elements in sequence data. In Saul, L. K., Weiss, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 17*, pages 193–200. MIT Press, Cambridge, MA. 71

[13] Bonnlander, B. V. and Weigend, A. S. (1994). Selecting input variables using mutual information and nonparametric density estimation. In *Proceedings of the 1994 International Symposium on Artificial Neural Networks (ISANN94)*. 64

[14] Bontempi, G., Birattari, M., and Bersini, H. (1999). Lazy learning for modeling and control design. *International Journal of Control*, 72(7/8):643–658. 40, 60

[15] Bowers, P. M., Cokus, S. J., Eisenberg, D., and Yeates, T. O. (2004). Use of logic relationships to decipher protein network organization. *Science*, 306(5705):2246–2249. 1, 8

[16] Buck, M. and Lieb, J. (2004). Chip-chip: considerations for the design, analysis, and application of genome-wide chromatin immunoprecipitation experiments. *Genomics*, 83. 122

[17] Burnham, K. P. and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. Springer-Verlag. 33

[18] Butte, A. J., , P. T., Slonim, D., Golub, T., and Kohane, I. (2000). Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. *Proceedings of the National Academy of Sciences*, 97(22):12182–12186. 70, 75

[19] Butte, A. J. and Kohane, I. S. (2000). Mutual information relevance networks: Functional genomic clustering using pairwise entropy measurments. *Pacific Symposium on Biocomputing*, 5:415–426. 2, 9, 75

[20] Caruana, R. and Freitag, D. (1994). Greedy attribute selection. In *International Conference on Machine Learning*, pages 28–36. 57, 58, 59, 83

[21] Chandra, B. and Halldórsson, M. M. (2001). Approximation algorithms for dispersion problems. *Journal of Algorithms*. 92

[22] Cheng, J., Greiner, R., Kelly, J., Bell, D., and Liu, W. (2002). Learning bayesian networks from data: An information-theory based approach. *Artificial Intelligence*, 137(1). 74, 80, 81

[23] Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *Information Theory, IEEE Transactions on*, 14. 74

[24] Chu, S., DeRisi, J., Eisen, M., Mulholland, J., Botstein, D., Brown, P., and Herskowitz, I. (1998). The transcriptional program of sporulation in budding yeast. *Science*, 282. 125

[25] Cornuejols Antoine, M. L. (2002). *Apprentissage Artificiel*. Eyrolles. 1, 3, 37, 38, 40, 41

[26] Cover, T. M. and Thomas, J. A. (1990). *Elements of Information Theory*. John Wiley, New York. 19, 20, 21, 22, 23, 24, 25, 44, 45, 46

[27] Cox, R. T. (1961). *Algebra of Probable Inference*. Oxford University Press. 27, 29, 52, 54

[28] Darbellay, G. and Vajda, I. (1999). Estimation of the information by an adaptive partitioning of the observation space. *IEEE Transactions on Information Theory*. 41

[29] Dash, M. and Liu, H. (IOS Press 1997). Feature selection for classification. *Intelligent Data Analysis*. 60

[30] Daub, C. O., Steuer, R., Selbig, J., and Kloska, S. (2004). Estimating mutual information using b-spline functions - an improved similarity measure for analysing gene expression data. *BMC Bioinformatics*, 5. 41

[31] Davies, S. and Russell, S. (1994). Np-completeness of searches for smallest possible feature sets. In *Proceedings of the AAAI Fall Symposium on Relevance*. 51

[32] Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*. 70

[33] den Bulcke, T. V., Leemput, K. V., Naudts, B., van Remortel, P., Ma, H., Verschoren, A., Moor, B. D., and Marchal, K. (2006). Syntren: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, 7(1):43. 101, 109, 111, 118

[34] DeRisi, J., Iyer, V., and Brown, P. (1997). Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278(5338). 125

[35] Devroye, L., Györfi, L., and Lugosi, G. (1996). *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag. 31, 39, 50, 51

[36] Diettrich, T. G. (1998). Approximate statistical tests for comparing supervised learning algorithms. *Neural Computation*, 10. 159

[37] Dijkstra, E. W. (1984). The threats to computing science. In *ACM South Central Regional Conference*. v

[38] Ding, C. and Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 3(2):185–205. 61

[39] Domingos, P. (2000). A unified bias-variance decomposition and its applications. In *Proc. 17th International Conf. on Machine Learning*, pages 231–238. Morgan Kaufmann, San Francisco, CA. 32, 37

[40] Domingos, P. and Pazzani, M. J. (1996). Beyond independence: Conditions for the optimality of the simple bayesian classifier. In *International Conference on Machine Learning*. 39

[41] Dougherty, J., Kohavi, R., and Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In *International Conference on Machine Learning*, pages 194–202. 45, 46

[42] Dreo, J., Petrowski, A., Siarry, P., and Taillard, E. (2003). *Métaheuristiques pour l'Optimisation Difficile*. Eyrolles. 57

[43] Dreyfus, G. (2002). *Réseaux de neurones*. Eyrolles. 39

[44] Duch, W., Winiarski, T., Biesiada, J., and Kachel, A. (2003). Feature selection and ranking filters. In *International Conference on Artificial Neural Networks (ICANN) and International Conference on Neural Information Processing (ICONIP)*, pages 251–254. 2, 9, 57, 61

[45] Elidan, G. and Friedman, N. (2005). Learning hidden variable networks: The information bottleneck approach. *Jounal of Machine Learning Research*. 81

