DEPARTMENT OF PHYSICS

FACULTY OF SCIENCES AND TECHNOLOGY
UNIVERSITY OF COIMBRA

Machine Learning approaches in Genome-Wide Association Studies

SNP2Net: a tool for gene-based predictive modeling

Master Thesis in Biomedical Engineering

João Valente Duarte

2009

Preface

The work presented in this thesis was conducted during my Erasmus experience, the mobility programme that allowed me to undertake this work at the University of Pavia, Italy. So, during the past year I was integrated in the Bioinformatics & Data Mining Group, working in the laboratory of Bio-medical informatics, Department of Computer Science and Systems.

This experience abroad allowed me to carry out scientific work on very interesting, useful and largely expanding fields as biomedical informatics and bioinformatics. Moreover, I was able to work together with highly qualified people in this area.

Beyond that, in addition to having known the reality and functioning of this group, I had the opportunity to have contact with people from different cultures and with different visions of the world.

Acknowledgments

Many people not only contributed to the success of this work but also of my entire academic passage.

I begin by sincerely acknowledging Professor Riccardo Bellazzi, the supervisor of this thesis, who welcomed, guided and taught me, giving me scientific training and formative experience of great value.

It would be insufficient to thank all the friends of the Bioinformatics & Data Mining Group, who made my experience in Pavia a great journey, making me feel one of them and making me go to the lab with great determination every day.

It would take a lot of space to name them all, but let me thank in particular Angelo and Malo, who helped and supported me unreservedly. I really thank you!

I also want to thank my course and Erasmus fellows (Ricardo, Tiago, Sofia, once again too many to name all…), that made my academic journey an extraordinary life experience. In particular, thanks to both André and Pedro for making me feel at home, miles away from it, with their friendship.

A warmly acknowledgment to my family, specially my parents Rui and Fátima, who always believed in me and allowed me to pursue my goals with indispensable support. Without them nothing of this would be possible!

A special thanks to Joana for her patience and understanding.

Purposely in the end because the latter are the most important, the last words of precious meaning thanks go to my brother Filipe, for his unconditional support and incalculable value of sharing every moment of my life, since even before our birth.

Acronyms

Acronym	Description
AS	Association Studies
bp	Base pair
BN	Bayesian network
CPT	Conditional probability table
CT	Classification tree
DAG	Directed acyclic graph
DNA	Deoxyribonucleic acid
EM	Expectation-maximization (algorithm)
GWAS	Genome-wide association studies
HD	High density
MEP	Minimal error pruning
SNP	Single Nucleotide Polymorphism

Contents

Abstract

Bayesian networks are powerful instruments to learn genetic models from association studies data, since they allow to derive the correlation between genetic markers and phenotypic traits, as well as the relationships between the markers themselves. However, learning Bayesian networks is often non-trivial due to the high number of variables to be taken into account in the model. Therefore, we developed a Matlab environment software tool, *SNP2Net*, which implements an approach based on classification trees to create an abstraction of the variable space that suitably reduces its dimensionality without losing information. Specifically, the SNPs (genetic variables in the model) related to the same gene are mapped to one meta-variable.

The tool allowed testing the innovative approach on several datasets and, furthermore, to evaluate the phenotype predictive performance of the models built using this method. The results showed that the approach presented is able to derive a gene-based predictive model based on SNPs data, obtaining higher predictive performances than the single-SNP model. Such model is more parsimonious than the one based on single-SNPs, while preserving the capability of highlighting predictive SNPs configurations.

The tool can then be applied to analyze the data coming from association studies, using genes as predictors for the phenotype, as a suitable alternative to haplotypes, which are often used as prediction factors, too.

Resumo

As redes Bayesianas são instrumentos poderosos para aprender modelos genéticos de dados de estudos de associação, uma vez que a correlação entre marcadores genéticos e traços fenotípicos pode ser derivada, bem como as relações entre os próprios. No entanto, o aprendimento das redes Bayesianas é muitas vezes não-trivial, devido ao elevado número de variáveis que devem ser tidas em conta no modelo. Desta forma, desenvolvemos uma aplicação de software em ambiente Matlab, *SNP2Net*, que implementa uma abordagem baseada em árvores decisionais para criar uma abstracção do espaço de variáveis que reduz a sua dimensão de forma adequada, sem perder informação. Especificamente, os SNPs (variáveis genéticas do modelo) pertencentes ao mesmo gene são mapeados para uma meta-variável.

A aplicação desenvolvida permitiu testar a abordagem inovadora em vários conjuntos de dados e, além disso, permitiu também avaliar o desempenho do método no que diz respeito à previsão fenotípica. Os resultados mostraram que a abordagem apresentada é capaz de extrair um modelo predictivo baseado em informação sobre genes a partir de dados de SNPs, obtendo desempenhos predictivos superiores aos obtidos pelo modelo baseado num único SNP. Este modelo é mais parcimonioso do que um modelo baseado em conjuntos de um único SNP, preservando simultaneamente a capacidade de destacar configurações de SNPs predictivas.

A ferramenta pode então ser aplicada na análise de dados provenientes de estudos de associação, usando genes como factores predictivos do fenótipo, e sendo uma alternativa adequada aos haplotipos, que muitas vezes são também utilizados como elementos de previsão.

Introduction

One of the most challenging goals of current biomedical research is to link the genotypic and phenotypic information generated by high-throughput experimental technologies (Botstein & Risch, 2003).

Nowadays, an investigator can sample hundreds of thousands of variables in parallel; thus, the traditional data inspection and analysis methods are quickly becoming inadequate. In an era in which an entire genome can be sequenced and annotated in a matter of days, *ad-hoc* analysis methods need to be developed in order to address the problems related with the statistical significance of the analysis results. These tools, however, should be easily accessible to the average researcher, facilitating the discovery process by providing high usability and effective automation. Therefore, the *Biomedical Informatics* has an increasingly crucial role, by developing the methods and tools that will allow researchers to bridge the gap between biomedical research and clinical applications.

The objective of the work described in this thesis is to address the problems mentioned above, by developing a user-friendly software tool for supporting studies aiming at linking genotypic and phenotypic information. Specifically, the goal is to develop a tool for phenotype prediction, through an approach of Bayesian networks (BNs), using genotype data (SNPs) coming from Genome-Wide Association Studies (GWAS).

BNs are a general modeling framework for knowledge representation, reasoning and probabilistic inference, and have been successfully applied in genetic analysis and association studies to study overt stroke in sickle cell anemia (Sebastiani, Ramoni, Nolan, Baldwin, & Steinberg, 2005). The complex association between phenotype and genetic factors can, in fact, be represented using a small number of parameters. However, learning these models from datasets with a high number of variables is often non-trivial.

In our work we built a model that uses an abstraction of the variable space that suitable reduces its dimensionality without losing information. We present a new strategy to achieve this goal by mapping the SNPs related to the same gene to one meta-variable, by employing an approach based on classification trees (CTs).

Outline of the thesis

This thesis is organized as follows:

- Chapter 1 describes the scientific context in which the work takes place. Then a background section is provided, reporting the biological processes underlying the developed work.
- Chapter 2 introduces the specific problem which the tool here presented aims to address: genome-wide association studies.
- Chapter 3 presents the novel approach to find a new way of explaining genotype-phenotype correlation. In this the implemented algorithms to achieve this goal are described and discussed.
- Chapter 4 describes *SNP2Net* at work: the developed tool, and the implemented options available to the user.
- Chapter 5 presents a summary of main results obtained when testing *SNP2Net* on two different datasets. We also briefly discuss these results.

1. Scientific Context

The work developed and described in this thesis addresses a specific characteristic of the post-genomic era: the correlation of genotypic and phenotypic information. The availability of modern biological high-throughput technologies and experiments is the basis of the emerging discipline of Biomedical Informatics, which may provide knowledge and tools for dealing with such an ambitious goal. In this context, the studies aimed at the so-called genetic dissection of complex traits represent a first crucial benchmark for Biomedical Informatics (Payne, Johnson, Starren, Tilson, & Dowdy, 2005) (Butte, 2008).

In this first chapter, the basic concepts of molecular biology are reported with particular focus on the biological entity containing genetic information: DNA molecules. Then the fundamental concepts of Medical Genetics are introduced, addressing the relationship between the information contained in genes and the visible phenotypic traits.

1.1. Linking genotype to phenotype

The most visible consequence of the rapid and widespread adoption of high-throughput experimental technologies is an exponential increase of the amount of data produced by each experiments, from DNA sequencing to genotyping, to gene expression analysis, to proteomics, to high-level observations on genotype/phenotype correlations. It is therefore quite clear that scientific advances will depend on being able to turn experimental data into new knowledge, by combining, transforming and interpreting them in an automated way. This is an essential requirement for *Systems Biology* (Boyle, Cavnor, Killcoyne, & Shmulevich, 2008) (Philippi & Kohler, 2006), which investigates biological processes on a large scale, relying on the measurement and analysis of thousands of variables in order to elucidate the structure and behavior of complex biological systems.

The emerging discipline of Biomedical Informatics will play an increasingly crucial role in modern biomedical research, by developing the

methods and tools that will allow researchers to fill the gap between biomedical research and clinical applications (Martin-Sanchez, et al., 2004), dealing with the ambitious goal of finding correlation between genotypic and phenotypic information (Botstein & Risch, 2003) (Lander & Stork, 1994).

1.2. Biological Background

This section introduces and defines biological terms used through the following chapters, and gives a brief explanation of the biological processes underlying the technological development discussed thereafter.

1.2.1. The genome

The cell is the structural unit of every living organism and they can be uni- or multicellular. Moreover, regarding the cell anatomy they can be prokaryotic, with no internal organelles, or eukaryotic, with a well defined nucleus. Is in the nucleus that is contained the ***genome***, a structure keeping the biological information needed to the organism living. The genome is the entire content of genetic information, stored and encoded in DNA.

DNA (*deoxyribonucleic acid*) is a polymer consisted of a pair of molecules, organized as strands held together by weak hydrogen bonds along their lengths, forming a double helical structure. Each strand is a chain of its structural unit, the ***nucleotide***. The nucleotides are differentiated by a nitrogenous base, which can be *adenine* (A), *cytosine* (C), *guanine* (G) or *thymine* (T). Hydrogen bonding occurs between laterally opposed bases, base pairs, of the two strands of the DNA duplex according to Watson-Crick rules: adenine specifically binds to thymine (A-T) and cytosine specifically binds to guanine (C-G) (Watson & Crick, 1953).

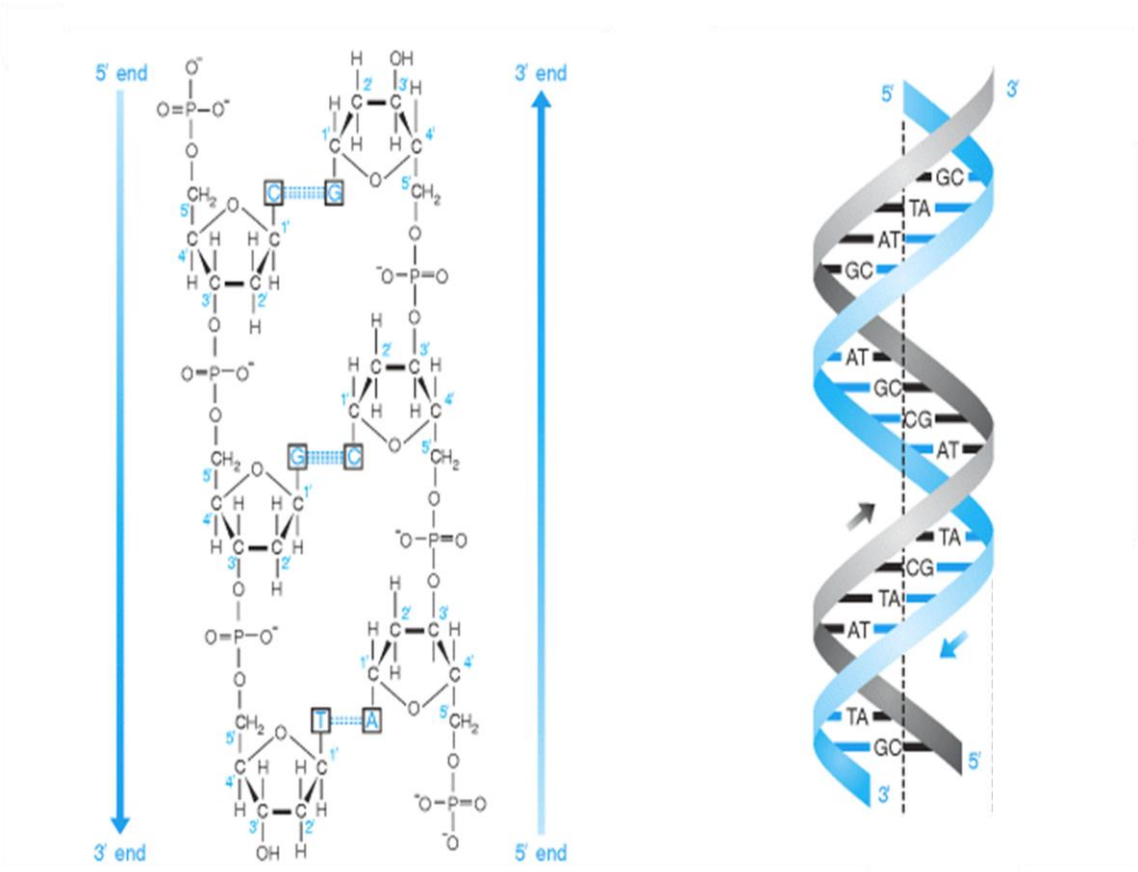

Figure 1.1 On the left the chemical structure of DNA and the antiparallel nature of the two strands. The two strands are antiparallel because they have opposite directions for linking of 3′ carbon atom to 5′ carbon atom. On the right the double helical structure of DNA (Strachan & Read, 1999).

The biological information contained in the genome is divided in main functional units, the ***genes*** (the zone between two genes is called *intergenic region*), that encode information to construct proteins, enzymes and other regulators. ***Gene expression***, or simply *expression*, is the complex multi-step process by which the information encoded into a gene is used to produce the corresponding coded protein.

1.2.2. Human genome

The human genome is almost all[1] contained in the nucleus of each cell, comprises about 3.000.000.000 base pair (***bp***) and is divided in 24 linear molecules of DNA, each one contained in one different ***chromosome***.

These chromosomes consist in 22 *autosomal* chromosomes and 2 sexual chromosomes, X and Y. Most cells are diploid, i.e. have 2 copies of each autosomal chromosome plus the 2 sexual chromosomes (XX in female and XY in male individuals), in a total of 46 chromosomes. The sexual cells, gametes, have only one copy of each autosomal chromosome and 1 sexual chromosome (23 chromosomes). The bunch of chromosomes of an individual is called ***karyotype***. Nowadays it is estimated that the human genome comprises about 80.000 genes, that consist however only 3% of the nuclear genome.

1.2.3. Genetic variability: SNP

The information for the proper living of cells (and thus of the living organisms) is contained on genes.

The specific position where a gene is located inside the genome is called ***locus***; the different possible forms of the gene are called ***alleles***. The allele makeup of an individual is its ***genotype***, while combinations of alleles at multiple loci that are transmitted together on the same chromosome are called ***haplotypes***. The expression of an organism's genes, as well as the influence of environmental factors and possible interactions between the two, result in observable characteristics of the organism, which is called the ***phenotype***.

Regarding their karyotype, human cells can be somatic diploid cells, containing two sets of homologous chromosomes (with parental and maternal origins), or haploid germinal cells (gametes), in which the process of meiosis halves the number of chromosomes. Therefore, meiosis represents the first level of genetic diversification: homologous parental

[1] A small portion of the genome (16.569 bp), called ***mitochondrial genome***, is contained in the organelles that produce energy, the mitochondria, in the form of circular molecules of DNA (about 10 copies in each mitochondria).

chromosomes of a diploid cell are subjected to recombination, so that 2 germinal cells are generated, each of which contains a chromatid[2] made of genetic materials coming from both maternal and paternal chromosomes (Figure 1.2).

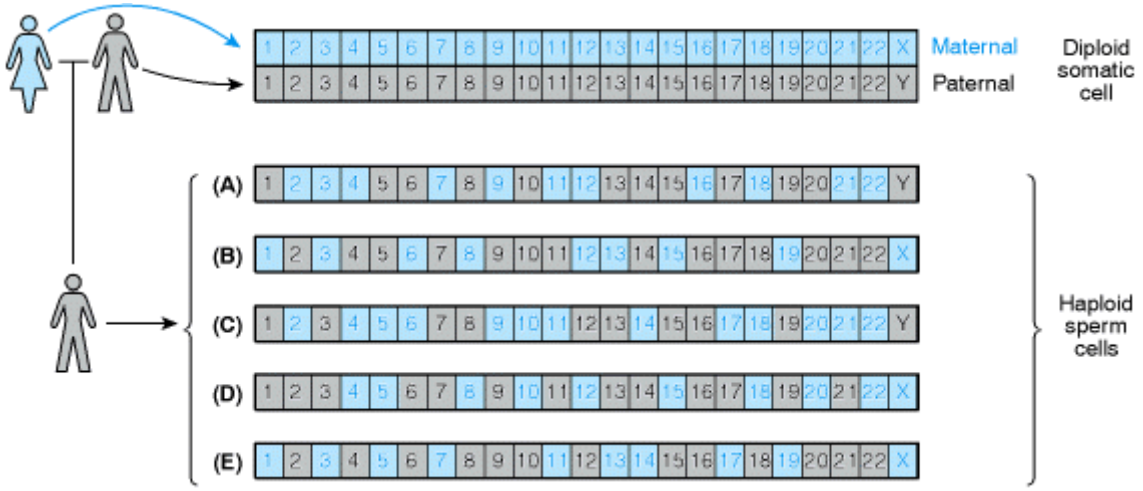

Figure 1.2 Five possible gametes among the 2^{23} possible combinations of genetic material coming from 2 homologous paternal and maternal chromosomes (Strachan & Read, 1999).

The fecundation process is the second level of genetic variability. During the fusion of the gametes, transforming the cell in a diploid cell again, there can take place the so-called *crossing-over*, i.e. the genetic material swap between maternal and paternal chromatids.

These variability features are the bases of the scientific formulation of the genetic heritability developed by Gregor Mendel (1822-1884). He observed living organisms (originally pees plants) and tried to identify the variability of the observed traits through planned mating among the individuals of the species. He was in fact measuring allele frequencies. Observing the phenotype variation over the generations, the aim was to identify the genes involved and next localize them. Even in the absence of molecular validation tools, a first rough "map" of the genome was done, with an order of magnitude comparable to genes.

[2] A *chromatid* is one among the two identical copies of DNA making up a replicated chromosome; in other words, a chromatid is "one-half of a replicated chromosome".

Then, during the XX century, the genome has been mapped with higher and higher resolution. In particular, the relation between genes mutations and some diseases have been found, as well as genes locations (Botstein & Risch, 2003). In 2000, the Human Genome Project made possible to decode the whole human genome (IHGSC, 2001) and provided a fine mapping at the level of single nucleotides, so that new genetic markers became available, in particular also within intergenic regions.