[46] Faith, J., Hayete, B., Thaden, J., Mogno, I., Wierzbowski, J., Cottarel, G., Kasif, S., Collins, J., and Gardner, T. (2007). Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biology*, 5. xii, 2, 8, 68, 70, 75, 76, 112

[47] Fayyad, U. and Uthurusamy, R. (1996). Data mining and knowledge discovery in databases. *Commun. ACM*, 39(11):24–26. 1

[48] Ferea, T., Botstein, D., Brown, P., and Rosenzweig, R. (1999). Systematic changes in gene expression pattern following adaptive evolution in yeast. *Proc.Natl.Acad.Sci.*, 96. 125

[49] Fleuret, F. (2004). Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555. 2, 8, 9, 57, 61, 64, 65, 83, 93

[50] François, D., Rossi, F., Wertz, V., and Verleysen, M. (2007). Resampling methods for parameter-free and robust feature selection with mutual information. *Neurocomputing*, 70(7-9):1276–1288. 61

[51] Gardner, T. S. and Faith, J. (2005). Reverse-engineering transcription control networks. *Physics of Life Reviews 2*. xi, 1, 2, 7, 68, 70

[52] Gasch, A., Spellman, P., Kao, C., Carmel-Harel, O., M-B-Eisen, Storz, G., Botstein, D., and Brown, P. (2000). Genomic expression programs in the respomse of yeast cells to environmental changes. *Mol.Biol.Cell*, 11. 125

[53] Gentleman, R. C., Carey, V. J., Bates, D. J., Bolstad, B. M., Dettling, M., Dudoit, S., Ellis, B., Gautier, L., Ge, Y., Gentry, J., Hornik, K., Hothorn, T., Huber, W., Iacus, S., Irizarry, R., Leisch, F., Li, C., Maechler, M., Rossini, A. J., Sawitzki, G., Smith, C., Smyth, G. K., Tierney, L., Yang, Y. H., and Zhang, J. (2004). Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, 5. 105, 106

[54] Gentleman R, I. R. (Journal of Computational and Graphical Statistics). R: A language for data analysis and graphics. *1996*, 5. 97, 105, 106

[55] Godard, P., Urrestarazu, A., Vissers, S., Kontos, K., Bontempi, G., van Helden, J., and André, B. (2007). Effect of 21 different nitrogen sources on global gene expression in the yeast saccharomyces cerevisiae. *Molecular and Cellular Biology*, 27:3065–3086. 125

[56] Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182. 5, 50, 55

[57] Guyon, I., Gunn, S., Nikravesh, M., and Zadeh, L. A. (2006). *Feature Extraction: Foundations and Applications*. Springer-Verlag New York, Inc. 2, 34, 57

[58] Harbinson, C., Gordon, D., Lee, T., Rinaldi, N., Macisaac, K., Danford, T., Hannett, N., Tagne, J.-B., Reynlds, D., Yoo, J., Jennings, E., Zeitlinger, J., Pokholok, D., Kellis, M., Rolfe, P., Takusagawa, K., Lander, E., Gifford, D., Fraenkel, E., and Young, R. (2004). Transcriptional regulatory code of a eukaryotic genome. *Nature*. 125

[59] Hartl, D. L. and Jones, E. W. (2001). *Genetics: Analysis of Genes and Genomes, 5th ed.* Jones and Bartlett Publishers. xii, 1, 4, 8, 149, 150, 152

[60] Hastie, T., Tibshirani, R., and Friedman, J. H. (2001). *The Elements of Statistical Learning : Data Mining, Inference, and Prediction.* Springer Series in Statistics. 31, 32, 35, 38, 40, 41, 56, 86

[61] Hausser, J. (2006). Improving entropy estimation and inferring genetic regulatory networks. Master's thesis, National Institute of Applied Sciences Lyon, http://strimmerlab.org/publications/msc-hausser.pdf. 43, 44

[62] Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation.* Prentice Hall International. 1, 38, 45

[63] Huang, A. G. M., Metzner, S., D.Botstein, Elledge, S., and Brown, P. (2001). Genomic expression responses to dna-damaging agents and the regulatory role of the yeast atr homolog mec1p. *Mol.Biol.Cell*, 12. 125

[64] Hughes, T. R., Marton, M. J., Jones, A. R., Roberts, C. J., Stoughton, R., Armour, C. D., Bennett, H. A., Coffey, E., Dai, H., He, Y. D., Kidd, M. J., King, A., Meaer, M., Slade, D., Lum, P., Stepaniants, S., Shoemaker, D., Gachotte, D., Chakraburtty, K., Simon, J., Bard, M., and Friend., S. (2000). Functional discovery via a compendium of expression profiles. *Cell*, 102. 125

[65] Hwang, K., Lee, J. W., Chung, S., and Zhang, B. (2002). Construction of large-scale bayesian networks by local to global search. In *7th Pacific Rim International Conference on Artificial Intelligence.* 61, 103

[66] Jain, A. and Zongker, D. (1997). Feature selection: evaluation, application, and small sampleperformance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19. 102

[67] Jakulin, A. and Bratko, I. (2003). Quantifying and visualizing attribute interactions. 8, 29, 55, 81

[68] Jakulin, A. and Bratko, I. (2004). Testing the significance of attribute interactions. In *Proc. of 21st International Conference on Machine Learning (ICML)*, pages 409–416. 55

[69] Jaynes, E. T. (2003). *Probability Theory: The Logic of Science.* Cambridge University Press. 21

[70] Junker, B. H. and Schreiber, F. (2008). *Analysis of Biological Networks*. Bioinformatics. Wiley-Interscience. 75

[71] Kendall, M. G., Stuart, A., and Ord, J. K. (1987). *Kendall's advanced theory of statistics*. Oxford University Press, Inc. 45, 63