Those new markers are called *Single Nucleotide Polymorphisms*, ***SNP*** (*"snip"*), and are natural variations affecting single nucleotides. SNPs are the most part of all variation elements in almost all genomes. In classical population genetics, loci are considered to be polymorphic if the frequency of the most common allele (a.k.a. *wild-type*) is lower than a threshold, commonly assumed to be 1% (Web 1).

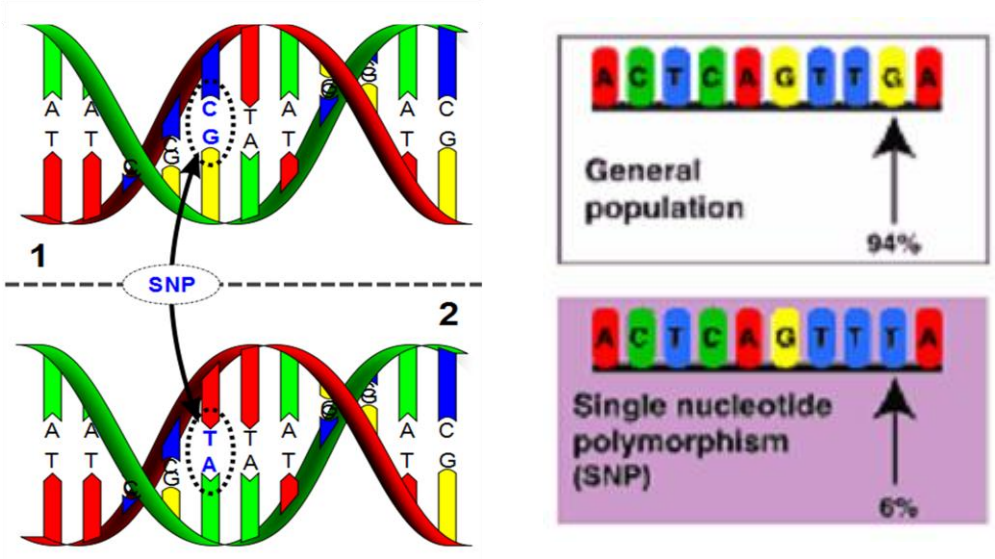

Figure 1.3 A Single Nucleotide Polymorphism: SNP (Web 2).

SNPs are classified according to the nucleotide type and function[3]. If SNPs are located on the 5' or 3' regions, which are neither transcript (NTR) or traduced (UTR), within an intron or in intergenic regions they are called *non coding*. On the contrary, if they cause an amino acid substitution (and

[3] SNPs are divided in *transitions*, in which either a purin (A,G) became another purin or a piramidin (C,T) became another piramidin, and *transversions*, in which a purin became a piramidin and vice versa.

then the protein translation) they are called *coding* SNPs. Or they may be *synonymous* SNPs (the transcript codon changes, not the amino acid). The two last ones may also alter gene functions by affecting regulatory mechanisms and they are particularly interesting because most of the SNPs are non coding and also because of the serious changes they cause on the normal behavior of many genes.

SNPs may allow the positional identification of loci involved in polygenic variation of the phenotype, the so-called *quantitative trait loci* (QTL). However, most of QTL effects may be due to SNP which have not yet been sequenced. In fact, the identification of significant variations in SNPs distributions between two groups of individuals showing different phenotypic variations is the aim of the association studies, which we will describe later in this chapter.

1.2.4. Genotyping microarrays

Genotyping provides a measurement of the genetic variation between members of a species. The challenge of the applications developed to detect SNPs, by hybridizing complementary DNA probes to the SNP site, is to reduce cross-hybridization between the allele-specific probes, what can be actually overcome by manipulating the hybridization stringency conditions (Rapley & Harbron, 2004).

Hundreds of thousands of probes are arrayed on a small chip, allowing for many SNPs to be interrogated simultaneously, in high density (HD) oligonucleotide SNP arrays (Rapley & Harbron, 2004). Several redundant probes are used to detect each SNP, as a way to somewhat overcome the target DNA potential to hybridize to mismatched probes. In fact, this is due to the fact that SNP alleles only differ in one nucleotide and because it is difficult to achieve optimal hybridization conditions for all probes on the array. This approach consists in having the SNP site in several different locations as well as containing mismatches to the SNP allele. Then, it is possible to determine specific homozygous and heterozygous alleles by comparing the differential amount of hybridization of the target DNA to each of these redundant probes (Rapley & Harbron,

2004). Although oligonucleotide microarrays have a comparatively lower specificity and sensitivity, the scale of SNPs that can be interrogated is a major benefit.

There are two main DNA arrays available today: those produced by Illumina (http://www.illumina.com) and those produced by Affymetrix (http://www.affymetrix.com). Illumina recently proposed new technologies with which more than 1.1 million markers can be sequenced per sample, with a median marker spacing of 1.5 kilo base (kb), by using a new generation of multi-sample Infinium HD products that support integrated SNP and copy number variation (CNV[4]) analysis. The Affymetrix Genome-Wide Human SNP Array 6.0 is a single array that features more than 1.8 million markers for genetic variation, including more than 906.600 SNPs and more than 946.000 additional probes for the detection of CNVs.

[4] A *copy number variation* (CNV) is a segment of DNA in which copy-number differences have been found by comparison of two or more genomes. The segment may range from one kilo base to several mega bases in size (Iafrate, et al., 2004) (Sebat, et al., 2004).

2. Genome-Wide Association Studies

The research of genetic causes for many complex diseases requires the use of innovative and powerful strategies: in this chapter we introduce and describe alternative approaches to the classic Mendelian genetics, represented by the Genome-Wide Association Studies. After a background section on this kind of studies, we present graphical models to formalize the relations between genotype and phenotype, widely used in Bayesian networks as an application of statistical association evidence research.

2.1. From Mendelian genetics to *complex traits*

The work of Gregor Mendel signed the beginning of the *quantitative biology*, resulting in the ***classic genetics***. He tried to understand the biological causes of the variations in the morphology of pees plants when mating them over several generations and proposed the existence of the nowadays called genes and built two laws:

- The *Law of Segregation* states that when any individual produces gametes, the copies of a gene separate, so that each gamete receives only one copy. A gamete will receive one allele or the other. So, for each individual, the gene coding for a characteristic is present as a couple of alleles, one inherited from the father and the other from the mother; the allele that manifests in phenotype is the ***dominant allele***, while the other is the ***recessive allele***.
- The *Law of Independent Assortment*, also known as *"Inheritance Law"*, states that alleles of different genes assort independently of one another during gamete formation.

This approach led to better and better comprehend the inheritance mechanisms, but it could not be applied to the Man species, because studies in human genome were not available yet. The discovery of new genetic variability elements enabled to come up with a methodology of inheritance

analysis in Man: the ***linkage analysis***. Genetic linkage occurs when particular genetic loci or alleles for genes are inherited jointly, having a co-segregation more frequent than they would have if they segregated in a completely casual way.

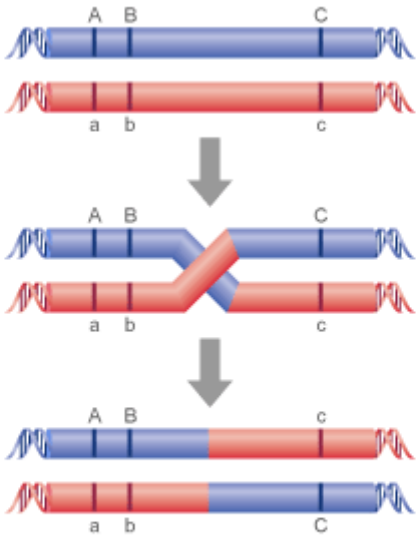

Figure 2.1 Principle of linkage analysis. The top diagram shows paternal (blue) and maternal (red) chromosomes aligned in a germ cell, a cell that gives rise to eggs or sperm. Three DNA sequences are shown, labeled A, B and C. The capital letters represent the paternal alleles and the lower case letters represent the maternal alleles. The middle panel shows the physical process of recombination, which involves crossing over of DNA strands between the paired chromosomes. The bottom panel shows what happens when the crossover is resolved. The maternal and paternal alleles are mixed (recombined) and these mixed chromosomes are passed to the sperms or eggs. If A is the disease gene and B and C are genetic markers, recombination is likely to occur much more frequently between A and C than it is between A and B. This allows the disease gene to be mapped relative to the markers B and C (Web 3).

The principle of linkage analysis is simple. All our chromosomes come in pairs, one inherited from our mother and one from our father. Each pair of chromosomes contains the same genes in the same order, but the sequences are not identical. This means it should be easy to find out whether a particular sequence comes from our mother or father. The linkage analysis indicates how much the distribution of the allelic frequencies departs from the second Mendelian law. This distribution is

reconstructed by positioning certain genetic traits along a family tree or *pedigree*: it is like reconstructing an experiment of laboratorial cross without the need of really perform it.

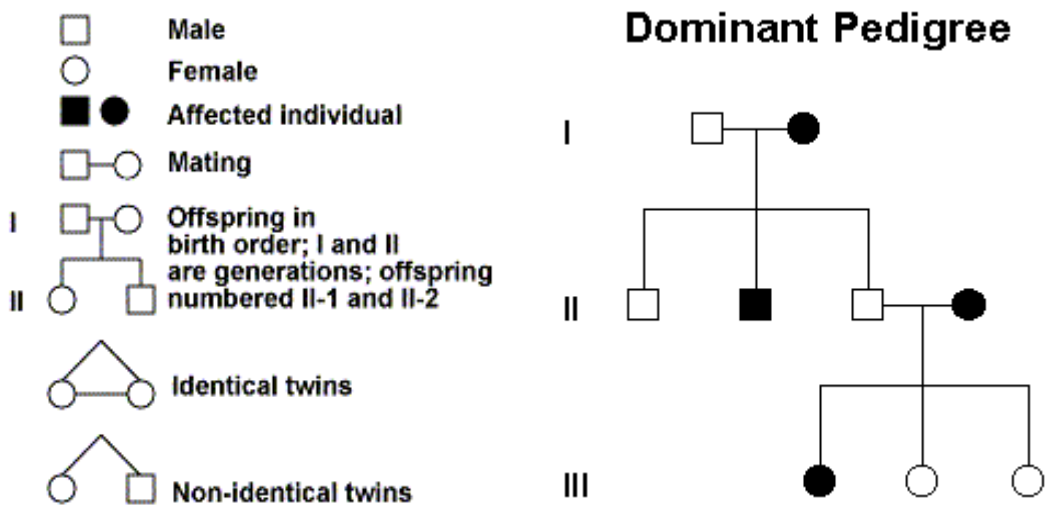

Figure 2.2 Principal symbols used when drawing a pedigree. Representation of the pedigree of a trait controlled by dominant gene action (Web 4).

This ability to determine the parental origin of a DNA sequence allows us to show whether recombination has taken place. As recombination occurs more or less at random, if there is a large distance between two DNA sequences on a chromosome, there is a good chance that recombination will occur between them and the maternal and paternal alleles will be mixed up (see A and C in Figure 2.1). In contrast, if two DNA sequences are very close together, they will recombine only rarely. The maternal and paternal alleles will tend to stay together (see A and B in Figure 2.1).

Disease genes are mapped by measuring recombination against a panel of different markers spread over the entire genome. In most cases, recombination will occur frequently, indicating that the disease gene and marker are far apart. Some markers however, due to their proximity, will tend not to recombine with the disease gene and these are said to be linked to it. Ideally, close markers are identified to flank the disease gene and define a candidate region of the genome between 1 and 5 million bp in

length. The gene responsible for the disease lies somewhere in this region (Web 3).

This kind of analysis doesn't produce however evidence of association between genotype and phenotype in the cases of rare diseases, which small amount of available data weakens the statistical validation of the used method, or in the cases of non Mendelian diseases, those that do not present dominant or recessive inheritance attributable to a single locus. These diseases are defined as ***complex traits***. In these cases it is not enough to perform linkage analysis, because the variations observed and measured are of the order of genes.

Therefore, we next discuss the association studies as a way to overcome this problem.

2.2. Genetic Association Studies

Association studies (AS) aim to find statistically significant differences in the distribution of a set of markers (the frequency of SNP alleles or genotypes) between a group of individuals showing a trait of interest (the *cases*) and a group of unrelated individuals who do not exhibit the trait (the *controls*) (Balding, 2006) (Cordell & Clayton, 2005). The fact that the presence of the SNP may enhance the risk of disease is indicated by an increase of the frequency of a SNP genotype or allele in cases compared with controls. However, it is not trivial to consider a genetic difference between cases and controls as directly linked to the disease of interest and not to other factors. One fundamental problem is that the genome is so large that patterns that are suggestive of a causal polymorphism could well arise by chance. In fact, as represented in Figure 2.3Figure 2.3, association may arise due to three main reasons:

- The association is mono-genic and it is verifiable all over the population because the target allele is the cause of the disease (Mendelian disease).

- The target allele is in *linkage disequilibrium*[5]*(LD)* with the causal variant but it is not the direct cause of the disease.
- The difference in frequencies due to the presence of subgroups of different ethnical origin (population stratification) can lead to fake association.

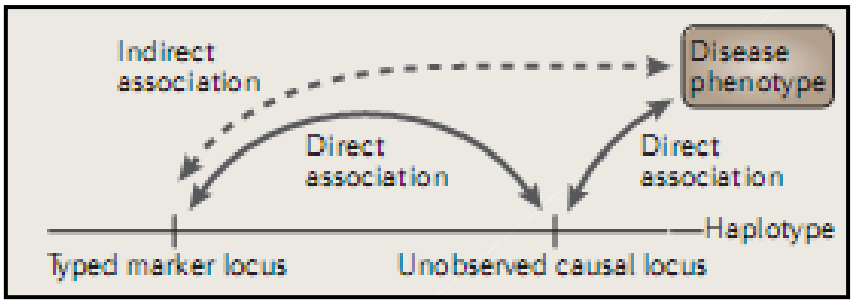

Figure 2.3 Different type of association (Balding, 2006).

Thus, to help distinguish causal from spurious signals, tight standards for statistical significance need to be established. The next sections address the tests of association, based on single-SNPs and haplotypes.

2.2.1. Single-SNP association

Because of the computational problems in analyzing such large datasets as those used in GWA studies, single-SNP tests remain the primary statistical tool used for this kind of analysis. In case-control studies, the most natural analysis of SNP genotypes and case–control status is to test the null hypothesis of no association on the data consisting of counts of the number of alleles (allele A and B) or genotypes (the two homozygotes and the heterozygote, AA, BB and AB) among cases and controls. The resulting 2x2 or 2x3 contingency tables can be directly analyzed using, among others, a Pearson chi-square test (with 1 or 2 degrees of freedom respectively for the allelic or genotypic association tests):

[5] The *Linkage Disequilibrium* is the statistical association within gametes in a population, of the alleles at two loci. It can be due to linkage, but it can also arise at unlinked loci; for example, because of selection or non-random mating.

$$\chi^2 = \sum_{i=1}^{g} \frac{(n_i - E_i)^2}{E_i} = \sum_{i=1}^{g} \frac{n_i^2}{E_i} - n$$

where n_i is the number of cases for the i-th state, E_i is the expected number of cases if H_0 is true (null hypothesis), and g is the number of possible states of the categorical variable to be tested.

If we perform association tests on a set of SNPs covering the whole genome, we are performing a *Genome-Wide Association Study*, which is particularly helpful when there is little or no a priori information about the location of the genetic cause of the phenotype of interest.

While in the case of association of a single-SNP the significance level α ensures the test significance, in GWAS it could lead to mistaken results, since hundreds of thousands of SNPs are tested at the same time. Even though each SNP is analyzed independently, some recognition of multiple testing is necessary. Considering, for example, a GWAS using a 500K array (500.000 SNPs) and the standard 5% level of significance for biomedical tests, we would have $500.000 \times 0.05 = 25.000$ SNPs wrongly associated to the trait. A generally used strategy to overcome this problem is to use the *Bonferroni correction*, defining the appropriate significance level α' for each single test as follows:

$$\alpha = 1 - (1 - \alpha')^n \rightarrow \alpha' \approx \frac{\alpha}{n}$$

where n is the number of considered SNPs. By applying this kind of correction we obtain a re-scaled significance level α' ($\alpha' = 0.05/500.000 = 1*10^{-7}$) to be applied to each single test of a 500K SNPs GWAS dataset.

Although the power of GWAS is often low, they represent useful instruments to pinpoint areas of the genome that may contain candidate genes, and to guide a subsequent, more targeted analysis step. Since the number of known, well characterized genetic markers in the human genome

is increasing, the genome-wide approach represents an increasingly cost effective way of generating testable research hypotheses.

2.2.2. Multilocus analysis: haplotypes association

As seen before, association studies commonly test a series of SNPs within a candidate gene, one at a time, so that each marker is analyzed independently. These tests can suffer from problems that are associated with many predictors. Despite of the Bonferroni correction application to overcome the multiple testing issue, the results from each test may not be independent if the tested SNPs are in linkage disequilibrium.

So, analysis methods based on single-SNPs have limited power to detect a true genetic effect that requires a specific allele at several SNPs, that may be detected with a strategy suggested by the block-like structure of the human genome, using haplotype-based methods. This approach tries to capture the correlation structure of SNPs in regions of little recombination by analyzing all SNPs concurrently.

An immediate problem is that haplotypes are not observed; they must be inferred instead, leading to some uncertainty when assessing the significance of the tests.

The simplest analysis on haplotypes is to test for independence in a 2 x k contingency table, where k is the number of distinct haplotypes, using the described techniques.

Several problems still exist when performing haplotype-based analyses. What should be done about rare haplotypes? Including them in analyses can lead to loss of power because there are too many degrees of freedom. How should similar but distinct haplotypes that might share recent ancestry be accounted for? Both might carry the same disease-predisposing variant but simple analyses will not consider their effects jointly and might miss the separate effects. Another problem with defining haplotypes is that block boundaries can vary according to the population sampled, sample size, SNP density and block definition. Often some recombination events can occur within a block, and conversely there can be between-block LD that will not be exploited by a block-based analysis.

Although haplotype analysis seems to be a natural approach, it might ultimately confer little or no advantage over analyses of multipoint SNP genotypes.

2.2.3. Multivariate approaches: proposed method

The methods described above do not include interaction analysis and other similar approaches capable to grasp the effect of interactions and across-genome combinations, rather than the major effect of single-SNPs or (despite more importantly) the major contribution of a specific haplotype in a locus.

So, a suitable alternative to the use of both single-SNP and haplotype-based association may be the use of a multivariate approach that takes into account the experience of gene annotation. This approach tries to exploit some biological knowledge, by taking into account the SNPs annotation to the corresponding gene and use genes as markers.

In the next chapter we describe the details of the proposed method and we also present some encouraging results about the utility of the approach.

3. The proposed method: gene-based networks

As seen before, genetic association studies are a powerful method to assess correlations between SNPs and traits differences occurring in a population. Once high-throughput technologies (section 1.2.4) allow performing these studies at a genome wide level, it is possible to take into account hundreds of thousands of different SNPs (Hirschhorn & Daly, 2005) but learning predictive models from such huge sets of data is often non-trivial due to the high number of variables (McCarthy, et al., 2008), with respect to the instances of the dataset. So, together with representing complex associations between phenotype and SNPs, finding the relations between the SNPs themselves can be extremely interesting and permits to highlight the genes associated to the phenotype allowing an abstraction of the variable space that suitably reduces its dimensionality without losing information. In this way, it is possible to identify how genes affect the phenotype by building a more parsimonious model with more easily interpretable connections between variables.