[72] Kennel, M. B., Shlens, J. B., Abarbanel, H. D. I., and Chichilnisky, E. J. (2005). Estimating entropy rates with bayesian confidence intervals. *Neural Computation*, 17(7). 61

[73] Kohane, I. S., Butte, A. J., and Kho, A. (2002). *Microarrays for an Integrative Genomics*. MIT Press. 151

[74] Kohavi, R. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324. 5, 28, 51, 53, 54, 55, 56, 57, 58, 60, 83

[75] Kojadinovic, I. (2005). Relevance measures for subset variable selection in regression problems based on k-additive mutual information. *Computational Statistics and Data Analysis*, 49. 8, 29, 53, 54, 55, 67, 87

[76] Koller, D. and Sahami, M. (1996). Toward optimal feature selection. In *International Conference on Machine Learning*, pages 284–292. 5, 52, 60, 63

[77] Kononenko, I. (1994). Estimating attributes: Analysis and extensions of RELIEF. In *European Conference on Machine Learning*, pages 171–182. 60, 62

[78] Krichevsky, R. and Trofimov, V. (1981). The performance of universal coding. *IEEE Transactions in Information Theory*. 44

[79] Lafitte, F. (2003). Inférence de réseaux génétiques par des méthodes basées sur l'information mutuelle. Master's thesis, Université Libre de Bruxelles (ULB), Belgium. 125

[80] Liang, K. and Wang, X. (2008). Gene regulatory network reconstruction using conditional mutual information. *EURASIP Journal on Bioinformatics and Systems Biology*. 74

[81] Liu, H., Hussain, F., Tan, C. L., and Dash, M. (2002). Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6. 45, 46

[82] Mackay, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press. 8

[83] Margolin, A. A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Favera, R. D., and Califano, A. (2006). ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7. 2, 8, 41, 68, 75, 77, 112

[84] McGill, W. J. (1954). Multivariate information transmission. *Psychometrika*, 19. 8, 27, 29

[85] Merz, P. and Freisleben, B. (2002). Greedy and local search heuristics for unconstrained binary quadratic programming. *Journal of Heuristics*, 8(2):1381–1231. 93, 104

[86] Meyer, P. E. and Bontempi, G. (2006). On the use of variable complementarity for feature selection in cancer classification. In et al., F. R., editor, *Applications of Evolutionary Computing: EvoWorkshops*, volume 3907 of *Lecture Notes in Computer Science*, pages 91–102. Springer. 9, 55, 85

[87] Meyer, P. E., Caelen, O., and Bontempi, G. (2005). Speeding up feature selection by using an information theoretic bound. In *The 17th Belgian-Dutch Conference on Artificial Intelligence (BNAIC'05)*. KVAB. 9, 85, 87, 88

[88] Meyer, P. E., Kontos, K., Lafitte, F., and Bontempi, G. (2007). Information-theoretic inference of large transcriptional regulatory networks. *EURASIP Journal on Bioinformatics and Systems Biology*, Special Issue on Information-Theoretic Methods for Bioinformatics. 9, 85, 111

[89] Meyer, P. E., Lafitte, F., and Bontempi, G. (2008a). Minet: An open source r/bioconductor package for mutual information based network inference. *BMC Bioinformatics*. xii, 9, 71, 73, 85, 106

[90] Meyer, P. E., Schretter, C., and Bontempi, G. (2008b). Information-theoretic feature selection using variable complementarity. *IEEE Journal of Special Topics in Signal Processing*, 2(3). 9, 85, 95

[91] Miller, A. (2002). *Subset Selection in Regression Second Edition*. Chapman and Hall. 56, 58, 59

[92] Mitchell, T. (1997). *Machine Learning*. McGraw Hill. 1, 3, 38, 39, 60

[93] Mitchell, T. M. (2005). Generative and discriminative classifiers: Naive bayes and logistic regression. 39

[94] Moret, B. M. E. and Shapiro, H. D. (1991). An empirical analysis of algorithms for constructing a minimum spanning tree. In Springer, editor, *Lecture Notes in Computer Science*, volume 519. 75

[95] Neapolitan, R. E. (2003). *Learning Bayesian Networks*. Prentice Hall. 25, 43, 68, 78, 79, 81

[96] Nemenman, I. (2004). Multivariate dependence, and genetic network inference. Technical Report NSF-KITP-04-54, KITP, UCSB. 74

[97] Nemenman, I., Bialek, W., and de Ruyter van Steveninck, R. (2004). Entropy and information in neural spike trains: Progress on the sampling problem. *Physical Review Letters*, 69. 8, 41, 43, 44

[98] Ogawa, N., DeRisi, J., and Brown, P. O. (2000). New componenets of a system for phosphate accumulation and polyphosphate metabolism in saccaromyces cerevisiae revealed by genomic expression analysis. *Mol.Biol.Cell*, 11. 125

[99] Olsen, C., Meyer, P. E., and Bontempi, G. (2008). On the impact of missing values on transcriptional regulatory network inference based on mutual information. *EURASIP Journal on Bioinformatics and Systems Biology*. 9, 85, 118, 125

[100] Paninski, L. (2003). Estimation of entropy and mutual information. *Neural Computation*, 15(6):1191–1253. 8, 41, 42, 112

[101] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc. 52, 74, 78

[102] Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press. 68, 77, 78, 79, 82

[103] Peng, H. and Long, F. (2004). An efficient max-dependency algorithm for gene selection. In *36th Symposium on the Interface: Computational Biology and Bioinformatics*. 57, 65

[104] 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. 1, 2, 8, 9, 54, 55, 64, 65, 66

[105] Perkins, S., Lacker, K., and Theiler, J. (2003). Grafting: fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3:1333–1356. 56

[106] Pisinger, D. (2006). Upper bounds and exact algorithms for dispersion problems. *Computers & OR*, 33:1380–1398. 92

[107] Portinale, L. and Saitta, L. (1998). Feature selection. *Applied Intelligence*, 9(3):217–230. 57, 60

[108] Provan, G. and Singh, M. (1995). Learning bayesian networks using feature selection. In *in Fifth International Workshop on Artificial Intelligence and Statistics*, pages 450–456. 51

[109] Provost, F., Fawcett, T., and Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 445–453, Morgan Kaufmann, San Francisco, CA. 71

[110] Quinlan, J. R. (1992). *C4.5: Programs for Machine Learning*. Morgan Kaufmann. 56

[111] Ravi, S. S., Rosenkrantz, D. J., and Tayi, G. K. (1994). Heuristic and special case algorithms for dispersion problems. *Operations Research*, 42(2):299–310. 92

[112] Reunanen, J. (2003). Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research*, 3:1371–1382. 49

[113] Rogers, S. and Girolami, M. (2005). A Bayesian regression approach to the inference of regulatory networks from gene expression data. *Bioinformatics*, 21(14):3131–3137. 111

[114] Rossi, F., Lendasse, A., François, D., Wertz, V., and Verleysen, M. (2006). Mutual information for the selection of relevant variables in spectrometric nonlinear modelling. *Chemometrics and Intelligent Laboratory Systems*, 2:215–226. 63, 64

[115] Sakamoto, Y. and Kitagawa, G. (1987). *Akaike information criterion statistics*. Kluwer Academic Publishers. 33, 37

[116] Särndal, C. E. (1974). A comparative study of association measures. *Psy- chometrika*, 39:165–187. 30

[117] Sauer, U., Heinemann, M., and Zamboni, N. (2007). Getting closer to the whole picture. *Science*, 316(5824):550–551. 7

[118] Schäfer, J. and Strimmer, K. (2005a). An empirical bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6):754–764. 43

[119] Schäfer, J. and Strimmer, K. (2005b). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4(32). 43, 68

[120] Schneidman, E., S.Still, Berry II, M. J., and Bialek, W. (2003). Network information and connected correlations. *Physical Review Letters*, 91. 74

[121] Schurmann, T. and Grassberger, P. (1996). Entropy estimation of symbol sequences. *Chaos*. 44

[122] Scott, D. W. (1992). *Multivariate Density Estimation. Theory,*. Wiley. 46, 47

[123] Shafer, G. (1997). Advances in the understanding and use of conditional independence. *Annals of Mathematics and Artificial Intelligence*. 68, 78

[124] Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*. 19

[125] Simonis, N., Wodak, S., Cohen, G., and van Helden, J. (2004). Combining pattern discovery and discriminant analysis to predict gene co-regulation. *Bioinformatics*, 20. 125

[126] Sokolova, M., Japkowicz, N., and Szpakowicz., S. (2006). Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation. In *Proceedings of the AAAI'06 workshop on Evaluation Methods for Machine Learning*. 72

[127] Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., and Futcher, B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol.Biol.Cell*, 9. 125

[128] Spirtes, P., Glymour, C., and Scheines, R. (2001). *Causation, Prediction, and Search*. MIT Press. 68, 78, 80, 82

[129] Studený, M. and Vejnarová, J. (1998). The multiinformation function as a tool for measuring stochastic dependence. In *Proceedings of the NATO Advanced Study Institute on Learning in graphical models*, pages 261–297. 29

[130] Tishby, N., Pereira, F., and Bialek, W. (1999). The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*. 8, 52, 54, 61

[131] Tourassi, G. D., Frederick, E. D., Markey, M. K., and C. E. Floyd, J. (2001). Application of the mutual information criterion for feature selection in computer-aided diagnosis. *Medical Physics*, 28(12):2394–2402. 54, 65

[132] Tsamardinos, I. and Aliferis, C. (2003). Towards principled feature selection: Relevancy, filters, and wrappers. *Artificial Intelligence and Statistics*. 78

[133] Tsamardinos, I., Aliferis, C., and Statnikov, A. (2003). Algorithms for large scale markov blanket discovery. In *The 16th International FLAIRS Conference*. 103

[134] van de Vijver, M. J., He, Y. D., van't Veer, L. J., Dai, H., Hart, A. A., Voskuil, D. W., Schreiber, G. J., Peterse, J. L., Roberts, C., Marton, M. J., Parrish, M., Atsma, D., Witteveen, A., Glas, A., Delahaye, L., van der Velde T, T., H, H. B., Rodenhuis, S., Rutgers, E. T., Friend, S. H., and Bernards, R. (2002). A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medecine*, 347. 1, 6

[135] van Someren, E. P., Wessels, L. F. A., Backer, E., and Reinders, M. J. T. (2002). Genetic network modeling. *Pharmacogenomics*, 3(4):507–525. 1, 2, 6, 68

[136] van 't Veer, L. J., Dai, H., van de Vijver, M. J., He, Y. D., Hart, A. A., Mao, M., Peterse, H. L., van der Kooy, K., Marton, M. J., Witteveen, A. T., Schreiber, G. J., Kerkhoven, R. M., Roberts, C., Linsley, P. S., Bernards, R., and Friend, S. H. (2002). Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 406. 2, 6

[137] Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S. Fourth Edition*. Springer. 105