Therefore, the basic strategy applied is made of the following main steps:

1. Candidate SNPs and candidate genes selection, from genetic association studies.
2. Mapping SNPs to genes (positional annotation) and generation of meta-variables (each one representing a gene). Assignment of values to genes in order to use them as predictors by using an approach based on classification trees for the set of SNPs mapping to the same gene and this way identifying the most relevant combination of SNPs values to predict the phenotypic status.
3. Learning gene-based Bayesian networks for phenotype prediction, in which the nodes are the genes and the phenotypic status. Every gene has a number of states which is a minimal subset of the Cartesian product of the values of each gene SNPs.

Aiming to predict the phenotype of the individuals, the model learned from data is then used to do inference and estimate the probability of a phenotypic trait given the genotype of an individual, represented as a

suitable collection of attributes (SNPs and genes). The classification accuracy (CA) of the BN inferred on genes is quantitatively assessed and compared with that achievable with a BN built using single-SNPs for the same data (Malovini, Nuzzo, Ferrazzi, Puca, & Bellazzi, 2009).

This approach has already been applied on arterial hypertension (AH) and led to the published paper *Phenotype forecasting with SNPs data through gene-based Bayesian networks* (Malovini, Nuzzo, Ferrazzi, Puca, & Bellazzi, 2009), in which it was shown that in fact meta-variables can really summarize the relationships between genes and phenotype without losing information. This way, using genes as predictors for the phenotype can be a suitable alternative to haplotypes, which are on the contrary frequently used also as prediction factors.

So, in this chapter we describe the main steps of the approach, from the SNPs and genes selection, through the Classification Tree learning on genes to the Bayesian network learning on both SNPs and genes datasets. Finally, we point the limitations of the approach and the specific objectives of the work developed in this thesis.

3.1. SNPs and genes selection

Using *PLINK* software (Web 5) it is possible to perform some standard preliminary analyses by calculating: i) genotyping/missing rate statistics (removal of samples and SNPs with missing rate > 10%); ii) minor allele frequency (MAF) (removal of SNPs with MAF < 1%); iii) Hardy Weinberg Equilibrium (removal of SNPs deviating from the Hardy Weinberg Equilibrium (p-value < 0.001) in the control population). In order to identify and remove genetic outliers, we also perform a multidimensional scaling plot (MDS plot) and a neighbours plot, based on the genome-wide identity-by-state (IBS) information. After this pre-processing phase, we perform both allelic and genotypic association tests (see section 2) to compare frequencies distribution between cases and controls in order to select only top-significant SNPs after the selection of an appropriate significance threshold ($p<1*10^{-4}$, corrected with permutation tests). To perform a positional annotation on the top associated markers we use *Genephony*, an

online tool for genomic dataset annotation (Web 6): one SNP is annotated to a certain gene if it is located in a 10Kb region around the coding sequence. After performing all the association tests we export the genotypes corresponding to the selected top-ranked SNPs and get, in output, two files: the PED file (.ped file) and the MAP file (.map file). The PED file has six descriptive columns: Family ID, Individual ID, Paternal ID, Maternal ID, Sex and a Phenotype column, a categorical affection status like "non-affected" or "affected" (values 1 and 2 respectively) for a certain disease and genotypes corresponding to the SNPs. From column 7th onwards we have genotypes corresponding to the SNPs, coded as A, C, G or T. By default, the missing genotype character is 0. An important note is that all SNPs have two alleles specified or both alleles are missing (i.e. 0 0). Concerning the MAP file, each row describes a single-SNP and contains exactly four columns: chromosome (coded in 1 to 22, X, Y or 0 if unplaced), SNP identifier, genetic distance (genetic distance is a measure of the dissimilarity of genetic material between individuals, presented in centimorgan[6]) and base-pair position (bp units). Each row in the MAP file corresponds to one column on the PED file. We use the MAP file as the input for *Genephony* and in the output we get the genes annotation (annotated MAP file) containing both SNPs and genes identification.

Family ID	Individual ID	Father ID	Mother ID	Sex	Phenotype	SNP A	SNP B	SNP C	SNP D	SNP E
ipert10001c	ipert10001c	0	0	2	1	1	3	2	2	3
ipert10002c	ipert10002c	0	0	2	1	1	1	3	1	3
ipert10003c	ipert10003c	0	0	1	2	3	2	2	1	1
ipert10004c	ipert10004c	0	0	1	1	2	2	3	1	2
ipert10006c	ipert10006c	0	0	2	2	1	2	1	2	1

Figure 3.1 Representation of the PED file.

[6] In genetics, a centimorgan (abbreviated cM) or map unit (m.u.) is a unit of recombinant frequency for measuring genetic linkage. It is often used to imply distance along a chromosome. The number of base-pairs it corresponds to varies widely across the genome (different regions of a chromosome have different propensities towards crossover). One centimorgan corresponds to about 1 million base pairs in humans on average (Scott, et al., 2004).

Chromosome	SNP ID	Genetic distance	Base-pair position
1	SNP A	0	46853252
1	SNP B	0	46853328
1	SNP C	0	111571946
1	SNP D	0	111597206
2	SNP E	0	46247319

Figure 3.2 Representation of the non-annotated MAP file.

Chromosome	SNP ID	Genetic distance	Base-pair position	Gene ID
1	SNP A	0	46853252	Gene 1
1	SNP B	0	46853328	Gene 2
1	SNP C	0	111571946	Gene 1
1	SNP D	0	111597206	Gene 1
2	SNP E	0	46247319	Gene 2

Figure 3.3 Representation of the annotated MAP file.

Note that the grey header lines in both files don't exist in the real PED and MAP files coming from *PLINK* (Web 5) and *Genephony* (Web 6). We merge this PED and MAP files in one single file, eliminating the unnecessary columns and so creating our original SNPs dataset, containing *n* examples in rows, the phenotype *c* and *v* SNP genotypes in columns. The final SNPs dataset is represented in Figure 3.4.

Original dataset: SNPs

Status	Gene 1 SNP A	Gene 2 SNP B	Gene 1 SNP C	Gene 1 SNP D	Gene 2 SNP E
1	a_1	b_3	c_2	d_2	e_3
1	a_1	b_1	c_3	d_1	e_3
2	a_3	b_2	c_2	d_1	e_1
1	a_2	b_2	c_3	d_1	e_2
2	a_1	b_2	c_1	d_2	e_1

Figure 3.4 Final SNPs dataset, with the genotypes and the genes annotation.

3.2. Meta-variables generation: Classification Trees

Having selected our dataset with n examples, a class c and v variables (as many as the number of selected SNPs), we want to convert it into a new meta-dataset containing the same number of examples n, the same class c but only m new meta-variables (as many as the number of selected genes). Each meta-variable (gene) shall contain, for each example, the information from all gene mapping SNPs values for that example.

Meta-dataset: Genes

Status	Gene 1	Gene 2
1	g_2	f_3
1	g_4	f_3
2	g_3	f_1
1	g_4	f_2
2	g_1	f_1

Figure 3.5 Meta-dataset with one column for each gene present in the SNPs dataset.

We do this by learning a Classification Tree (CT), one of the most widely used classification tools (Mitchell, 1997).

By definition, classification is the task of learning a target function *f* that maps an input attribute set x into its predefined class value c (Tan, Steinbach, & Kumar, 2006).

The learned classification model can be either predictive or descriptive. Descriptive models are those able of distinguish examples of different classes and explain what combination of features values define each class. Predictive models aim at predicting the class of unknown examples. In fact, our input is a collection of SNPs and we want to both describe data and predict unknown genotype values. The learning algorithm is employed to identify a model that best fits the relations between the attribute set and the class of the input data and its main goal is to build a model with good generalization capability (Tan, Steinbach, & Kumar, 2006).

There are three different types of nodes in the classification tree:

- The **root node** that has no incoming edges and zero or more outgoing edges.
- Internal **non-leaf nodes**, each of which has exactly one incoming edge and two or more outgoing edges.
- **Leaf nodes**, which have exactly one incoming edge and no outgoing edges.

Accordingly, each SNP in the dataset is represented by a non-leaf or interior node, which is associated with a test on the value (genotype) of that SNP; at the beginning of the process there is only the root node, containing the original set, with all examples and all SNPs. Each leaf node is assigned a class value (phenotype). So, classifying a test example is straightforward once a classification tree has been built. Starting from the root node we apply the test condition on each node to the example and follow the appropriate branch based on the outcome of the test, leading us either to another internal node, for which a new test condition is applied, or to a leaf node. Thereby, the path from the root to the final leaves give us the combination of the SNPs values that we expect to be able to correctly predict the phenotype for each example.

3.2.1. C4.5 algorithm

There are different available algorithms to learn a classification tree from a dataset. Recursive partitioning enables to create a classification tree that intends to correctly classify examples of the population based on several dichotomous dependent variables. The method presents several advantages: it provides a simple and intuitive method for classification, it may identify nonlinear relationships and it considers prior probabilities and penalties for misclassification during its attribute selection process.

In applying this strategy, we use an implementation of the C4.5 algorithm (Quinlan, 1993) (Witten & Frank, 2000) to learn a classification tree for each gene, separately. We recursively split the source set in each node into successively purer subsets based on the ***most informative*** SNP value test, i.e. the SNP that better separates instances with respect to their class value. The algorithm provides a method for specifying the SNP splitting condition test, discussed later in this section.

If, in one node, the most informative SNP has three possible values (*1*, *2* and *3*) the algorithm will create three child nodes, each one containing the examples in which the splitting SNP (corresponding to the parent node) has the value *1*, *2* or *3* respectively. In the new node, the most informative SNP for the "parent set" is no longer present and the new most informative SNP is chosen among the other SNPs of the "child set".

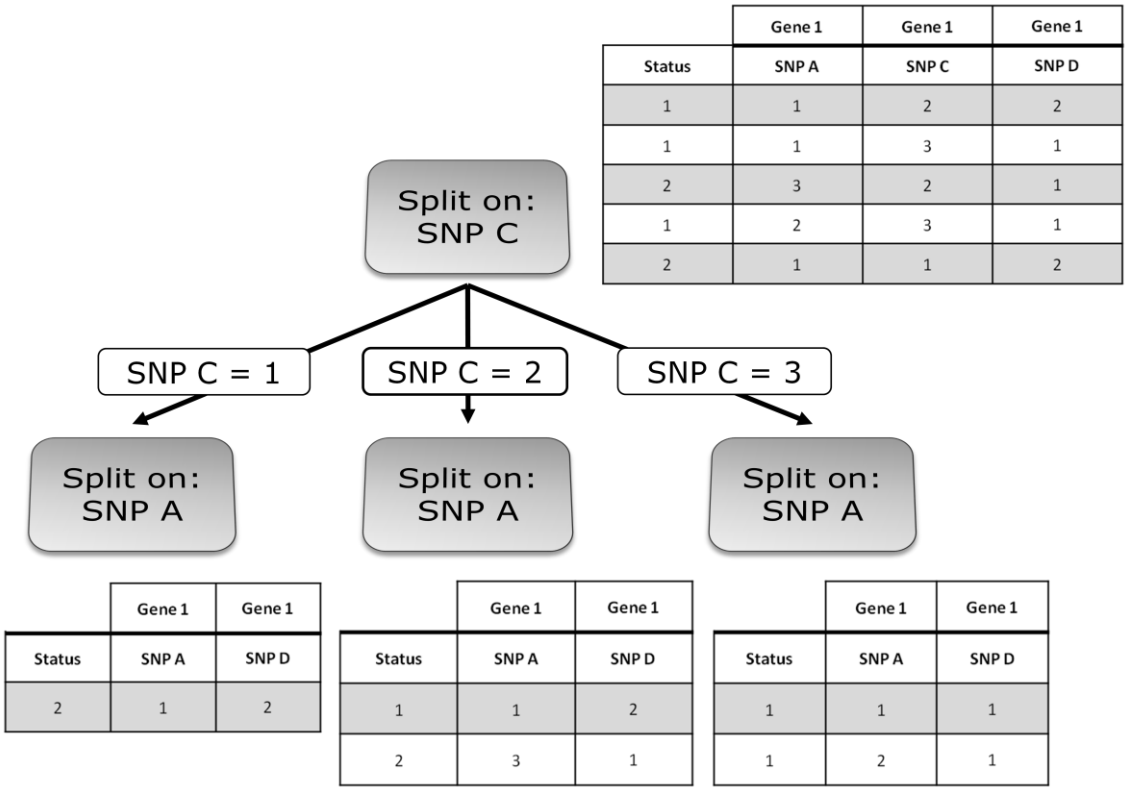

Figure 3.6 Example of recursive partitioning, strategy used in the C4.5 algorithm.

So, for every possible value of that SNP, there is a branch descending to a child, corresponding to a logical conjunction (*AND*). It is also possible to use disjunctions (*OR*) to join two or more paths together, as long as they lead to the same classification result.

This process is repeated on each derived subset until the recursion is completed: when splitting is either non-feasible (all the examples in the node have identical SNPs values) or the same classification can be applied to each example of the derived subset (if all the examples in one node have the same class value regardless of the SNPs values, it is possible to split the data again but it will be useless because the classification cannot be improved). On both cases, the node is declared a leaf node with the same class value as the majority class of the examples associated with this node.

Thus, a leaf node represents the phenotype given the values of the SNPs represented by the path from the root and is assigned a class value. Therefore the path going from the root node to a leaf node can be described with a classification rule of the kind "*IF* SNP C2 = BB *AND* (SNP C1 = Aa *OR*

SNP C1 = aa) *THEN* Phenotype = affected" that allow the classification of each example.

In a more intuitive way for health care providers, we can say that recursive partition creates a rule such as *"If a patient has findings x, y, or z then probably he/she has disease q"*.

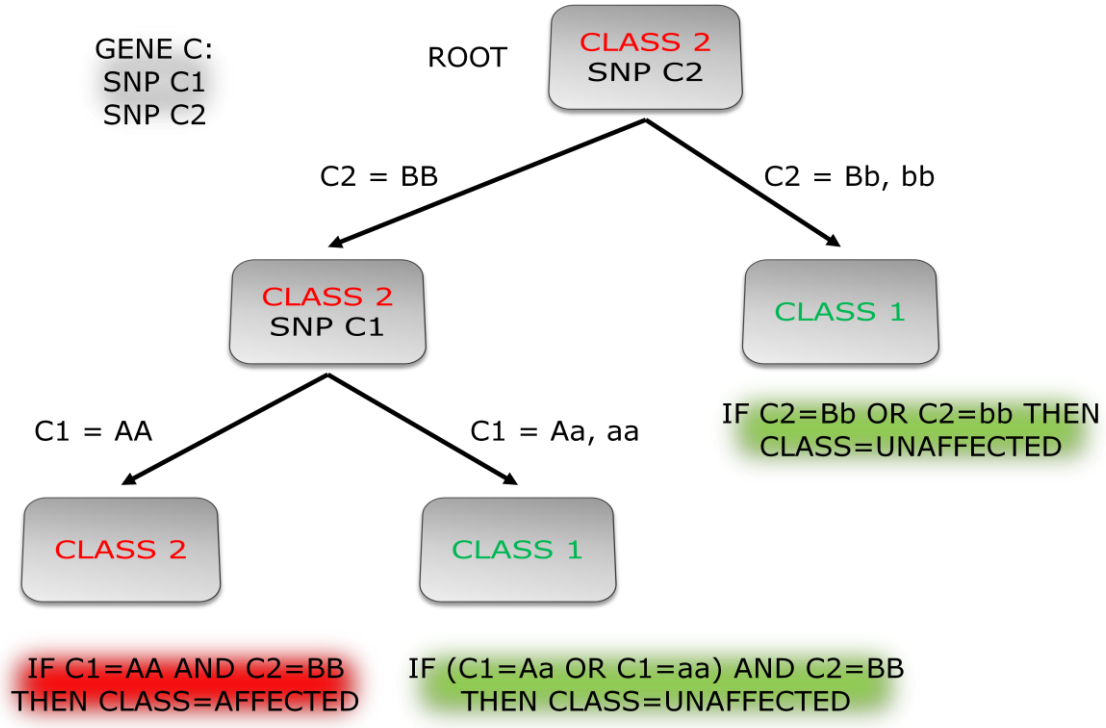

Figure 3.7 Example of a classification tree obtained by recursive partition. The path from the ROOT to a leaf node creates a rule IF … THEN.

CTs can be very large and complex; when complexity increases they are susceptible to a phenomenon known as overfitting. To soften the data overfitting we use a couple of pruning techniques, some of them provided by the implementation of the algorithm. Pruning helps by trimming the branches of the initial tree in a way that improves the generalization capability of the classification tree (Tan, Steinbach, & Kumar, 2006). Overfitting and pruning are discussed later, in section 3.3.

Choosing the best split

There are many measures that can be used to determine the best way to split the examples set in one node of the classification tree, where all of them are based on the class distribution of the examples before and after splitting: entropy, Gini index or classification error.

Regarding the information theory, some information, I, is needed to classify an object. Once we know the value of the attribute, we only need a certain amount of residual information to perform the classification, $Ires$. So, the most informative attribute is that minimizing the value of $Ires$, i.e. maximizing the Gain of the attribute.

$$\boldsymbol{Gain\ (attribute) = I(class) - Ires\ (attribute)} \qquad \textbf{Equation 3.1}$$

Given an example with P the probability of occurrence, the associated information is:

$$\boldsymbol{I = -\log_2(P).} \qquad \textbf{Equation 3.2}$$

The mean amount of information (in *bits*) necessary to classify an object (assigning one of c values for the class) is given by the measure of the object entropy:

$$\boldsymbol{I(class) = -\sum_c P(c)\ \log_2 P(c)} \qquad \textbf{Equation 3.3}$$

After using the attribute A, the examples set S is divided in subsets according to the values of A, v. The residual information of the attribute, $Ires(A)$ is the weighted sum of the information I in each subset.

$$\boldsymbol{I_{res}(attribute) = -\sum_v P(v) \sum_c P(c|v) \log_2 P(c|v)} \qquad \textbf{Equation 3.4}$$

$$\boldsymbol{Gain\ (attribute) = I(class) - Ires\ (attribute)} \qquad \textbf{Equation 3.5}$$

The most informative attribute is chosen as the one having the higher Gain value.

However, impurity measures such as entropy (the same for Gini index) tend to favor attributes that have a large number of distinct values. These attributes divide the set *S* in many subsets, which is not desirable because the number of examples associated with each partition is too small to enable us to make a reliable prediction. A possible solution is the one applied in the C4.5 algorithm that is based on a splitting criterion known as **gain ratio** or normalized information gain (difference in entropy) to determine the goodness of a split.

The amount of information necessary to determine the value of an attribute is:

$$\boldsymbol{I(attribute) = -\sum_v P(v)\ \log_2 P(v)\,.} \qquad \textbf{Equation 3.6}$$

And so, the information gain ratio is defined as following:

$$\boldsymbol{Gain\ Ratio\ (attribute) = \frac{Gain\ (attribute)}{I\ (attribute)} = \frac{I\ (class) - I_{res}\ (attribute)}{I\ (attribute)}} \qquad \textbf{Equation 3.7}$$

With this corrected measure, if an attribute produces a large number of splits, its split information will also be large, which in turn reduces its gain ratio.