[138] Wang, Y., Klijn, J. G., Zhang, Y., Sieuwerts, A. M., Look, M. P., Yang, F., Talantov, D., Timmermans, M., van Gelder, M. E. M., Yu, J., Jatkoe, T., Berns, E. M., Atkins, D., and Forekens, J. A. (2005). Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365. 1, 2

[139] Webb, A. (2002). *Statistical Pattern Recognition*. John Wiley. 1, 32, 34, 38

[140] Whittaker, J. (1990). *Graphical Models in Applied Multivariate Statistics*. Wiley. 2, 6, 25, 68, 78

[141] Wienholt, W. and Sendhoff, B. (1996). How to determine the redundancy of noisy chaotic time series. *International Journal of Bifurcation and Chaos*, 6(1):101–117. 8, 54, 55

[142] Wolpert, D. H. (1997). On bias plus variance. *Neural Computation*, 9(6):1211–1243. 35

[143] Wu, L., Neskovic, P., Reyes, E., Festa, E., and Heindel, W. (2007). Classifying n-back eeg data using entropy and mutual information features. In *European Symposium on Artificial Neural Networks*. 43, 44

[144] Yang, J. and Honavar, V. (1997). Feature subset selection using A genetic algorithm. In *Genetic Programming 1997: Proceedings of the Second Annual Conference*, page 380. Morgan Kaufmann. 57

[145] Yang, Y. and Webb, G. I. (2003a). Discretization for naive-bayes learning: managing discretization bias and variance. Technical Report 2003/131 School of Computer Science and Software Engineering, Monash University. 46

[146] Yang, Y. and Webb, G. I. (2003b). On why discretization works for naive-bayes classifiers. In *Proceedings of the 16th Australian Joint Conference on Artificial Intelligence*. 45, 46, 107

[147] Yao, Y. Y., Wong, S. K. M., and Butz, C. J. (1999). On information-theoretic measures of attribute importance. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 30, 90

[148] Yu, L. and Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224. 2, 8, 52, 53, 54, 55, 61, 62, 65, 83

[149] Zhao, W., Serpedin, E., and Dougherty, E. R. (2008). Inferring connectivity of genetic regulatory networks using information-theoretic criteria. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5(2). 74

[150] Zhao, Z. and Liu, H. (2007). Searching for interacting features. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*. 55

Appendix A

Introduction

A.1 Biological Background

Biological applications have motivated many choices of methods and algorithms in this work. As a consequence, we depict here a short biological background. A word of caution is necessary here: the mechanisms described in the following are a crude picture of the much more complex processes of gene expression and regulation that happen inside a cell. More details can be found in [59].

A.1.1 Gene expression

A protein is a molecule composed of a sequence of amino acids. This sequence determines the shape of the protein and its function. From the DNA coding sequence composed of the four letters A, C, G, T, denoting the bases, to the operational protein, there are two main steps. In the first one, called *transcription*, proteins called *RNA-polymerase* "stick" to the DNA and copy, letter by letter, the coding sequence on a molecule which is called a messenger RNA *(mRNA)*. In the second step, called *translation*, cellular structures called the *ribosomes*, read the mRNA, by words of three letters, called *codons* and assign an amino acid to each codon. Hence, the linear structure of coding DNA determines the linear sequence of amino acids that forms the protein [59], see Figure A.1.

A.1.2 Gene regulation

Some particular proteins, called *transcription factor*s, can bind to the DNA in a region close to a coding sequence. The presence of that protein bound to the DNA can have various effects. If the molecule has some affinity with the RNA-polymerase, then the latter will bind more easily to the DNA, resulting in a facilitated transcription of the coding sequence. As a result, the concentration of mRNA of the coded sequence increases which

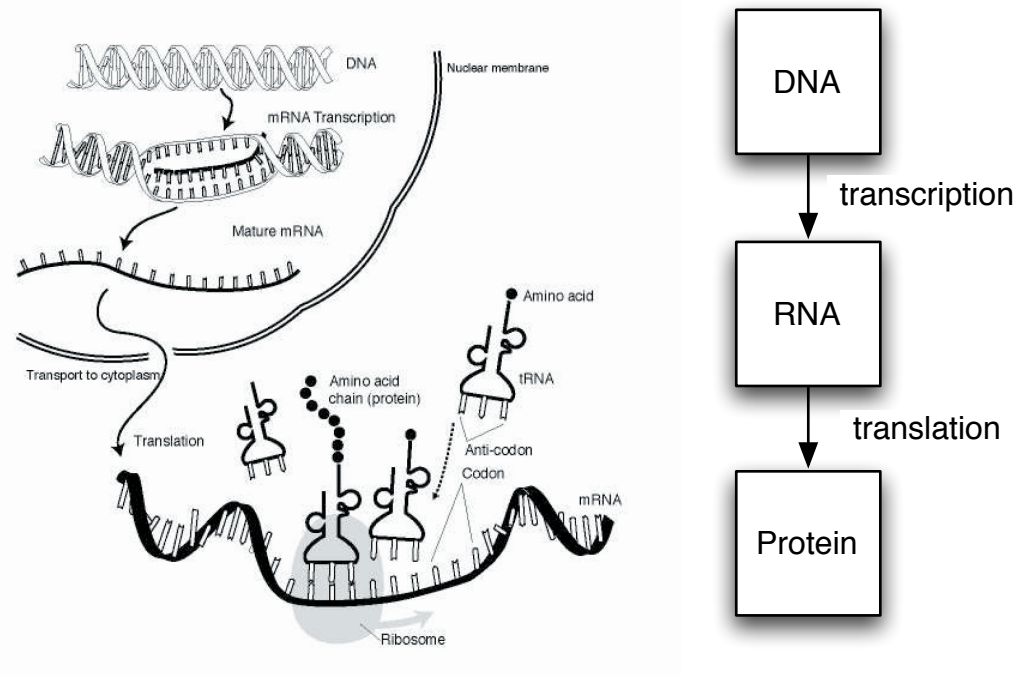


Figure A.1: Principles of gene expression: transcription and translation [nhg]

in turn increases the concentration of the coded protein. In that, the transcription factor is said to activate the transcription and is called an *activator*. However, a transcription factor can also prevent the RNA-polymerase to stick on the DNA, hence blocking the transcription process and reducing the production of the resulting protein. In the latter case, the transcription factor is called a *repressor* [59].

In a bacteria called Escherichia coli, there is a transcription factor that regulates the three genes responsible for the metabolism of lactose. The presence of this transcription factor prevents the RNA-polymerase to transcribe the three genes. Hence, it is a repressor. However, when the cell is in presence of lactose, the transcription factor reacts with the lactose and its structure is changed. As a consequence, the transcription factor cannot bind to the DNA anymore, hence it does not block the transcription process anymore. The three proteins are then being produced and the metabolism of lactose happens. Once the cell is back in an environment without lactose, the concentration of the transcription factor in its original conformation increases and the concentration of the proteins responsible for the metabolism of lactose decreases. This sort of mechanism prevents a cell to continuously produce proteins that are not required [59].

Each protein is a molecular machine that achieves some specific action. Each situation encountered by the cell requires different proteins. The amount of each type of protein in

the cell is continuously adapted as a function of the environment [2]. These mechanisms of regulation can also explain the apparent paradox of cellular differentiation. Indeed, every cell of the same (multicellular) organism contains the same DNA. However, these cells can be very different (for example, a skin cell vs an eye cell) because of the different concentrations of each type of proteins present in it.

A.1.3 Microarray

Microarray is a technology that allows to measure the concentrations of a huge set of mRNA in one experiment. The principle of a microarray experiment is the following. Sequences of genes mRNA are generated synthetically and put on a blade. This blade is composed of thousands of different dots. Different dots correspond to different mRNA sequences, each dot including thousands of identical sequences. The whole mRNA of the cell is taken and by a biological inverse transcription process, a copy of the original DNA (called a *cDNA*) is created. A supplementary fluorescent marker is then added to that cDNA. When the fluorescent cDNA is in contact with the blade containing the synthetic mRNA, an hybridization between them occurs. Each hybridization increases the level of fluorescence of the small square containing the mRNA sequence being measured. Hence, the higher the fluorescence of a square, the higher the concentrations of that mRNA sequence present in the cell. The level of fluorescence of each square is finally measured with an optical measurements (see figure A.2) (more information on DNA chips and microarray experiments can be found in [73]).

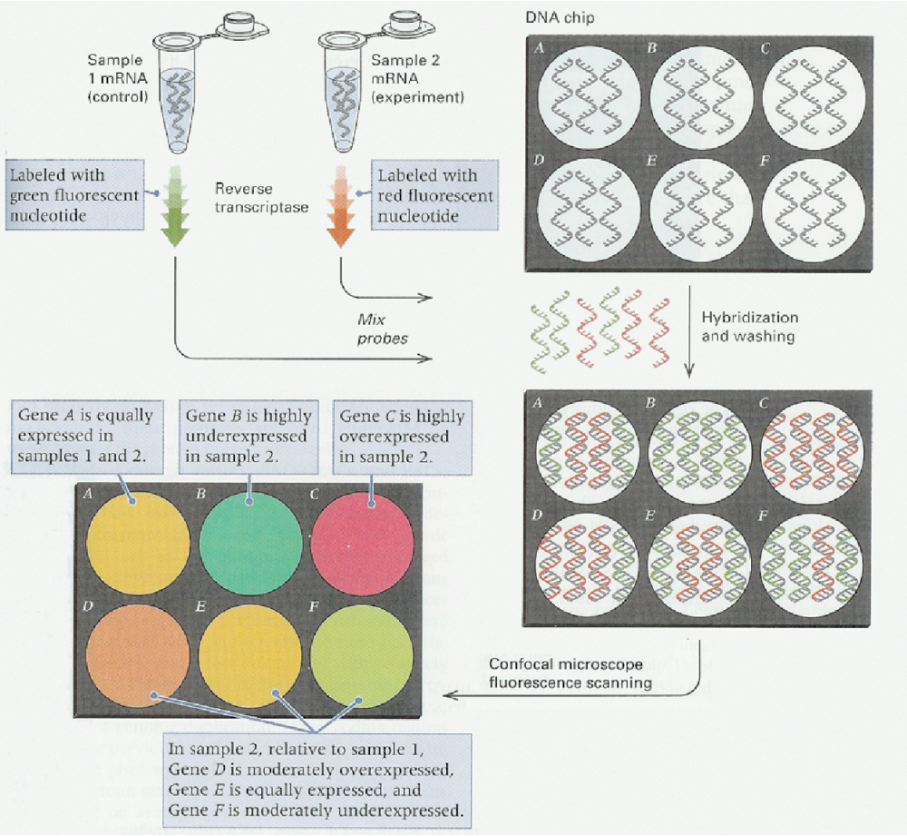


Figure A.2: Principle of DNA chips (from [59])