3.3. Model overfitting and pruning

A classification model can produce training errors and generalization errors. Training errors represent the number of misclassification errors on training examples, i.e. in those examples used to learn the model. On the other hand, generalization errors are those expected of the model on

previously unseen examples, i.e. new data. A model that fits the training data too well can have a poorer generalization error than a model with a higher training error, incurring in the so called overfitting phenomenon.

Usually, increasing the model complexity reduces the training error of the model as, for example, the leaf nodes of the tree can be expanded until perfectly fitting the training data. However, in this case the generalization error can be large because the tree may contain nodes that accidentally cover some noise points in the training data, and so degrading the tree performance. Thereby, the overfitting phenomenon may be due either to the presence of noise, i.e. misclassified examples in the training data, or to lack of representative samples, i.e. classification models based on a small number of training examples (Tan, Steinbach, & Kumar, 2006). As the model does not have access to the test set before its construction, it does not know how well the tree will perform on unseen examples.

Nevertheless, as said before, overfitting of data can be avoided using some pruning strategies, improving the tree's classification ability on new data.

Pre-pruning approaches stop the tree-growing algorithm before generating a fully grown tree that perfectly fits the entire data. This can be done by applying a threshold to the gain of the impurity measure, below which the node is not expanded and it remains a leaf node. Even though overly complex subtrees can be avoided, it is difficult to choose the right threshold. High thresholds can result in myopic models that do not describe well the data, while too low thresholds may not overcome the overfitting problem. As so, we choose not to apply any pre-pruning but only post-pruning strategies.

With a post-pruning approach the classification tree initially grows to its maximum size. Then, we apply a bottom-up trimming. If a leaf node or subtree leads to a negligible improvement in the classification performance it can be discarded, making the tree smaller. When a leaf is removed, either the parent node turn into a leaf node containing all the examples present in the removed leaf if there weren't more leaves on that node, or the remaining leaves are also checked for pruning, according to previously defined pruning criteria. The new leaves are assigned the majority class of the examples affiliated with the subtree or lower level leaf. The tree-pruning

terminates when no further improvements are observed. So, in order to prune the tree we apply the following criteria: minimal error pruning strategy, minimum number of instances in leaves and maximum number of states for each gene.

3.3.1. Minimal Error Pruning

Minimal Error Pruning (MEP) is a bottom-up pruning strategy that estimates the error on new data directly from the growing-set (the data used for building the tree) and so it doesn't need a pruning set like other strategies. MEP uses Bayesian method for probability estimation (either Laplace or *m*-estimate) and it prunes the tree so that estimated classification error is minimal (Niblett & Bratko, 1986).

If $E(T)$ is the error of the tree *T* we have to decide whether to prune or not in each node *v* with subtrees T_1, T_2 and T_3.

First we define the static error at node v

$$\boldsymbol{e(v) = P(class \neq c|v)}$$

Equation 3.8

where c is the most likely (preferred) class at v. If *T* is pruned at v then

$$\boldsymbol{E(T) = e(v).}$$

Equation 3.9

If *T* is not pruned at v than its (backed-up) error is

$$\boldsymbol{E(T) = P_1\, E(T_1) + P\, E(T_2) + P\, E(T_3).}$$

Equation 3.10

So we prune *T* on node v if the static error on v is equal or less than the backed-up error on the subtree, which means the subtree does not improve the performance of *T*. If the tree is pruned in v, the node turns into

a leaf with the class assigned being the majority class on the node itself. Accordingly, we define the error $E(T)$ as:

$$\boldsymbol{E\ (T) = \min\ (e(v), \sum_i\ P_i\ E\ (T_i))} \qquad \textbf{Equation 3.11}$$

We use *m*-estimate of probability to estimate static errors $e(v)$ on nodes.

At node v we have *N* examples, k possible values for the class and n_m the number of examples associated to the non-majority class (these examples are misclassified). Then, using *m*-estimate we can compute the static error on node v as

$$\boldsymbol{e_v = \frac{p_{ca} \times m + n_m}{N + m}} \qquad \textbf{Equation 3.12}$$

where p_{ca} is the priory probability of the non-majority class value and *m* is a non-negative parameter.

It's important to note that: *m*-estimate takes into account prior probabilities; this way pruning is not sensitive to number of classes. In the presence of few noisy data we can set *m* to small values, leading to little pruning. On the contrary, if our data is highly noisy we must choose a high *m*, leading to much pruning, to avoid overfitting of the data.

3.3.2. Minimum number of instances in leaves

After the MEP step, it is possible that there are still some leaves with few examples that can lead to a poorer performance of the classification tree, due to the fragmentation problem. As the number of examples may be too small to make a statistically significant classification of the class representation of the leaves nodes, we remove leaves containing less than a threshold number of examples (usually a percentage of the total number of examples in the dataset). By doing this, we avoid that noisy examples

accidentally define a classification rule that would not be useful when predicting previously unseen examples of new data.

3.3.3. Maximum number of leaves

As an additional option to avoid the overfitting of data caused by too large and complex trees, we check the final number of leaves after the previous two pruning steps. If the number of leaves is higher than a certain threshold (once again this is a parameter chosen by expertise) we cut the subtree with the lowest number of instances and its parent node turns into a leaf, just like in the MEP step.

3.4. Characteristics of Classification Trees

Many data mining methods are available; so, why one should use CTs?

Classification trees are easy to interpret and, in fact, with a brief explanation people are usually able to understand them. Even though, classification tree accuracies are also comparable to other more complex and expensive classification techniques.

The presence of redundant attributes does not adversely affect the accuracy of classification trees. An attribute is redundant if it is strongly correlated to another attribute in the data. One of the two redundant attributes will not be used for splitting once the other attribute has been chosen. However, if the dataset contains many irrelevant attributes, i.e., attributes that are not useful for classification, then some of them may be accidentally chosen during the tree-growing process, making the tree larger than necessary. Feature selection techniques during pre-processing phase can help to improve the accuracy of classification trees by eliminating the irrelevant attributes.

Techniques developed for constructing decision trees are computationally inexpensive, making it possible to quickly construct models even when the dataset is very large. Large amounts of data can be analyzed using personal computers in a reasonable amount of time. CTs are quite robust to the

presence of noise, especially when applying methods for avoiding overfitting.

Thereby, using CTs to generate meta-variables can be a suitable method, regarding the data structure and the analysis goal.

3.5. Meta-variables states assignment

Having a set of SNPs annotated to one gene, we run the classification tree and we obtain a set of classification rules for these data, one rule for each leave of the pruned tree. Then we consider the antecedent of each rule as a meta-state. So, the possible values that the gene variable can assume are the set of all rules obtained from the classification tree.

Next, in order to use genes to learn Bayesian networks, we create a new column for the gene variable and we assign one of the possible gene meta-states to each example. This procedure is performed by analyzing all meta-states and testing which of the combination of SNPs values match the real SNPs values in the original dataset, i.e. we search for the meta-state that covers each example and assign it to the correspondent example line. So, at the end, we will have a new dataset, with the same number of examples but a much smaller dimension with respect to the number of attributes.

3.5.1. An example

Considering gene 1, the classification tree is learned from the set of SNPs A, C and D.

Original dataset: SNPs

Status	Gene 1	Gene 1	Gene 1
	SNP A	SNP C	SNP D
1	AA	Cc	Dd
1	AA	cc	DD
2	aa	Cc	DD
1	Aa	cc	DD
2	AA	CC	Dd

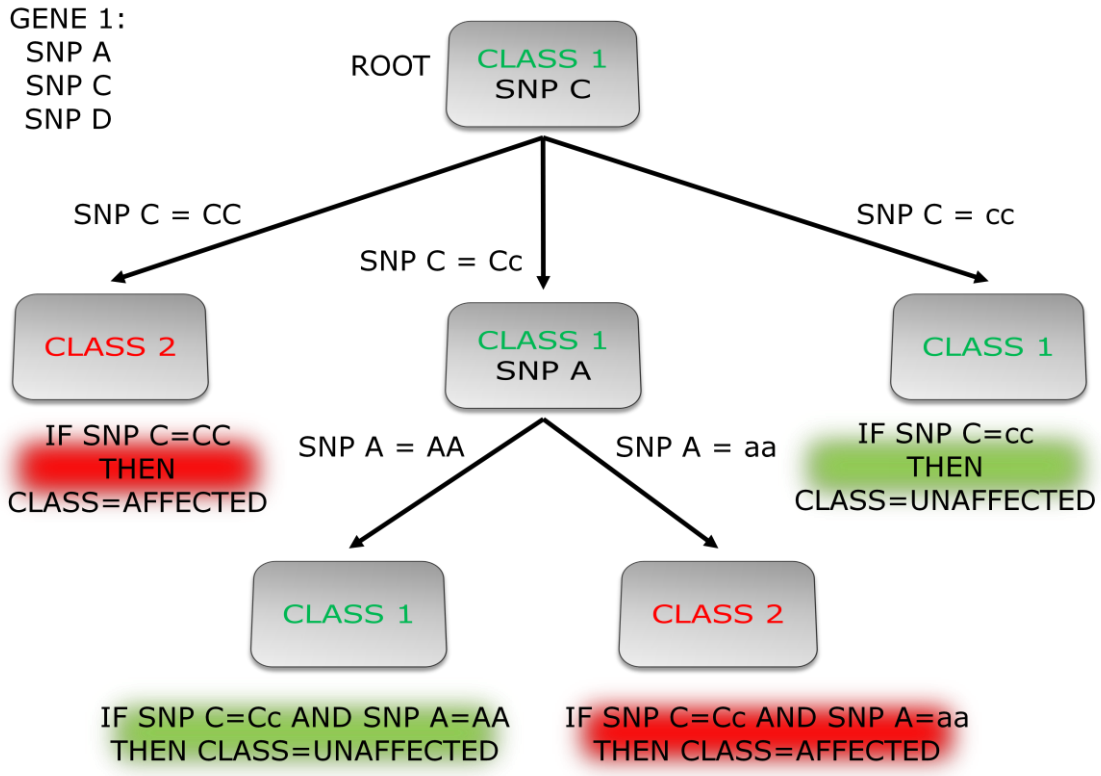

Figure 3.8 Illustration of classification tree learning and derivation of the rules.

So, we obtain the following meta-states from the set of classification rules derived:

State 1: SNP C = CC

State 2: SNP C = Cc *AND* SNP A = AA

State 3: SNP C = Cc *AND* SNP A = aa

State 4: SNP C = cc

Now we create the meta-variable corresponding to gene A in the new meta-dataset:

Meta-dataset: Genes	
Status	**Gene 1**
1	
1	
2	
1	
2	

Figure 3.9 Creation of the meta-variable corresponding to Gene 1.

We must take into account that a certain attribute might not be used in the classification tree splitting process. For example, if all the examples in one node of the CT already classify for the same phenotype, even if there are still other attributes in the set it is useless to split data again, as it will make the tree bigger (one more node or leave is created) without improving the classification tree performance.

So, a SNP that is not used to split data is also not used to create the rules for that gene, i.e. the SNP value was not tested in any of the CT nodes. Thus, when creating the rules, these SNPs are not referred and it means that they were not taken into account, so its value can be any when searching for the match between one meta-state and the real original combination of SNPs values.

Then, beginning with the first example, we look for the state that covers it and we assign it to Example 1 in the gene variable:

Example 1			
	SNP A	**SNP C**	**SNP D**
	AA	Cc	Dd

Meta-states for Gene 1				**Covered ?**
State 1	"any"	CC	"any"	No
State 2	**AA**	**Cc**	**"any"**	**YES**
State 3	aa	Cc	"any"	No
State 4	"any"	cc	"any"	No

Meta-dataset: Genes	
Status	**Gene A**
1	2
1	
2	
1	
2	

Figure 3.10 Example of meta-state assignment to an example of the dataset.

We apply this process to each example until we complete the new meta-dataset:

Meta-dataset: Genes

Status	**Gene A**
1	2
1	4
2	3
1	4
2	1

Figure 3.11 Meta-variable filled with the states derived with the classification tree.

We perform this assignment individually for each gene in order to create the new meta-dataset that will be represented by the phenotype column and one column for each gene (Figure 3.12).

Original dataset: SNPs

Status	Gene 1	Gene 2	Gene 1	Gene 1	Gene 2
	SNP A	SNP B	SNP C	SNP D	SNP E
1	a_1	b_3	c_2	d_2	e_3
1	a_1	b_1	c_3	d_1	e_3
2	a_3	b_2	c_2	d_1	e_1
1	a_2	b_2	c_3	d_1	e_2
2	a_1	b_2	c_1	d_2	e_1

Meta-dataset: Genes

Status	Gene 1	Gene 2
1	g_2	f_3
1	g_4	f_3
2	g_3	f_1
1	g_4	f_2
2	g_1	f_1

Figure 3.12 Meta-dataset built with one attribute corresponding to each gene in the original SNPs dataset.

3.6. Bayesian networks learning

Bayesian networks (BNs), also called probabilistic networks, are powerful instruments to learn genetic models from GWAS. In fact, they can be used to derive the existing correlation between SNPs and phenotype as well as to find the nonlinear relations between SNPs themselves (Sebastiani & Abad-Grau, Bayesian Networks for Genetic Analysis, 2007). In this section we recall some background on BNs definition and we present the method for learning BN structures, discussing the algorithms implemented in this thesis.

3.6.1. Bayesian networks: genetic analysis graphical models

Considering the complex nature of the genetic processes, due to stochastic origins and quantified with frequencies measurement, it is immediately clear that the analytical tools more suitable to perform such studies shall be probabilistic models (Beaumont & Rannala, 2004). Among others, BNs are a natural approach to graphically express the dependence between the variables. Bayesian networks model situations in uncertainty conditions (Charniak, 1991). Furthermore, BNs provide the possibility of

exploit efficient computational algorithms able to quantify the effects in study (Lauritzen & Sheehan, Graphical Models for Genetic Analysis, 2003).

Bayesian networks are ***directed acyclic graphs*** (**DAGs**), where the nodes are random variables and the arcs specify the independence assumptions between them, usually reflecting the cause-effect relations within the domain. The quantitative representation of these relations is given by the probability distribution of the possible values of the nodes in the network, assigning to each node a ***conditional probability table*** (**CPT**). As an example, suppose to model a situation in which the value of a variable *Wet Grass* is a non deterministic result of the values by the variables *Rain* and *Sprinkler*, which are in turn dependent on the variable *Cloudy*. The causal graph corresponding to this situation and the CPT of each node are represented in Figure 3.13.

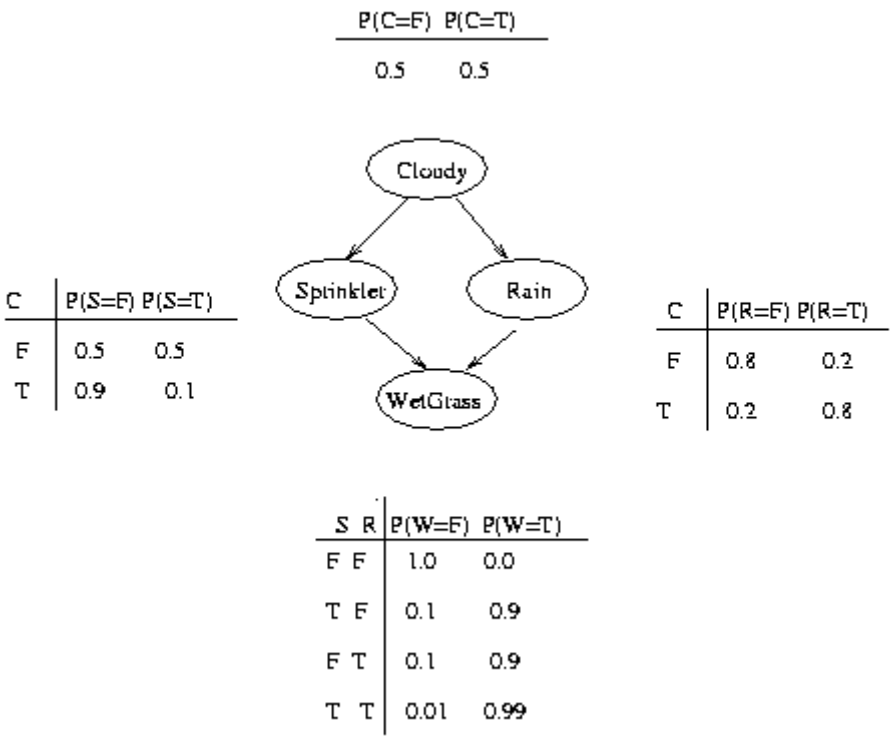

Figure 3.13 The graph represents qualitatively the dependence between the variables. The CPT of each node is the quantification of the relations represented in the graph. The nodes with no parent are given the *a priori* probabilities. For each child node, the CPT is specified through the probability that the node has a certain value when its parents have different values.

The joint probability distribution of a set of variables increases exponentially with the number of variables; in the case of having n Boolean variables the distribution is given by $2^n - 1$ joint probabilities. We can easily understand that this was the biggest obstacle to the use of the probability theory to model uncertainty situations, due to the fact that it would be needed to calculate, for example, tens of thousands of numbers for a network with tens of nodes. The first great advantage of BNs stands in their ability to quantify a joint probability distribution by computing a small number of conditional probabilities by resorting to the conditional independence assumptions underlying the graph structure.

In probability theory, two events *R* and *B* are conditionally independent given a third event *G* if, given the value of *G*, the knowledge of the values of *B* does not provide further information about the value of *A*. The occurrence or non-occurrence of *R and B* is an independent event in their conditional probability distribution given *G*.

$$P(R|B,G) = P(R|G)$$

Looking at the graph, if each possible course from *R* to *B* contains *G*, then *R* is conditionally independent from *B* given *G* (Lauritzen, Dawid, Larsen, & Leimer, 1990).

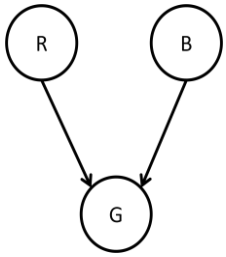

Figure 3.14 Representation of the dependence between three variables: *R*, *B* and *G*.

This assumption, exploiting the fact that the network is a DAG, allows showing that the probability of any node is expressed by the conditional probabilities table of that node given the parent nodes only (Charniak,

1991). Thus, the joint probability of all the n values of a possible instantiation of the network can be computed by the so-called *chain rule*, as follows:

$$\boldsymbol{P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P(x_i|\pi_i)} \qquad \textbf{Equation 3.13}$$

where x_i is the instantiation of the i-th variable and π_i represents the instantiation of the parent nodes of x_i. So, the joint probability of the network is no longer exponential but the product of n variables (Cooper & Herskovits, 1992). Accordingly, the Bayesian network can be defined as a structure containing a DAG with casual variables and a matrix with the conditional probabilities of the each node of the DAG, given its parent nodes.

As computational tool, BNs are used to update the joint probability distributions given the available evidence on certain nodes. Therefore, since BNs allow to update the model (and in consequence the comprehension of the phenomenon in study) when new observations are available, once a BN is learned, it is an efficient device of probabilistic inference, i.e. can calculate the posterior probabilities of unobserved variables on the basis of evidence on the values of other variables in the network (Lauritzen & Spiegelhalter, Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems, 1988).