Appendix B

Preliminaries

B.1 Probability and Estimation Theory

B.1.1 Probability Space

Definition B.1: *A probability space is a triple* $(\Omega, \mathcal{F}, P)$*, where* Ω *is the set of all possible outcomes,* $\mathcal{F}$ *is the* σ*-field of* Ω*, and* P *is a probability measure on the* σ*-field, which satisfies*

1. $0 \leq P(a) \leq 1$ for $a \in \mathcal{F}$
2. $P(\emptyset) = 0$ and $P(\Omega) = 1$
3. If $a_1, a_2, ...$ is a finite or a countably infinite sequence of disjoint sets belonging to $\mathcal{F}$, then
$$P(\bigcup_k a_k) = \sum_k P(a_k)$$

Theorem B.1: *Product rule*

$$P(a \cap b) = P(a, b) = P(a|b)P(b) = P(b|a)P(a)$$

where $p(a|b) = \frac{p(a,b)}{p(b)}$ *if* $p(b) \neq 0$.

Theorem B.2: *Sum rule*

$$P(a \cup b) = P(a + b) = P(a) + P(b) - P(a, b)$$

Theorem B.3: *Bayes Theorem*

$$P(a|b) = \frac{P(b|a)P(a)}{P(b)}$$

Theorem B.4: *Law of Total Probability*

$$P(b) = \sum_{k} P(b, a_i)$$

B.1.2 Random Variable and Expectation

A random variable X is specified by the set $\mathcal{X}$ of values x that the random variable X can assume, and a probability assignment $\{p(x);\ x \in \mathcal{X}\}$. More formally,

Definition B.2: *A real-valued random variable X defined over a probability space $(\Omega, \mathcal{F}, P)$ is a real-valued function (i.e., $X : \Omega \to \mathbb{R}$), such that $\{\omega : X(\omega) \leq x\} \in \mathcal{F}$ for each real x.*

Definition B.3: *The* expectation *of a real-valued random variable X that admits a probability density function $f(x)$, is denoted by $E[X]$ and is defined by*

$$E[X] = \int_{-\infty}^{\infty} x f(x) dx$$

Definition B.4: *The* expectation *of a discrete random variable X that can assume values $-\infty \leq x \leq \infty$, denoted by $E[X]$ is defined by*

$$E[X] = \sum_{x=-\infty}^{\infty} x p(x)$$

B.1.3 Estimation

Definition B.5: *An estimator for θ for sample size m is a function $\hat{\theta} : (\mathcal{X} \times \mathcal{Y})^m \to \Theta$*

Definition B.6: *The error of the estimator $\hat{\theta}$ for a given sample x_k is defined as $L(x_k) = \hat{\theta}(x_k) - \theta(x_k)$.*

Definition B.7: *The mean square error (MSE) of $\hat{\theta}$ is defined as the expected value (probability-weighted average, over all samples) of the squared errors; that is, $MSE(\hat{\theta}) = E[(\hat{\theta} - \theta)^2]$.*

Definition B.8: *The variance of $\hat{\theta}$ is defined as $var(\hat{\theta}) = E[(\hat{\theta} - E[\hat{\theta}])^2]$.*

Definition B.9: *The bias of $\hat{\theta}$ is defined as $B(\hat{\theta}) = E[\hat{\theta}] - \theta$.*

B.2 Interpretations of entropy

Let's take the following example: there are several horse races with the same 8 horses programmed next month and we have to send which horse is the winner of the race to 100 people via a network. Let assume now that we have to pay 1 euro for every bit sent on the network. The simplest way of coding the result could be to send three bits that refer to the index of the winning horse (see Table B.1). Indeed, all the combinations of zeros and ones on three different positions make $2^3 = 8$ possibilities. In that case, it costs us $3bits \times 100people \times 1euro = 300euros$ by race.

However, it is possible to gain some money if the a priori probability of winning for each horse is known. Indeed, in this case it makes sense to use shorter descriptions for more likely events (see coding 2 in Table B.2).

$$H(Horses) = \frac{1}{2} \times \underbrace{-\log_2 \frac{1}{2}}_{1bit} + ... + \frac{4}{64} \times \underbrace{-\log_2 \frac{1}{64}}_{6bits} \tag{B.1}$$

The average length of the new coding scheme is: $\frac{1}{2} \times 1bit + \frac{1}{4} \times 2bits + \frac{1}{8} \times 3bits + \frac{1}{16} \times 4bits + \frac{4}{64} \times 6bits = 2bits$. As a result, it costs us on average: $2bits \times 100people \times 1euro = 200euros$ instead of 300. Note that this is an average, it might be possible to pay more than 300 euros on one race but averaged over a large number of races it should not be the case. This minimal average number of bits required to describe a random variable is precisely its *entropy* (where the base of the logarithm is two). In other words, "the more the randomness, the higher the number of bits required to describe the variable".

Note that it is possible to have an even better coding scheme, such as coding 3 in Table B.2. However, this code is not uniquely decodable. In other words if you receive the sequence of several races, such as 10001. You cannot tell if it is 1-0-001, 10-00-1 or 1-00-01 that has been sent. In other words, entropy measures the minimal average number of bits necessary to code a probability distribution with the additional constraint that each sequence of events has only one possible interpretation given the code.

Horses	1	2	3	4	5	6	7	8
coding 1	000	001	010	100	011	110	101	111

Table B.1: Three bits indicates one of the eight horses.

Horses	1	2	3	4	5	6	7	8
probabilities of winning	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{64}$
coding 2	0	10	110	1110	111100	111111	111101	111110
coding 3	0	1	01	10	00	11	001	011

Table B.2: In these coding schemes, several bits refers a horse in function of its probability of winning (the higher the probability of winning, the lower the number of bits required).