So, how to build a BN? First of all, the construction of the network contains two steps: structure learning, i.e. the identification of the network topology, and learning of the parameters, i.e. the quantification of the CPTs of the nodes. As the goal of this work is to identify the dependence relations between genotypes and phenotype, the crucial part of learning the BN can be approached as a model selection problem (it is the network topology that allows showing the presence of association between variables), in which different network models are compared on the basis of their posterior probability with respect to the available data. We search for the network that better describes the relations of probabilistic dependence in data. This is a difficult problem because of the high number of possible DAGs, even

with few nodes. In the following, we will discuss how to learn the best network structure.

3.6.2. K2 algorithm

If *D* is the cases group in the database of study and *Z* the set of variables in *D*, we can calculate the probability of the ratio between B_{S_i} and B_{S_j}, two possible structures containing variables in *Z*, and establish a comparative ordering of these structures regarding their posterior probability. Knowing that the joint probability of any variables *A* and *B* is defined as

$$P(A,B) = P(A|B)P(B) \qquad \textbf{Equation 3.14}$$

we can write that probability of the ratio as follows:

$$\frac{\boldsymbol{P(B_{S_i}|D)}}{\boldsymbol{P(B_{S_j}|D)}} = \frac{\frac{\boldsymbol{P(B_{S_i},D)}}{\boldsymbol{P(D)}}}{\frac{\boldsymbol{P(B_{S_j},D)}}{\boldsymbol{P(D)}}} = \frac{\boldsymbol{P(B_{S_i},D)}}{\boldsymbol{P(B_{S_j},D)}}. \qquad \textbf{Equation 3.15}$$

Therefore, the conditional probability can be calculated through the joint probability, which in turn can be calculated under the *Cooper assumptions*:

1. No missing values.
2. Use of discrete variables.
3. Given a network, cases occur independently
4. Uniform density function $f\ (B_P|B_S)$.

Given these assumptions, the following result is proven (Cooper & Herskovits, 1992):

Theorem

Let *Z* be a set of n discrete variables, where a variable x_i can assume r_i possible values: $(v_{i1}, \dots, v_{ir_i})$. Let *D* be a database with *m* cases, each one containing the variables in *Z*. Let B_S be the network structure that contains the variables in *Z*. Each variable x_i in B_S has a set of parents, which we represent with a list of variables π_i. Let w_{ij} denote the j-th instantiation of π_i relative to *D*. Suppose there are q_i such unique instantiations of π_i. Define $N_{ijk}$to be the number of cases in *D* in which variable x_i has the value v_{ik} and π_i is instantiated as w_{ij}. Let $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. Given assumptions 1 through 4, it follows that

$$\boldsymbol{P(B_S, D) = P(B_S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i-1)!}{(N_{ij}+r_i-1)!} \prod_{k=1}^{r_i} N_{ijk}!} \qquad \textbf{Equation 3.16}$$

The double product corresponding to $P(D|B_S)$ is known as *marginal likelihood*.

Concerning the joint probability (which is called *Bayesian measure*) calculation, we can use some appropriate restrictions (like bounding *u,* the maximum number of parents of any node, and also bounding *r*, the number of possible values for one variable) and then the overall complexity becomes $O(m, n)$ (Cooper & Herskovits, 1992).

Furthermore, if we assume that the database is generated by some network containing the variables in *Z*, then we can compute $P(D)$ by summing $P(B_S, D)$ over all possible B_S containing just the variables in *Z*. In the general case, if all the possible structures that can originate data contain the same variables *Z* and defining *Q* as the set of all those network structures that contain just the variables in *Z,* the marginal probability on *D* is obtained summing the joint probability values over all possible structures:

$$P(B_{S_i}|D) = \frac{P(B_{S_i}, D)}{\sum_{B_S \epsilon Q} P(B_S|D)} \qquad \textbf{Equation 3.17}$$

The computation of Equation 3.17 is heavily conditioned by the fact that it is possible to find *Y* a subset of *Q* that makes possible to perform the marginalization (sum in denominator) over B_S. There may be more than one structure. In our case $P(D)$ is known because the dataset is fixed. So, we can search the most probable network considering that Equation 3.15 results $P(B_S, D) \propto P(B_S|D)$.

To find the most probable network it would be necessary to exhaustively apply Equation 3.16 for every possible B_S. We easily understand that, as a function of the number of nodes, the number of possible structures grows exponentially, making this search unfeasible. For this reason, it was derived the following efficient recursive function for determining the number of possible network structures that contain *n* nodes (Cooper & Herskovits, 1992):

$$\boldsymbol{f(n) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} f(n-i)} \qquad \textbf{Equation 3.18}$$

For example, to have an idea of the growth of the number of possible structures, $f(2) = 3$, $f(3) = 25$, $f(5) = 29281$, $f(10) \approx 4{,}2 \times 10^{18}$.

Thus, some heuristic algorithms were developed to restrict the best DAG research space, maintaining the definition of Equation 3.16. The criterion that better reduces the computational complexity of the task is the ordering of the nodes. If we suppose that is possible to define that x_i precede x_j, then it is not possible to build an arc going from x_j to x_i. We can therefore express the maximization of Equation 3.16 as

$$\boldsymbol{\max_{B_S}[P(B_S, D)] = \prod_{i=1}^{n} \max_{\pi_i} \left[P(\pi_i \rightarrow x_i) \prod_{j=1}^{q_i} \frac{(r_i-1)!}{(N_{ij}+r_i-1)!} \prod_{k=1}^{r_i} N_{ijk}! \right]}$$

Equation 3.19

where $\pi_i \rightarrow x_i$ indicates that π_i is the set of possible parents of x_i in B_S.

Despite the adoption of this criterion, still remains $2^{\binom{n}{2}} = 2^{n(n-1)/2}$ possible configurations of the network. We must therefore take into account the complexity due to the calculation of $P(\pi_j \rightarrow x_j)$.

The first algorithm suggested by (Cooper & Herskovits, 1992) to render computationally feasible the assessment of Equation 3.19 is based on the assumptions that:

i. There is an ordering of the nodes.
ii. There is a limit for the number of possible parents of each node.
iii. $P(\pi_i \rightarrow x_i)$ and $P(\pi_j \rightarrow x_j)$ are marginally independent for $i \neq j$, in order to efficiently compute the prior distributions.

This algorithm, known as **K2**, consists in searching, for each node, for the set of parent nodes that maximizes the function:

$$\boldsymbol{g(i,\pi_i) = \prod_{j=1}^{q_i} \frac{(r_i-1)!}{(N_{ij}+r_i-1)!} \prod_{k=1}^{r_i} N_{ijk}!} \qquad \textbf{Equation 3.20}$$

At the beginning, K2 assumes that a node has no parents, and then adds incrementally that parent which addition most increase the probability of the resulting structure. In every step K2 computes the calculation of Equation 3.20 making an instantiation of π_i with the considering nodes. The procedure stops when the addition of no single parent can increase the posterior probability (Cooper & Herskovits, 1992).

A pseudo code representation of K2 algorithm is shown in Figure 3.15.

```
Algorithm K2
INPUT: A set of n nodes, an ordering on the nodes, an upper bound u on the number of parents a node may have, and a database D containing m cases.
OUTPUT: For each node, a printout of the parents of the node.
BE GIN K2
    FOR i := 1 TO n DO
        BEG IN
            Π_i := 0;
            P_old := g(i, Π_i):
            OKToProceed := TRUE
            WHI LE OKToProceed AND |Π_i| < u DO
                BEG IN
                    Let Z be the node in Pred(X_i) – Π_i that
                     maximizes g(i, Π_i ∪ {Z});
                    P_new := g(i, Π_i ∪ {Z});
                    IF P_new > P_old THEN
                        BEG IN
                            P_old := P_new;
                            Π_i := Π_i ∪ {Z}
                        END
                    ELSE OKToProceed := FALSE;
                END;
            WRITE('Node:,' X_i, 'Parents of this node:,' Π_i)
        END;
END K2.
```

Figure 3.15 Pseudo-code of the K2 algorithm **(Cooper & Herskovits, 1992)**.

The algorithm guarantees a topology not containing any cycles, as it prevents any arc from connecting the lower-ordered nodes to the higher-ordered nodes. Nevertheless, the network topology may change if we apply a different ordering scheme to the variables (Tan, Steinbach, & Kumar, 2006).

This strategy searches for the most probable network given the available observations and permits to learn both the network structure and the conditional probabilities.

3.6.3. Probabilistic Inference: Junction Tree

Having created the BN, we can now use it for inference. In general, probabilistic inference on a network is the process of computing the posterior probability distribution $P(V = v \mid E = e)$, or simply $P(v|e)$, where v is a value of a variable V and e is an assignment of values to a set of variables E in the network.

There are several different algorithms to perform inference efficiently, where each one makes different tradeoffs between speed, complexity, generality, and accuracy. We have used the propagation algorithm ***junction tree***, which is the mother of all exact inference algorithms, developed by Lauritzen and Spiegelhalter (Lauritzen & Spiegelhalter, Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems, 1988) and refined by Jensen (Huang & Darwiche, 1996).

3.7. Application results

As referred before, this approach was first applied and led to the published paper *Phenotype forecasting with SNPs data through gene-based Bayesian networks* (Malovini, Nuzzo, Ferrazzi, Puca, & Bellazzi, 2009).

The paper used data came from a genome-wide scan on a case population of 288 individuals affected by arterial hypertension (AH) and a control population of 271 nonagenarians without history of AH. A standard statistical analysis was performed, obtaining a list of 93 highly significant SNPs. After SNPs annotation and selection of only those genes represented by at least two SNPs, the final analyzed dataset consisted of 559 examples (288 cases and 271 controls) and 24 SNPs, mapping to 9 genes.

Then, using the whole dataset, a Bayesian network was learned using SNPs as variables. Since SNPs mapping to the same gene resulted highly correlated, this network seemed to be able to correctly infer a direct dependence between them.

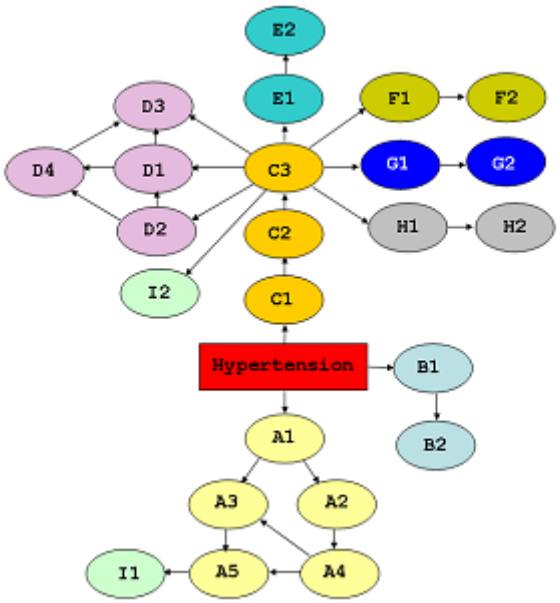

Figure 3.16 SNP-based BN learned using the whole dataset **(Malovini, Nuzzo, Ferrazzi, Puca, & Bellazzi, 2009)**.

These results showed that it was possible to reduce the variables space without losing information by creating a meta-variable for each gene. Furthermore, the status (hypertension) appeared directly connected to 3 genes and, among SNPs mapping to the same gene, it was always connected to the marker with the highest gain ratio.

Next, a new network was built using the meta-variables, whose states were derived as outlined in section 3.5, and the results showed that the phenotype was connected to the same genes as in the network learned with all SNPs.

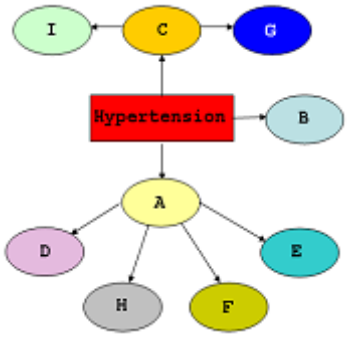

Figure 3.17 Meta-variable BN learned using the whole dataset **(Malovini, Nuzzo, Ferrazzi, Puca, & Bellazzi, 2009)**.

Thereby, these results confirmed that meta-variables are really able to summarize the relationships between genes and phenotype without losing information.

As the goal of the analysis was to assess the ability of the learned BN models to predict the phenotype status given a certain configuration of SNPs or genes, the predictive accuracy on both BN was calculated on the whole dataset, obtaining 62.79% for the single-SNP network and 64.22% for the gene network. However, as training accuracy is affected by overfitting, it is much more interesting to evaluate the generalization accuracy. In order to do that, a random sampling hold-out was repeated 5 times, using 75% of the dataset (419 examples) as training set and the remaining 25% as test set (140 examples) and regarding stratification, so that both sets had the same distribution of phenotypic classes as the entire dataset. So, the learning of the meta-variables and the BN was performed on the training set, while the accuracy of phenotype prediction was calculated on the test set. Note that the mapping of the SNPs on the test set into meta-variables was done using the rules derived on the training set.

The mean accuracies resulting from the 5 tests were 58.99% and 64.28% respectively for the SNP network and for the gene network. These results suggested that the proposed method is able to achieve a better classification performance than the single-SNPs one.

3.8. Limitations and problems

Considering the description of the method done until now we can easily realize that the complexity of the analysis compromises its reproducibility, making it hard to perform replication or further generalization of the process.

First of all, the use of different software tools for each step of the method makes it impossible to automatically do the analysis:

- To achieve the original datasets with SNPs data we used *PLINK* (Web 5);
- To annotate genes we used *Genephony* (Nuzzo & Riva, 2008) (Web 6).
- Then, to learn the classification trees we used *Orange* (Web 7).
- Finally, to learn the BNs from data and infer on new data we used *Bayesware Discoverer* (Web 8).

Besides, some steps of the analysis are done manually:

- If we want to use a smaller dataset, containing only some top-scored significant SNPs and not all SNPs coming out from *PLINK* (Web 5) we have to manually select them with a text editor, after having calculated its scores (Gain Ratio score, for instance).
- As we have seen before, to apply the classification tree to each gene data we must have only the SNPs mapping to that gene. The original dataset contains data from all SNPs from all genes. So, once again we have to manually separate the data corresponding to SNPs mapping to each gene in a new file, creating as much files as the number of genes. Only then we can learn the classification tree on each gene and create the meta-variable. After assigning the states to each meta-variable we have to put them all back together in one single file to learn the BN on genes data.

All of this becomes more difficult because we must have into consideration the file formats used by the different software tools.

Furthermore, the definition of states for the new meta-attributes is based on graphical inspection of the pruned trees. Obviously, in addition to being more laborious, it is also more prone to errors during its execution.

Finally, the validation of the analysis with methods like hold-out or k-fold cross validation requires the whole process to be done a very high number of times, which is manually unfeasible.

3.9. Objectives

So, aside from redo the whole analysis process, the main goal of this thesis was to overcome all the limitations and problems described in section 3.8 by developing a tool able to perform an overall automation of the strategy. This way, using the developed tool we can easily achieve the following main objectives:

- Automate the meta-attributes states definition and its assignment.
- Integrate all the steps in a single framework.
- Enable a more robust evaluation of the performance of the proposed approach.

So, we developed *SNP2Net*, a MATLAB tool that aims at automatically performing the analysis process described in section 3, in which we can also be able to easily set the many parameters required over the implementation of the strategy by using a graphical user interface (GUI).

Moreover, *SNP2Net* intend to enable a properly assessment of the method performance.

Finally, this tool is designed to be a general instrument, i.e. to make possible further analysis on different datasets.

In the next chapter we describe in detail the structure of the tool and the implemented algorithms and strategies.

4. Developed tool: *SNP2Net*

SNP2Net was developed in MATLAB environment and is provided with a graphical user interface (GUI), making it user-friendly and intuitive. As reported in the previous section, *SNP2Net* aims at performing an overall automation of the strategy presented in section 3 and providing also an assessment of the analysis performance. Moreover, by using *SNP2Net,* the method can be tested on different datasets.

In this chapter we will present the structure of the tool modules, the implemented algorithms, as previously outlined in section 3, and the specific parameters to be set by the user.

This tools allows for an authomatization of the overall strategy described in (Malovini, Nuzzo, Ferrazzi, Puca, & Bellazzi, 2009): the only need is to coordinate *SNP2Net* with the software *PLINK* (Web 5) and the on line annotation resource *Genephony* (Web 6).

The input of *SNP2Net* is an annotated SNPs dataset, i.e. the final SNPs dataset showed in Figure 3.4. Then, the approach consists of the following modules:

- Data pre-processing
- Features selection
- Classification Tree
- Bayesian Network
- Validation

Figure 4.1 shows a schematic representation of the structure of *SNP2Net*:

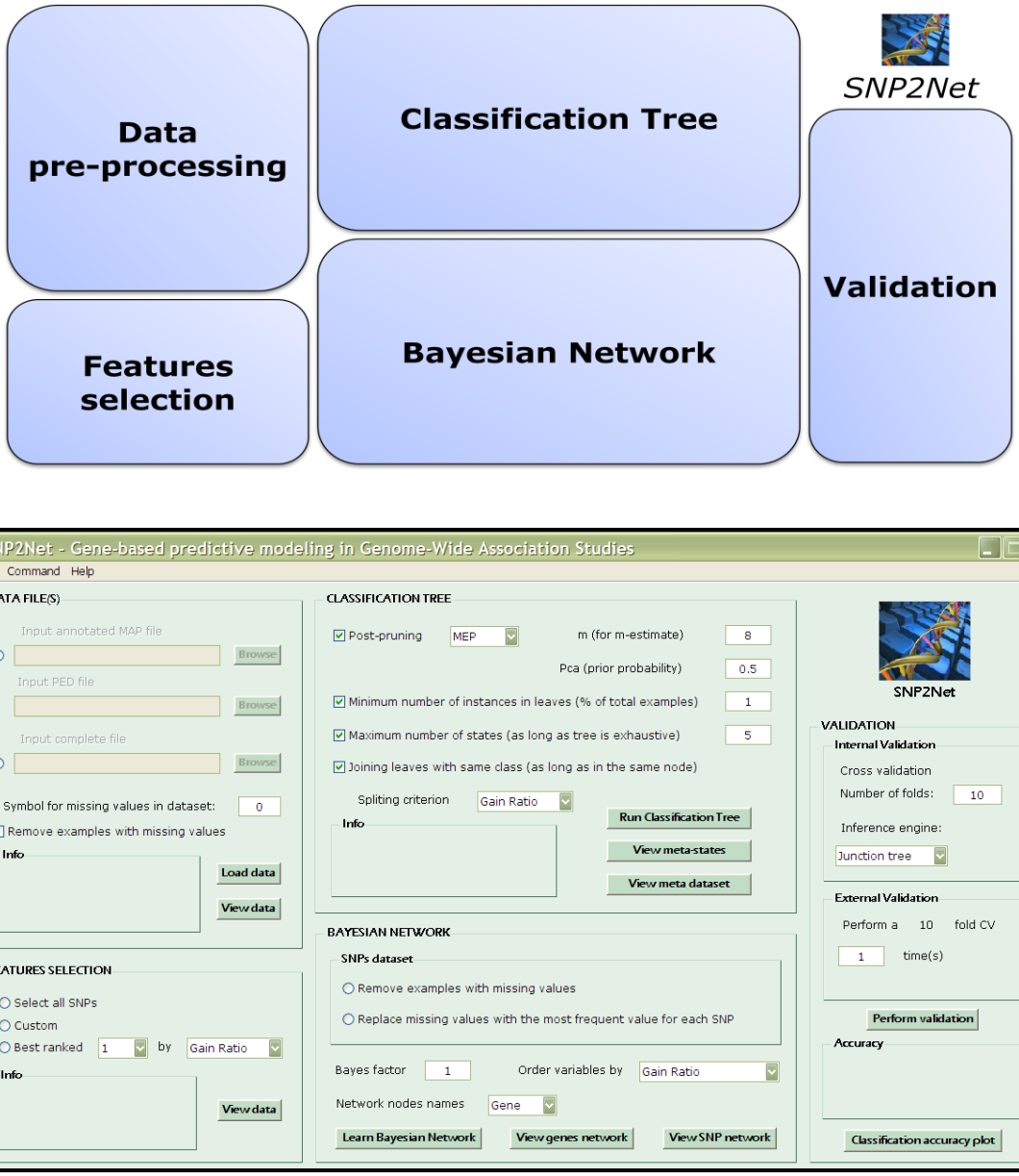

Figure 4.1 On top, schematic representation of the *SNP2Net* modules. Below, a screenshot of *SNP2Net.*

In the next sections we describe in detail each module and the implemented devices.

4.1. Data pre-processing

Figure 4.2 shows the graphical layout of the *Pre-processing* module of *SNP2Net* that allows the user to upload the dataset and choose how to handle missing values.

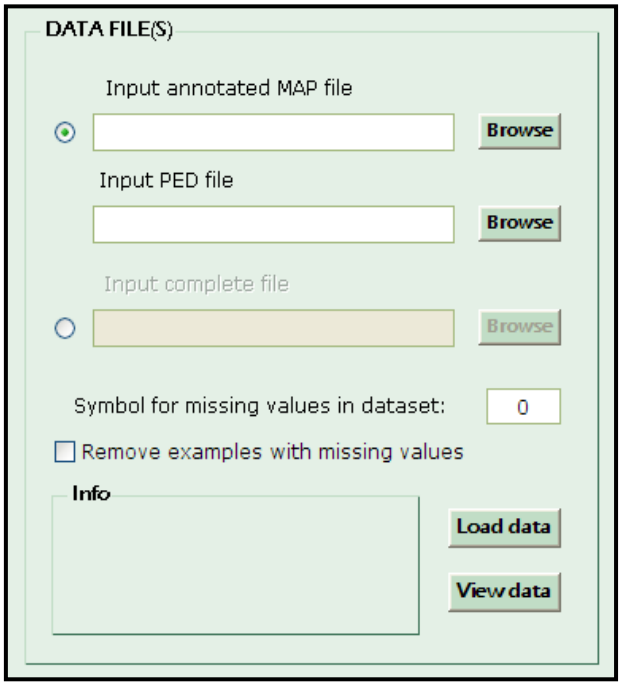

Figure 4.2 Screenshot of the *Pre-processing* module.

The user can upload two different data formats: *complete files* or *annotated MAP/PED files*.

If the ***Input complete file*** option is selected, the user can input data in a format represented in Figure 3.4, i.e. a file containing SNPs genotypes and two header lines, the 1st having the genes annotation and the 2nd having the SNPs identification. Let us remember that, in this kind of files, the genes annotation was performed manually, writing in the 1st row the official gene symbol for each SNP.

Selecting ***Input annotated MAP file*** and ***Input PED file***, the user can upload in *SNP2Net* the annotated MAP and PED files as generated by *PLINK*,

as described in section 3.1 (except for the MAP file that must be manually annotated).

The user is first requested to specify the symbol used to represent missing values in the dataset (***Symbol for missing values in dataset)***

As outlined in section 3.6.2, K2 learning algorithm cannot analyze dataset with missing data (a frequent issue in GWAS); so the user can choose to remove rows in the dataset that contain missing values for one or more SNPs (***Remove examples with missing values*** option). However, by removing incomplete examples, the dimension of the resulting dataset becomes function of the genotyping rate of the analyzed dataset (see section 1.2.4) and may result in a loss of statistical power.

Once the dataset has been uploaded (***Load data***), a routine eliminates unnecessary information (1st to 5th columns) and loads data into a suitable structure to the remaining processes.

After loading data, *SNP2Net* calculates i) the number of examples, ii) the number of examples with missing values (and the corresponding percentage in respect to the total number of examples) and iii) the number of attributes (SNPs) in the dataset.

Figure 4.3 shows the graphical interface of this module after loading data.

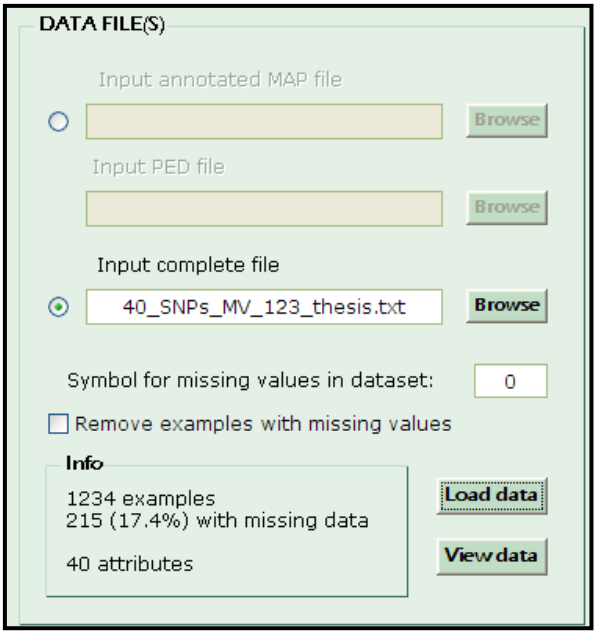

Figure 4.3 Screenshot of the *Data pre-processing* module after loading data.

After loading the data it is possible to perform a graphical inspection of the data (see Figure 4.4) and to export the new dataset into a text format file.

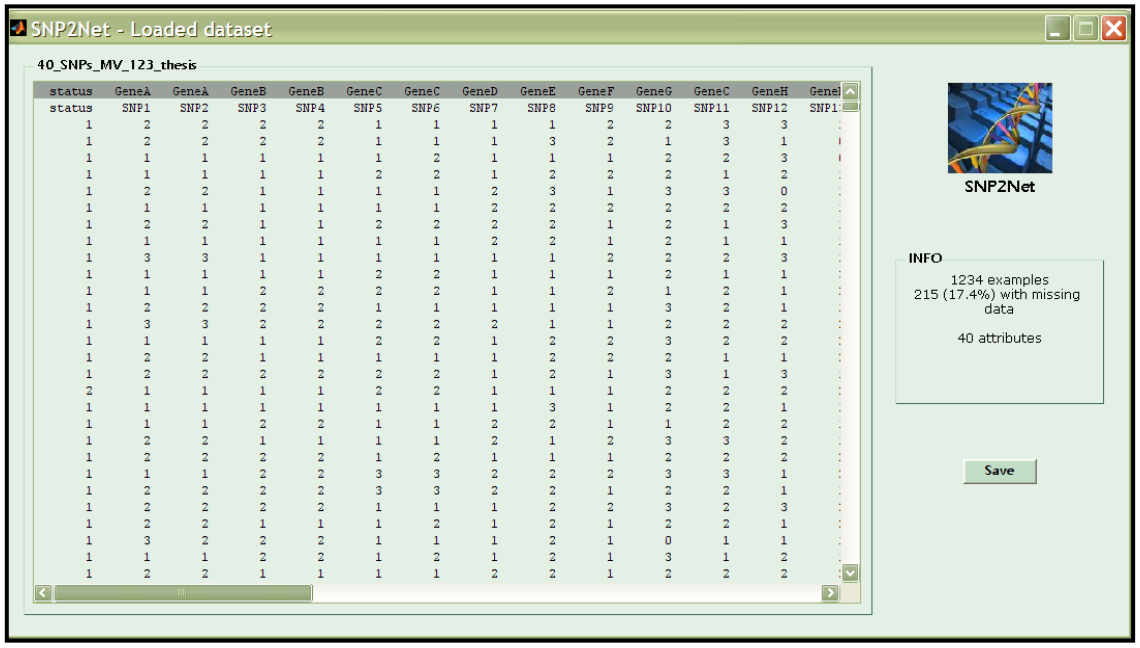

Figure 4.4 Dataset visualization window representing the loaded data.

4.2. Features selection

As referred before, datasets can have a large number of features. The learning algorithms can be very expensive when performing on such large datasets. So, a reduction of dimensionality of data can be very helpful once the amount of time and memory required by the data mining algorithms is reduced, in particular if the number of attributes in the data is lower. Also the data can be more easily visualized when reduced. Furthermore, reduction of dimensionality can itself eliminate irrelevant features and reduce noise (Tan, Steinbach, & Kumar, 2006). Selecting new attributes that are a subset of the old is known as *feature subset selection* or simply *features selection*.

SNP2Net eliminates irrelevant attributes like Sex or Family ID, as we know that those attributes do not contain information of interest for our study purposes. But since redundant attributes can be still present, the *Features Selection* module implements a strategy to perform features selection known as *Filter approach*, that allows to discard irrelevant features.

There are three options to perform features selection: *Select all SNPs*, *Custom* and *Best ranked*.

Choosing ***Select all SNPs***, all the available SNPs are analyzed (Figure 4.5).

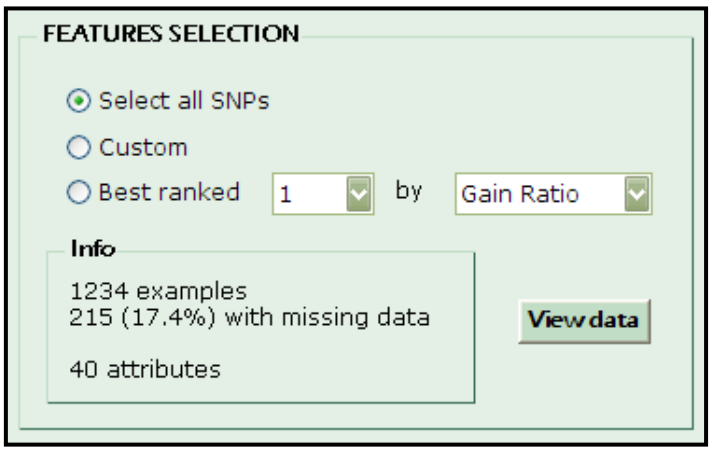

Figure 4.5 Screenshot of the *Features Selection* module selecting all available SNPs.

The ***Custom*** features selection option enables the user to select only the SNPs of interest: a graphical interface (Figure 4.6) shows a list of the available SNPs in the original dataset and the corresponding official gene symbols, enabling the user to select only the SNPs of interest.

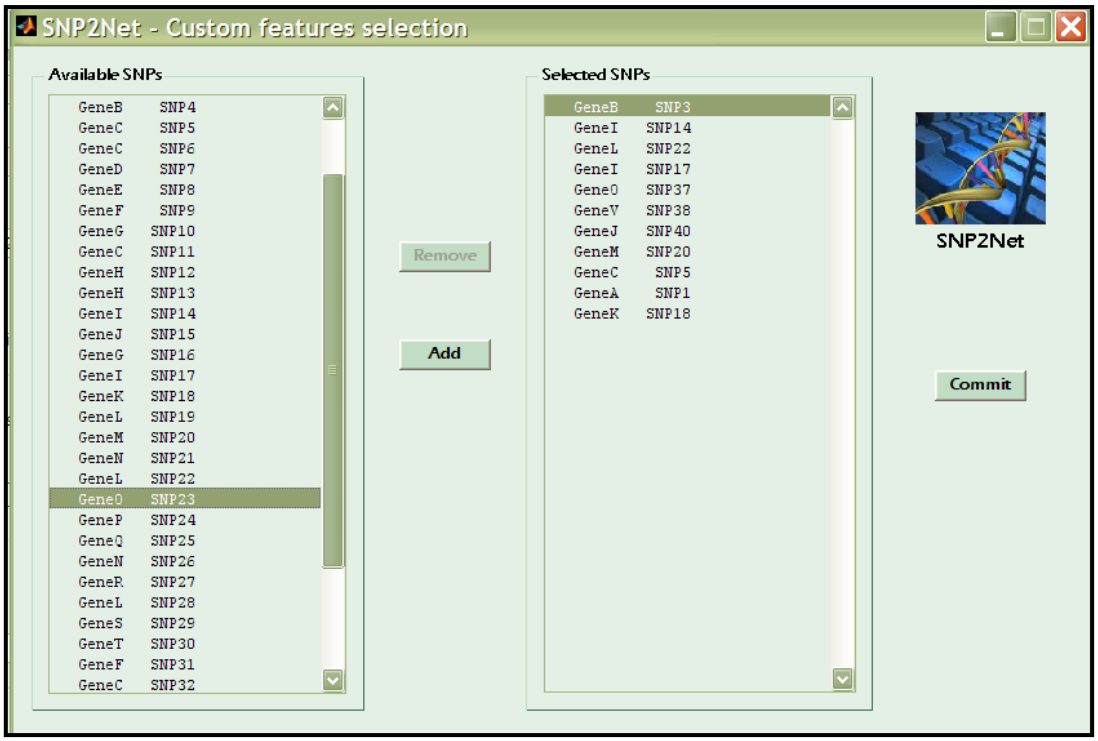

Figure 4.6 Screenshot of the *Custom* features selection window.

Finally, by choosing the ***Best ranked*** option, it is possible to select the most informative SNPs, sorted by descending order of Gain Ratio score.

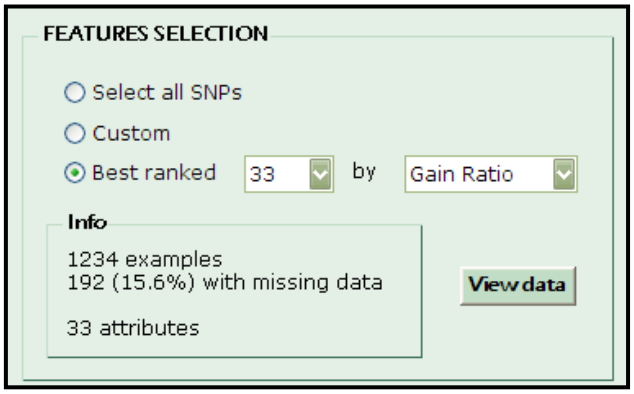

Figure 4.7 Screenshot of the *Features Selection* module selecting *Best ranked* SNPs.

SNP2Net implements the Gain Ratio as a scoring measure since it calculates a robust normalized value that takes into account for the number of feature values in the dataset. Hereafter, in the future the tool will also implement different score measure to be used for the features ranking.

After the features selection process, the tool generates a dataset for each gene (represented by the SNPs mapping to it and the phenotype) to be analyzed by the *Classification Tree* algorithm.

4.3. Classification Trees

The *Classification Trees* generation (the *Classification Tree* module is shown in Figure 4.8) represents a crucial step in the proposed method, since it enables the definition of the values that each meta-variable representing a gene may assume.

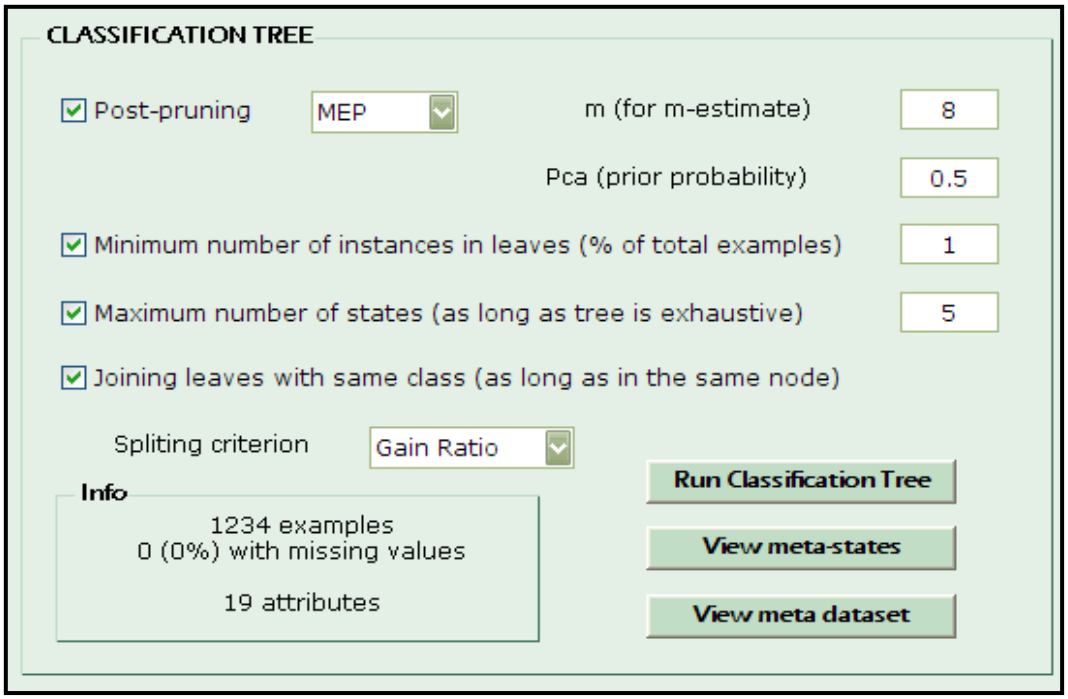

Figure 4.8 Screenshot of the *Classification Tree* module.

This module implements the algorithm C4.5 (Quinlan, 1993) (described in section 3.2.1) to learn the CTs, since it represents an exhaustive strategy. Accordingly, the criterion used to split data on tree nodes is represented by the Gain Ratio score.

The user can set the following parameters:

- *m*, for the error estimation in each node of the CT, using *m*-estimate; note that the pruning strategy implemented is the *Minimal Error Pruning*, although other strategies can be implemented in future.
- *Minimum number of instances in leaves*, as a percentage of the total number of examples in the dataset.
- *Maximum number of states* for each gene, i.e. the maximum number of values a gene meta-variable can assume; however be aware that the tree should be exhaustive and, therefore, cover all examples in the dataset. Thus, if a leaf contains less examples than the threshold number set in *Minimum number of instances in leaves* (which means this leaf would be discarded and its examples would not be included in the model) the maximum number of states (corresponding to the final leaves) allowed is increased by one, and the leaf is not eliminated.
- *Joining leaves with the same class*; if there are two leaves descending from the same node that classify for the same class value, then a new leaf is created, containing the examples of the two previous leaves. In this case, a logical conjunction *OR* will be used to indicate the possible values of the SNP corresponding to that leaf.

Note that the *Pca* parameter, corresponding to the prior probability of the majority class, is fixed to 0.5, assuming that is equally likely, a priori, that a given example is a case or a control example.

Once the pruned CT is obtained, the classification rules are derived and the meta-states are assigned to genes, as described in section 3.5.

In order to overcome the problem of missing data (what often happens in biological data of the type we want to study) we implemented a strategy named *multiplication of examples*, using the following procedure:

1. Given an example E, create missing(*E*) with all SNPs containing missing values for that example.
2. For E, build examples with all possible values of the missing SNPs.

3. If $P(A_{i,j})$ is the probability of attribute A_i having the j[th] value, the weight of each example is $w_E = \prod_{A_i \in missing(E)} P(A_{i,j})$.