Another intuition for entropy is given by a guessing game. In the guessing game, player one chooses one of the eight horses in his mind and player two has to guess player one's choice. If player one chooses a horse according to the distribution of Table B.2, then the minimal average number of YES-NO questions before the identification of player one's choice is precisely the entropy of the distribution. In this example, two questions are needed in average to find the right horse (have you chosen horse 1? have you chosen horse 2?...). In other words, entropy is a measure of unpredictability of a random variable.

B.3 Bias-variance trade-off

Since $KL(p;\hat{p})$ is not symmetric, one can also develop $E_{D_m}[KL(\hat{p};p)]$ instead of $E_{D_m}[KL(p;\hat{p})]$. That is,

$$E_{D_m}[KL(\hat{p};p)] = E_{D_m}[\sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}\hat{p}(y,x)\log\hat{p}(y|x) - \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}\hat{p}(y,x)\log p(y|x)]$$

$$= E_{D_m}[\textstyle\sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}\hat{p}(y,x)\log\hat{p}(y|x)] - \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}E_{D_m}[\hat{p}(y,x)]\log p(y|x)$$
$$-\sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}E_{D_m}[\hat{p}(y,x)]\log E_{D_m}[\hat{p}(y|x)] + \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}E_{D_m}[\hat{p}(y,x)]\log E_m[\hat{p}(y|x)]$$

$$= E_{D_m}[\textstyle\sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}\hat{p}(y,x)\log\hat{p}(y|x)] - \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}E_{D_m}[\hat{p}(y,x)]\log E_{D_m}[\hat{p}(y|x)]$$
$$+\sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}E_{D_m}[\hat{p}(y,x)]\log E_{D_m}[\hat{p}(y|x)] - \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}E_{D_m}[\hat{p}(y,x)]\log p(y|x)$$

$$= \underbrace{E_{D_m}[KL(\hat{p};E_{D_m}[\hat{p}])]}_{variance} + \underbrace{KL(E_{D_m}[\hat{p}];p)}_{bias}$$

Here the variance term is precisely the expected divergence between the model and the expected model over all datasets.

Appendix C

State-of-the-art

C.1 Theorem 3.3

[Strong and weak relevance]A variable X_j is "strongly relevant" iff there exists some x_j, y and x_{-j} for which $p(x) > 0$ such that

$$p(y|x) \neq p(y|x_{-j})$$

A variable X_j is "weakly relevant" iff it is not strongly relevant, and there exists a subset of variables X_S of X_{-j} for which there exists some x_j, y and x_S with $p(x_j, x_S) > 0$ such that

$$p(y|x_j, x_S) \neq p(y|x_S)$$

A variable is irrelevant iff it is not relevant (weakly or strongly).

Hence, a variable is irrelevant if

$$\forall X_S \subseteq X_{-j} \ : \ p(Y|X_S, X_j) = p(Y|X_S)$$

and relevant otherwise

$$\exists X_S \subseteq X_{-j} \ : \ \ p(Y|X_S, X_j) \neq p(Y|X_S)$$

by applying Theorem 2.7, we obtain:

$$p(Y|X_S, X_j) = p(Y|X_S) \Leftrightarrow I(X_j; Y|X_S) = 0$$

and

$$p(Y|X_S, X_j) \neq p(Y|X_S) \Leftrightarrow I(X_j; Y|X_S) > 0$$

A variable is strongly relevant if

$$p(Y|X_{-j}, X_j) \neq p(Y|X_{-j})$$

and weakly relevant if

$$p(Y|X_{-j}, X_j) = p(Y|X_{-j}) \text{ and}$$
$$\exists X_S \subset X_{-j} \ : \ p(Y|X_S, X_j) \neq p(Y|X_S)$$

by Theorem 2.7:

$$p(Y|X_{-j}, X_j) = p(Y|X_{-j}) \Leftrightarrow I(X_j; Y|X_{-j}) = 0$$

and

$$p(Y|X_S, X_j) \neq p(Y|X_S) \Leftrightarrow I(X_j; Y|X_S) > 0$$

As a result, a variable X_j is irrelevant to Y if:

$$\forall X_S \subseteq X_{-j} \ : \ I(X_j; Y|X_S) = 0 \tag{C.1}$$

A variable X_j is strongly relevant to Y if:

$$I(X_j; Y|X_{-j}) > 0 \tag{C.2}$$

A variable X_j is weakly relevant to Y if:

$$I(X_j; Y|X_{-j}) = 0 \quad \text{AND} \quad \exists X_S \subset X_{-j} \ : \ I(X_j; Y|X_S) > 0 \tag{C.3}$$

C.2 Theorem 3.4

Let $X_{M_j} \subset X_{-j}$. X_{M_j} is a Markov blanket for X_j if X_j is conditionally independent of $(Y, X_{-(j,M_j)})$ given X_{M_j}

Using theorem 2.7 to the definition, we have:

X_j is conditionally independent of $(Y, X_{-(j,M_j)})$ given X_{M_j}
$\Leftrightarrow I(X_j; (Y, X_{-(j,M_j)})|X_{M_j}) = 0.$

Appendix D

Contributions

D.1 McNemar test

The McNemar test [36] states that if two algorithms A and B have the same error rate, then

$$p\left(\frac{(|N_{AB} - N_{BA}| - 1)^2}{N_{AB} + N_{BA}} > 3.841459\right) < 0.05$$

where N_{AB} is the number of incorrect edges of the network inferred from algorithm A that are correct in the network inferred from algorithm B and N_{BA} is the counterpart.