So, the computation of Information Gain, Gain Ratio and all other counts are performed using weighted examples, a combinatorial strategy that involves an increase in the number of examples that is proportional to the number of attributes with missing data. The multiplication of examples allows to manage missing data by assigning the most likely combination of the set of SNPs mapping to each gene to each example. Hence, this strategy represents an alternative either to: i) imputation of data, often generating artifacts and ii) the removal of data, resulting in true but smaller datasets.

So, *SNP2Net* learns a CT and assigns the states to each gene independently and generates automatically the new dataset, represented by genes instead of SNPs as variables. The configurations of the states each gene can assume are available to the user. Finally, the user can also visualize the new gene-based dataset and export it into a text format file.

4.4. Bayesian Network

Once the genes-based dataset has been generated, it is possible to use it to learn a BN as a phenotype predictive model.

The Bayesian network generation module implements the K2 algorithm (see section 3.6.2) using functions from the Matlab Bayes Net Toolbox (BNT) by Kevin Murphy (Murphy, 2007).

K2 algorithm requests the setting of a searching order among the variables. A modification of the function that learns the best topology of the BN (*learn_struct_K2)*, has been applied in order to allow the user to set the value of the ***Bayes Factor*** *(BF)*. This parameter is given by the ratio between the score of the new and the actual network structure during the greedy search referred above. As the K2 algorithm does not admit missing values in data, the user is also requested to choose between two options, regarding the missing data management:

- *Remove examples with missing values;* this option does not manipulate data but originates smaller datasets.
- *Replace missing values with the most frequent value for each SNP*; this option maintains the dataset dimensions by imputing missing data.

Once the BNs are learned from both SNPs and genes datasets, the user can visualize and export them into an image file format.

Figure 4.9 shows the graphical layout of the *Bayesian Network* module and an example of a genes-based Bayesian network.

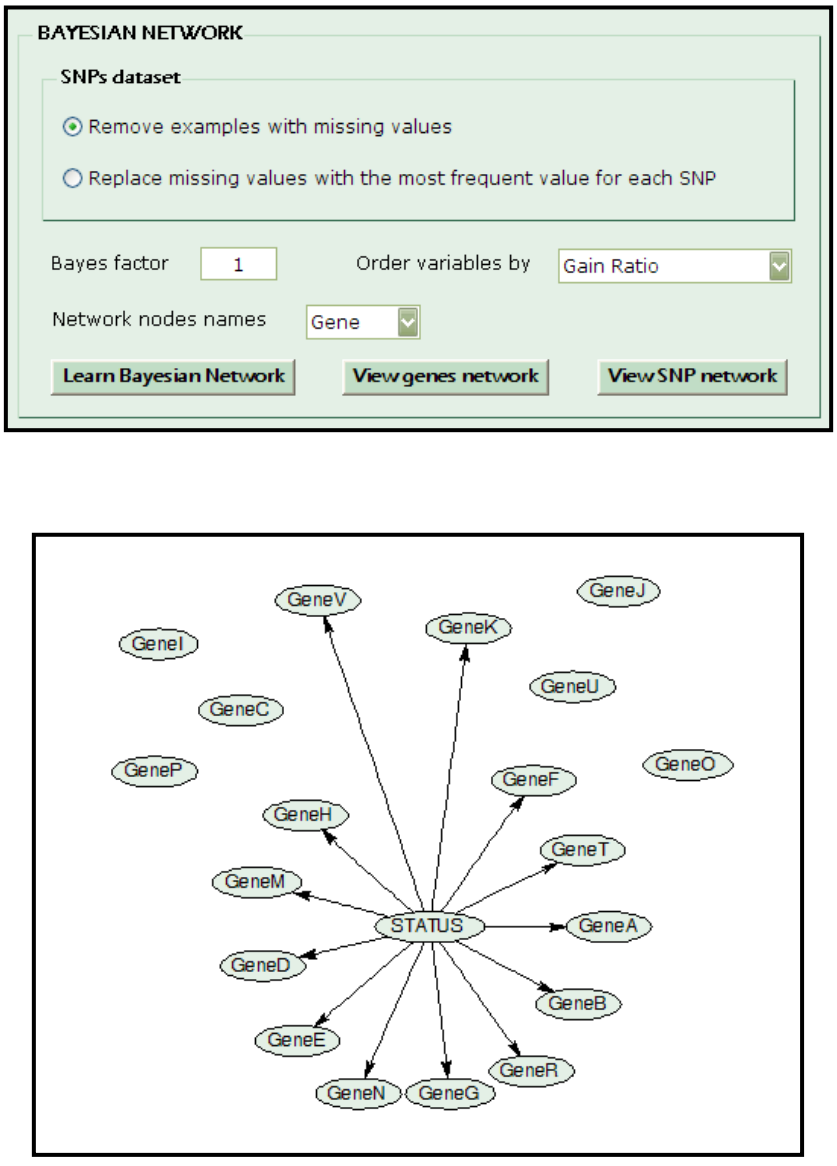

Figure 4.9 On top, screenshot of the *Bayesian Network* module; below, screenshot of the representation of a genes network topology.

4.5. Validation

The *Validation* module, shown in Figure 4.10, was implemented in order to allow the user to perform an evaluation of the performances obtained by the tool. S*NP2Net* allows not only for an *Internal Validation*, by applying the whole scheme to subsets of the original dataset, but also an *External Validation*, by repeating the analysis a very high number of times, in order to estimate its robustness.

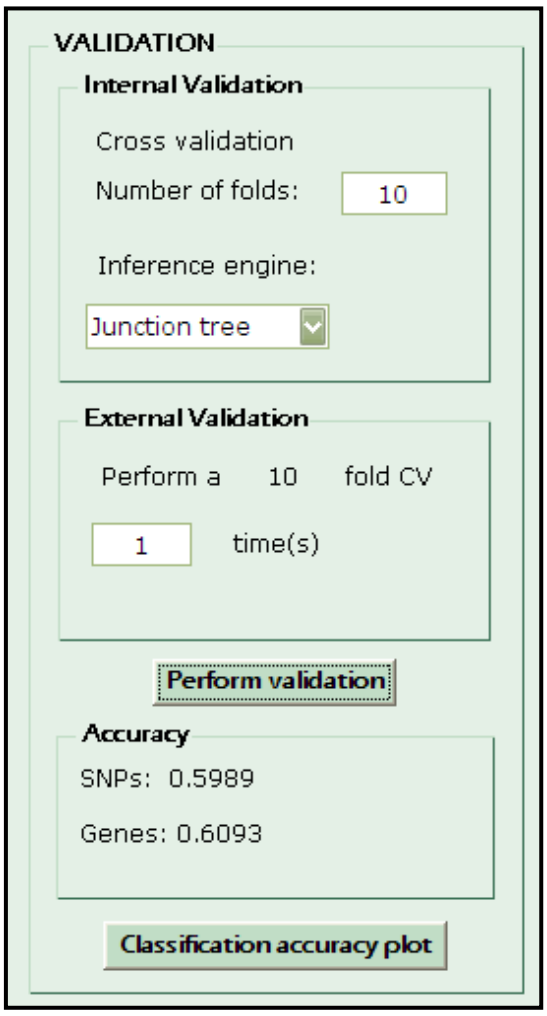

Figure 4.10 Screenshot of the *Validation* module.

4.5.1. Internal Validation

As the final goal of the analysis is to build a disease predictive model given a set of features, the predictive performances of the learned networks are tested on subsets of data that are independent from those used for learning (training set).

To perform this assessment, we implemented a **K-fold Cross-Validation** (CV) strategy. As described in Figure 4.11, the original dataset is randomly divided in K folds, properly regarding the stratification issue, i.e. each fold has the same class values distribution as the original dataset. Then, the strategy consists in considering for K times (K is equal to the number of folds set by the user) one fold as the test set and all the remaining K-1 folds as training set.

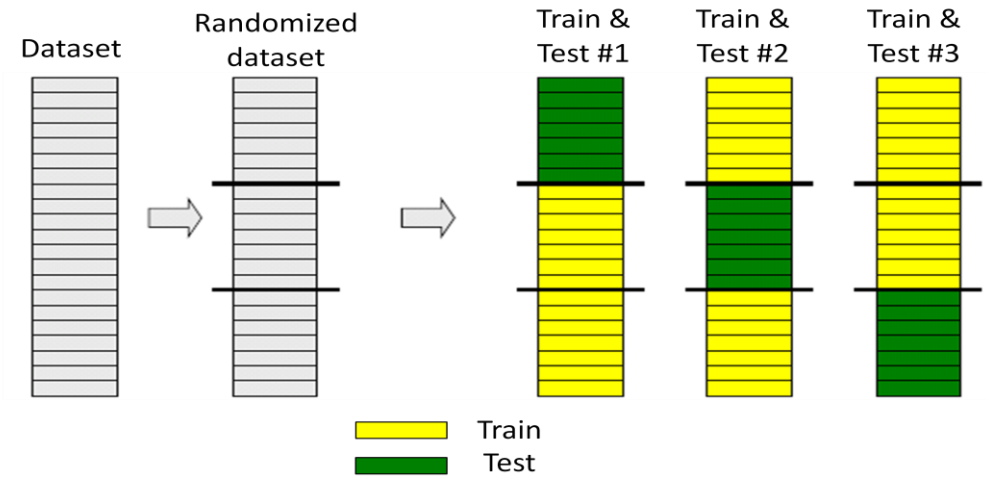

Figure 4.11 Representation of a 3-fold Cross Validation strategy.

Then, *SNP2Net* performs the following steps for each *Train & Test* pair:

- The CT is learned on training set.
- The corresponding derived rules are used to assign the genes states on both training and test sets.
- The BN is learned on both SNPs and genes training sets.
- The phenotype is inferred on test set.

To do inference of phenotype, we use the *junction tree* algorithm, also implemented in BNT (Murphy, 2007). Inferring means computing the probability distribution of one node (phenotype) given the values of the other nodes (SNPs or genes). The network is tested on the test set and the classification accuracy is computed as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

where: *TP* is the number of True Positives, i.e. the number of examples correctly classified as having the reference class (in our case class 1, unaffected); *TN* is the number of True Negatives, i.e. the number of examples correctly classified as having the non reference class (class 2, affected); *FP* is the number of False Positives, i.e. the number of examples classified as 1 when the real class is in fact 2; *FN* is the number of False Negatives, i.e. the number of examples classified as 2 when the real class is in fact 1.

So, the assessment of the ability of the learned BN models is performed by predicting the phenotype status given a certain configuration of SNPs or genes.

4.5.2. External Validation

The *External Validation* consists in performing the K-fold CV a high number of times (set by the user), which gives an approximation of the performance strength with respect to the fold sampling. Note that, each time a new K-fold CV is performed, the original dataset is randomly sorted.

When performing a K-fold CV, the user obtains the median value of the predictive accuracy on both SNPs and genes data, over the K *Train & Test* pairs.

If the user chooses to perform the K-fold CV more than one time, *SNP2Net* shows the median accuracy values over the total *N* times and also allows the user to visualize a box plot of the median accuracy values of each K-fold CV.

Thus, performing the K-fold Cross-Validation a high number of times gives a measure of the stability of the performance with respect to the fold sampling.

An example of a box plot obtained from a 1000-times analysis repetition is shown in Figure 4.12:

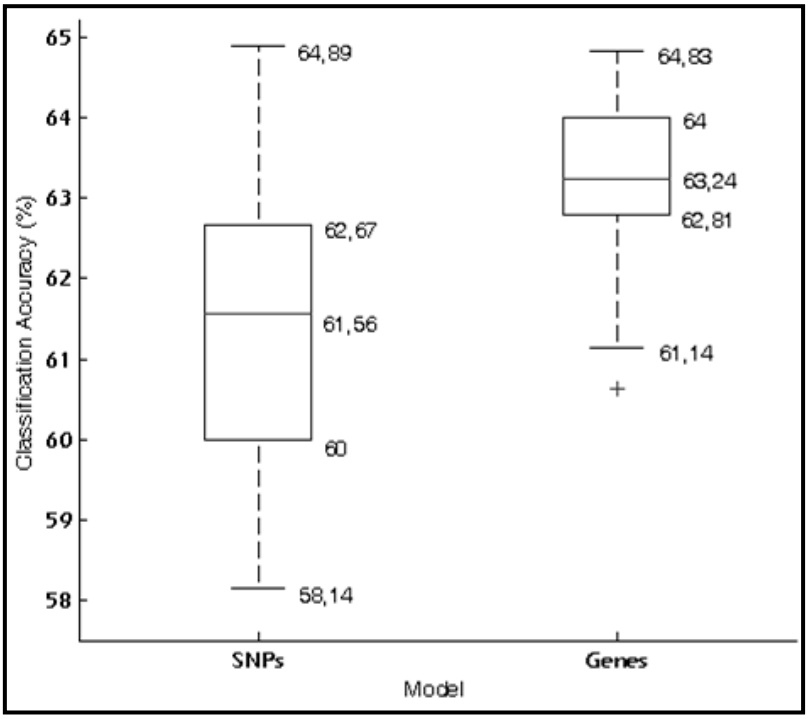

Figure 4.12 Box plot of the classification accuracy distribution obtained from a 1000-times replication of 10-fold CV, on both SNPs and genes (meta-attributes) networks.

4.6. Global workflow description

In order to allow for a clear understanding of the whole analysis, we now present a list of the several steps, from the beginning until the end of the process:

- Using *PLINK* (Web 5)
  - Perform GWAS to select top-significant SNPs selection.
- Using *Genephony* (Web 6)
  - Annotate genes.
- Using *SNP2Net*
  - Create meta-variables corresponding to genes.
  - Learn BNs on both SNPs and genes training sets.
  - Perform inference on both SNPs and genes test sets.
  - Validate the analysis through a K-fold CV strategy.

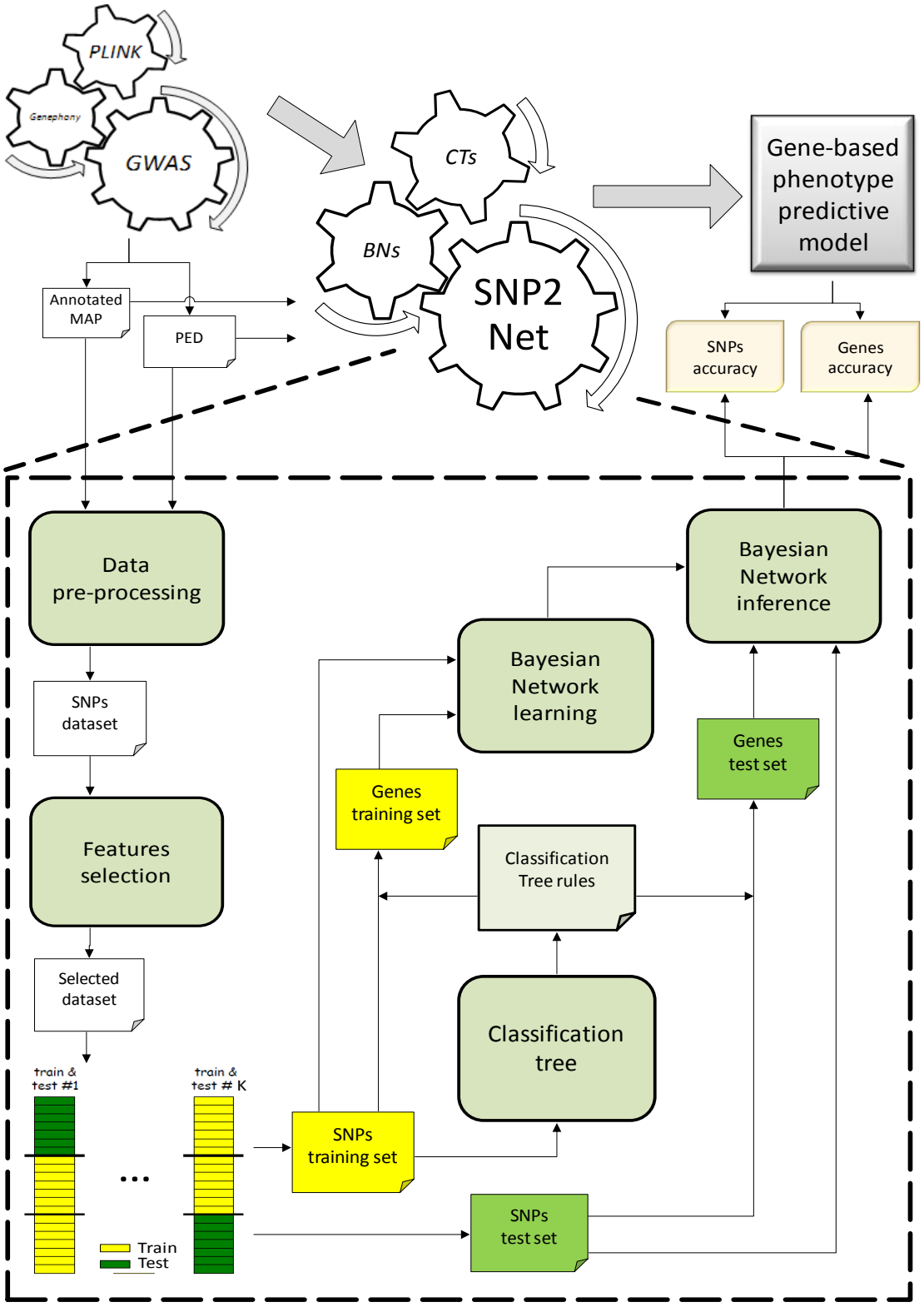

Figure 4.13 Schematic representation of the whole analysis process.

5. Results and discussion

The main goal of the work done on this thesis was the development of an application able to perform an overall automation of the analysis process (see section 3) and allowing the user to easily redo the process not only a high number of times but also apply it to different datasets, through an integrated framework. Furthermore, once this tool was available, an additional objective of this work was to test it on different datasets, in order to evaluate the performance of the strategy and confirm the conclusions obtained in (Malovini, Nuzzo, Ferrazzi, Puca, & Bellazzi, 2009).

In fact, the *SNP2Net* application it will be soon make available at http://bioinfo.unipv.it to general use. Then we tested *SNP2Net* on different datasets (different dimensions and containing different SNPs and genes), among which a dataset containing data coming from a genome-wide scan involving 570 35-55 years old patients affected by arterial hypertension (AH) and a control population of 664 individuals without an AH historical. This dataset contains data corresponding to the 40 top-associated SNPs, which map to 22 genes. Since these data still have to be biologically validated, we will denote genes by letters and SNPs by numbers.

In Figure 5.1 we present a representation of the topology of the Bayesian network learned using single-SNPs as variables and employing the whole dataset. In this network the SNPs within each gene appear connected, probably because SNPs mapping to the same gene present correlated configurations and thus the BN learning algorithm correctly infers a direct dependence between them. This result supports the hypothesis that considering the SNPs mapping to the same gene as a unique meta-variable is an appropriate way to make an abstraction of the network structure without losing information.

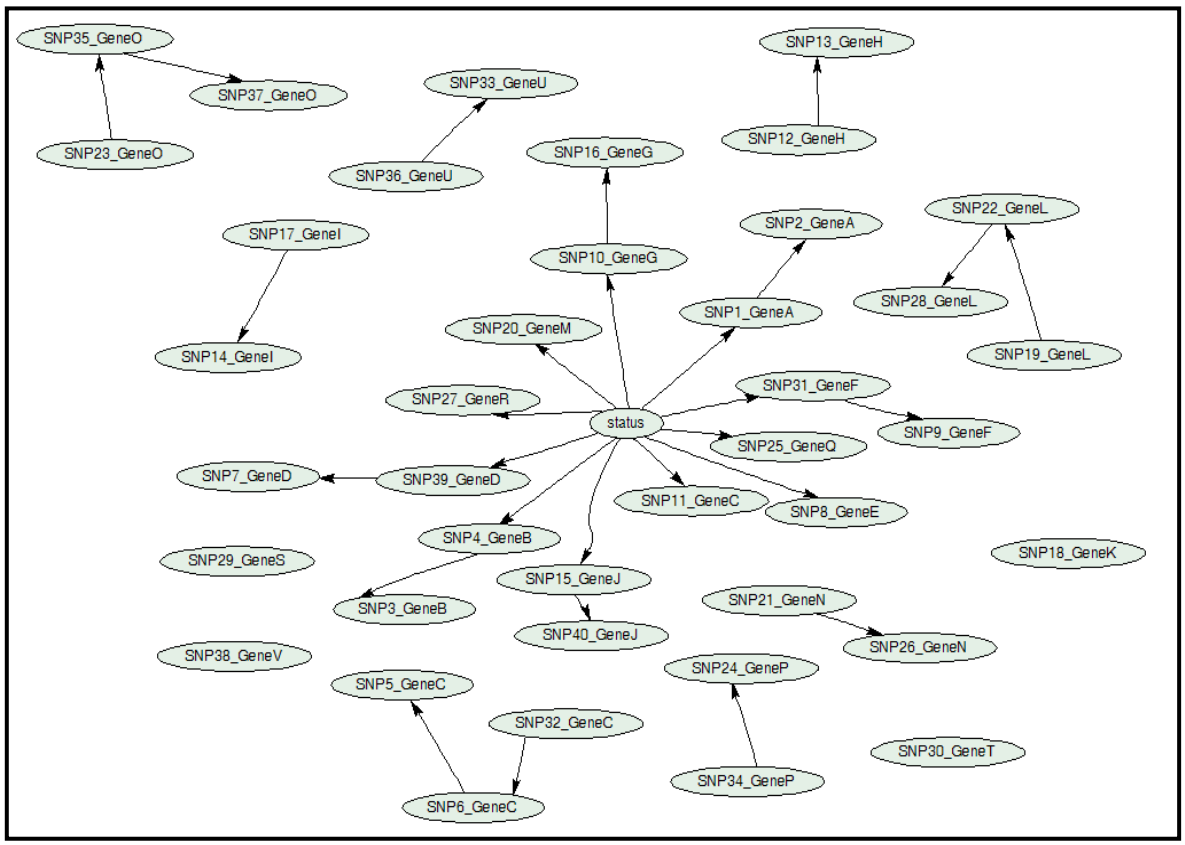

Figure 5.1 Bayesian network learned on the whole dataset using single-SNPs as variables.

Figure 5.2 shows a representation of the topology of the Bayesian network learned using single meta-variables corresponding to genes and employing the whole dataset. We can see that the phenotype is directly connected to the same genes as in the network learned with all SNPs.

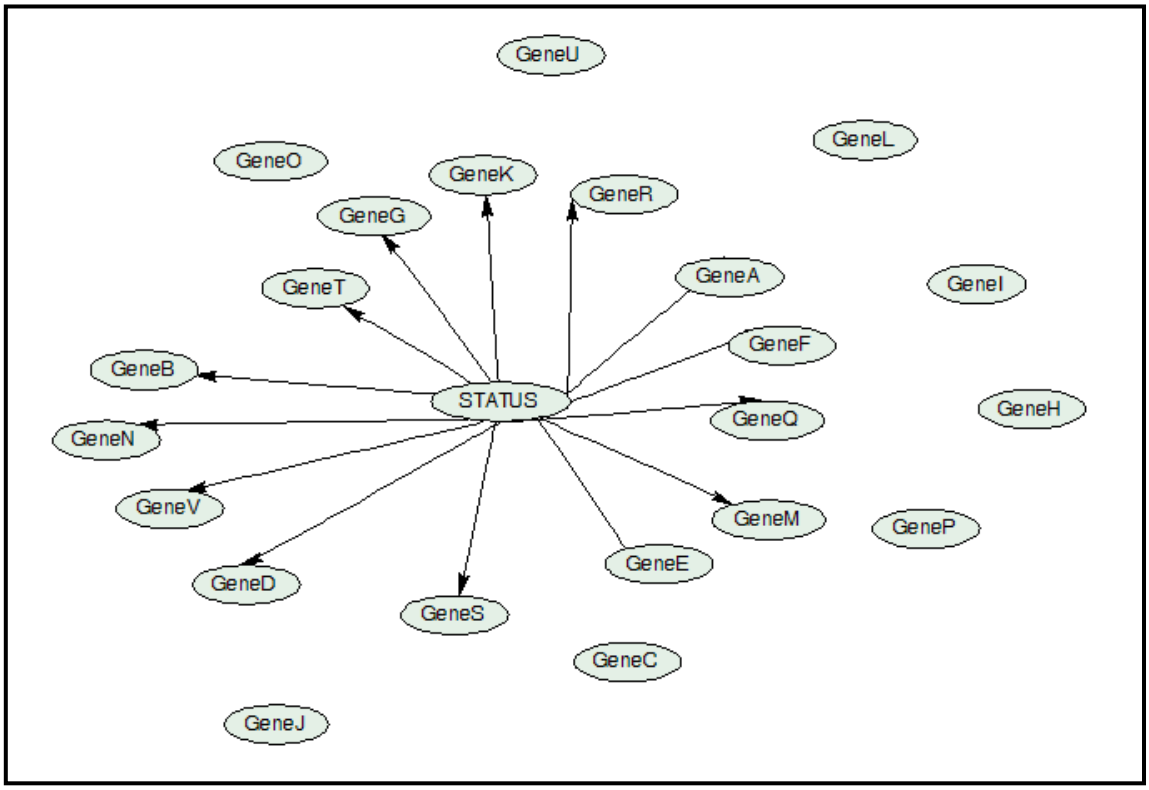

Figure 5.2 Bayesian network learned on the whole dataset using meta-variables associated to each gene.

In order to evaluate the predictive ability of the networks, after performing a 10-fold CV we obtained a classification accuracy of 58,18% for the SNP-based network and 62,80% for the gene-based net. Finally, performing a 150-times replication of the 10-fold CV, we obtained two different accuracy values distribution summarized in the box plot shown in Figure 5.3Figure 5.3.

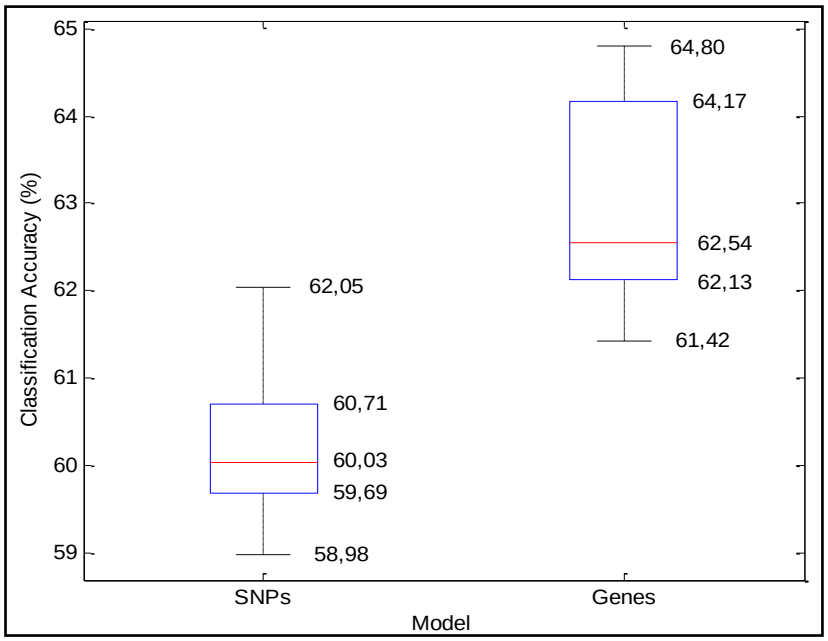

Figure 5.3 Box plot of the classification accuracy distribution obtained from a 150-times replication of 10-fold CV, on both SNPs and genes networks. SNP-based model has a distribution around the median accuracy of 60,03%, while the gene-based model has a distribution around a higher median value, 62,54%.

As *SNP2Net* easily allows applying the analysis strategy, we now present a table with the median values of the classification accuracy for several tests on different datasets.

Classification performances on test sets					
# 10-fold CV	Model			SNP-based	Gene-based
	# examples	# SNPs	# Genes	Classification accuracies (%)	
1	1234	17	10	60,03	61,23
1	1234	60	25	56,76	62,17
1	559	24	9	61,26	63,05
10	559	24	9	61,13	62,58
150	1234	40	22	60,03	62,54
1000	559	24	9	61,56	63,24

Figure 5.4 The table summarizes the results obtained by performing a 10-fold CV with random sampling in several sets of data.

Conclusions

Biological and biomedical research are increasingly becoming *data-intensive* disciplines, as a result of the widespread adoption of high-throughput experimental techniques able to generate very large amounts of data, at a low cost and in a short time. These new technologies allow studying biological systems on a large scale, which greatly expands our understanding of the interplay between genetic and environmental factors, all the way up to the relationship between the genotype and medically relevant phenotypes. This change affects not only the way in which biomedical data is generated but also, more fundamentally, the way in which it needs to be handled, interpreted and mined for new knowledge. To follow the rapid evolution of high-throughput techniques and to ensure that the results they produce are being exploited to their full potential, researchers increasingly need powerful, effective and easy-to-use software tools to automatically manage large volumes of data and to extract new knowledge from them.

So, the work presented and discussed in this thesis regards the development of an analysis tool for association studies data. The data used during this work correspond to a non Mendelian disease, the arterial hypertension. Then, the research context is the investigation of the genetic origins of this type of diseases, *complex traits*, which cannot be analyzed using the standard techniques for the analysis of Mendelian diseases. The association studies of *case-control* type use data collected not only from the patients but also from their relatives and a control population. Thus, in addition to the collection and data management, it is important to optimize and automate the data analysis processes.

Therefore, the work carried on consisted in the development of *SNP2Net*, a software tool that uses *data mining* techniques to look for evidence of association between the genotype and the phenotype of the individuals. Using the theory of Bayesian networks we implemented an algorithm with great inference potential over the variables of the domain of interest (predicting the value of the phenotype variable, given the values of the other variables). Specifically, we developed a tool that aims at deriving a

gene-based predictive model based on SNPs data, through the use of classification trees to create the variables corresponding to genes.

Such model is more parsimonious than the one based on single-SNPs, while preserving the capability of highlighting predictive SNPs configurations and prediction performances. Its limited number of nodes provides an abstract view of the relationship between genes and the phenotype of interest, and therefore represents an alternative way to analyze the available data. Moreover, since our proposed method has fewer variables than the SNP-based one, it is also less prone to over fitting. Thanks to the availability of *SNP2Net*, it was possible to test the prediction performance of this approach in several sets, noting that it was consistently superior to the SNP-based model. Another interesting remark is the fact that our proposed method seemed to have less variance than the SNP-based one.

The proposed model is a suitable alternative to haplotypes, which are on the contrary frequently used also as prediction factors.

It is needless to say that the implemented tool easily allows further generalization. For example, it is possible for the user to define lists which contains SNPs that may be related, even if they do not belong to specific gene. In this case, it is possible to implement a customized two step prediction strategies that may allow using different kinds of knowledge and user-provided information.

Regarding further developments of the tool, there are some issues that can still be addressed.

The management of missing data is a major issue in the field of data mining and machine learning. A suitable way of handling missing values can be implemented, in order to allowing the user to work on more reliable datasets. The choice of removing examples with missing data is a simple way of using a true dataset. However, it could be useful to use for example the expectation-maximization (EM) algorithm, which finds maximum likelihood estimates of parameters in probabilistic models, where the model depends on unobserved latent variables.

Also the available score measures used to perform features selection or ordering the variables input on the BNs can be extended, allowing the user to introduce different model design details and thus expand the range of the experiments, in order to find the more appropriate way to analyze data.

Finally, we think *SNP2Net* makes available to users a new framework to support data management in genome-wide studies.

We showed that the system can effectively be used to look for answers to complex questions about genotype-phenotype correlations.

References

Balding, J. D. (2006). A tutorial on statistical methods for population association studies. *Nature Reviews Genetics* , 7(10):781-791.

Beaumont, M. A., & Rannala, B. (2004). The Bayesian revolution in genetics. *Nature Genetics* , 5:251-261.

Botstein, D., & Risch, N. (2003). Discovering genotypes underlying human phenotypes: past successes for mendelian disease, future approaches for complex disease. *Nature Genetics* , 33:228-237.

Boyle, J., Cavnor, C., Killcoyne, S., & Shmulevich, I. (2008). Systems biology driven software design for the research enterprise. *BMC Bioinformatics* , 9:295.

Butte, A. J. (2008). Translational Bioinformatics: Coming of Age. *Journal of American Medical Informatics Association* , 15(6):709-714.

Charniak, E. (1991). Bayesian networks withou Tears. *Artificial Intelligence Magazine* , 50-63.

Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning* , 9:309-347.

Cordell, J. C., & Clayton, D. G. (2005). Genetic association studies. *Lancet* , 366(9491):1121-1131.

Hirschhorn, J. N., & Daly, M. J. (2005). Genome-wide association studies for common diseases and complex traits. *Nature Reviews* , 6(2):95-108.

Huang, C., & Darwiche, A. (1996). Inference in belief networks: a procedural guide. *International Journal of Approximate Reasoning* , 15 (3): 225-263.

Iafrate, A., Feuk, L., Rivera, M. N., Listewnik, M. L., Donahoe, P. K., Qi, Y., et al. (2004). Detection of large-scale variation in the human genome. *Nature Genetics* , 36:949-951.

IHGSC, (. H. (2001). Initial sequencing and analysis of the human genome. *Nature* , 409:860-921.

Lander, E. S., & Stork, N. J. (1994). Genetic Dissection of Complex Traits. *Science* , 265,5181,2037-2048.

Lauritzen, L. S., Dawid, A. P., Larsen, B. N., & Leimer, H. G. (1990). Independence properties of directed Markov fields. *Networks* , 20(5):491-505.

Lauritzen, S. L., & Sheehan, N. A. (2003). Graphical Models for Genetic Analysis. *Statistical Science* , 18(4):489-514.

Lauritzen, S. L., & Spiegelhalter, S. J. (1988). Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *Journal of the Royal Statistical Society Series B* , pp. 50(2):157-224.

Malovini, A., Nuzzo, A., Ferrazzi, F., Puca, A. A., & Bellazzi, R. (2009). Phenotype forecasting with SNPs data through gene-based Bayesian networks. *BMC Bioinformatics , Suppl 2:S7*, 5-10.

Martin-Sanchez, F., Iakovidis, I., Norager, S., Maojo, V., de Groen, P., Van der Lei, J., et al. (2004). Synergy between medical informatics and bioinformatics: facilitating genomic medicine for future health car. *Journal of Biomedical Informatics* , 1;6 Suppl 4:S18.

McCarthy, M. I., Abecassis, G. R., Cardon, L. R., Goldstein, D. B., Little, J., Ioannidis, J. P., et al. (2008). Genome-wide association studies for complex traits: consensus, uncertainty and challenges. *Nature reviews* , 9(5):356-369.

Mitchell, T. M. (1997). *Machine Learning.* McGraw-Hill.

Murphy, K. (2007). *Bayes Net Toolbox for Matlab*. Retrieved June 28, 2009, from http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html

Niblett, T., & Bratko, I. (1986). *Progress in Machine Learning.* England: Sigma Press.

Nuzzo, A., & Riva, A. (2008). A Knowledge Management Tool for Translational Research. *AMIA Summit on TRanslational Bioinformatics.* San Francisco, California.

Payne, P. R., Johnson, S. B., Starren, J. B., Tilson, H. H., & Dowdy, D. (2005). Breaking the translational barriers: the value of integrating biomedical informatics and translational research. *Journal of Investigation in Medicine* , 53(4):192-200.

Philippi, S., & Kohler, J. (2006). Addressing the problem with life-science databases for traditional uses and systems biology. *Nature Reviews Genetics* , 7(6):482-8.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufman.

Rapley, R., & Harbron, S. (2004). *Molecular Analysis and Genome Discovery.* Chichester: John Wiley & Sons Ltd.

Scott, M. P., Matsudaira, P., Lodish, H., Darnell, J., Zipursky, L., Kaiser, C. A., et al. (2004). Molecular Cell Biology. San Francisco: W. H. Freeman.

Sebastiani, P., & Abad-Grau, M. M. (2007). Bayesian Networks for Genetic Analysis. *Systems Bioinformatics: An Engineering Case-Based Approach* (pp. 205-227). Artech House.

Sebastiani, P., Ramoni, M. F., Nolan, V., Baldwin, C. T., & Steinberg, M. H. (2005). Genetic dissection and prognostic modeling of overt stroke in sickle cell anemia. *Nature Genetics* , 37(4):435-440.

Sebat, J., Lakshmi, B., Troge, J., Alexander, J., Young, J., Lundin, P., et al. (2004). Large-sclae copy number polymorphism in the human genome. *Science* , 305:525-528.

Strachan, T., & Read, A. P. (1999). *Human Molecular Genetics 2.* BIOS Scientific Publishers Ltd.

Tan, P. N., Steinbach, M., & Kumar, V. (2006). *Introduction to Data Mining.* Addison Wesley.

Watson, J. D., & Crick, F. (1953). Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid. *Nature* , 171,737-738.

Web 1. (n.d.). Retrieved June 28, 2009, from Wikipedia - Single-nucleotide polymorphism: http://en.wikipedia.org/wiki/Single-nucleotide_polymorphism

Web 2. (n.d.). Retrieved June 20, 2009, from Genome News Network: http://www.genomenewsnetwork.org/

Web 3. (n.d.). Retrieved June 21, 2009, from The Human Genome: http://genome.wellcome.ac.uk/doc_wtd020778.html

Web 4. (n.d.). Retrieved June 21, 2009, from Mendelian Genetics: http://www.ndsu.nodak.edu/instruct/mcclean/plsc431/mendel/mendel9.htm

Web 5. (n.d.). Retrieved June 16, 2009, from PLINK: Whole genome association analysis toolset: http://pngu.mgh.harvard.edu/~purcell/plink/

Web 6. (n.d.). Retrieved June 21, 2009, from Genephony: http://bioinformatics.ufl.edu:8080/gp/main/

Web 7. (n.d.). Retrieved June 16, 2009, from Orange: Data mining fruitful & fun: http://www.ailab.si/orange/

Web 8. (n.d.). Retrieved June 16, 2009, from Bayesware Limited: http://bayesware.com/products/discoverer/discoverer.html

Witten, I. H., & Frank, E. (2000). *Data mining: Practical machine learning tools and techniques with Java implementations.* San Francisco, CA, USA: Morgan Kaufman.