Probabilistic Graphical Models for Prognosis and Diagnosis of Breast Cancer

PROBABILISTIC GRAPHICAL MODELS FOR PROGNOSIS AND DIAGNOSIS OF BREAST CANCER

BY

MAHMOUD KHADEMI, M.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTING AND SOFTWARE

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

Master of Science (2013) McMaster University
(Computing and Software) Hamilton, Ontario, Canada

TITLE:	Probabilistic Graphical Models for Prognosis and Diagnosis of Breast Cancer
AUTHOR:	Mahmoud Khademi M.Sc., (Computer Engineering) Sharif University of Technology, Tehran, Iran
SUPERVISOR:	Dr. Ned Nedialkov
NUMBER OF PAGES:	xi, 61

To my mother and father.

Abstract

One in nine women is expected to be diagnosed with breast cancer during her life. In 2013, an estimated $23,800$ Canadian women will be diagnosed with breast cancer and $5,000$ will die of it. Making decisions about the treatment for a patient is difficult since it depends on various clinical features, genomic factors, and pathological and cellular classification of a tumor.

In this research, we propose a probabilistic graphical model for prognosis and diagnosis of breast cancer that can help medical doctors make better decisions about the best treatment for a patient. Probabilistic graphical models are suitable for making decisions under uncertainty from big data with missing attributes and noisy evidence.

Using the proposed model, we may enter the results of different tests (e.g. estrogen and progesterone receptor test and HER2/neu test), microarray data, and clinical traits (e.g. woman's age, general health, menopausal status, stage of cancer, and size of the tumor) to the model and answer to following questions. How likely is it that the cancer will extend in the body (distant metastasis)? What is the chance of survival? How likely is that the cancer comes back (local or regional recurrence)? How promising is a treatment? For example, how likely metastasis is and how likely recurrence is for a new patient, if certain treatment e.g. surgical removal, radiation therapy, hormone therapy, or chemotherapy is applied. We can also classify various types of breast cancers using this model.

Previous work mostly relied on clinical data. In our opinion, since cancer is a genetic disease, the integration of the genomic (microarray) and clinical data can improve the accuracy of the model for prognosis and diagnosis. However, increasing the number of variables may lead to poor results due to the curse of dimensionality dilemma and small sample size problem. The microarray data is high dimensional. It consists of around $25,000$ variables per patient. Moreover, structure learning and parameter learning for probabilistic graphical models require a significant amount of computations. The number of possible structures is also super-exponential with respect to the number of variables. For instance, there are more than 10^{18} possible structures with just 10 variables.

We address these problems by applying manifold learning and dimensionality reduction techniques to improve the accuracy of the model. Extensive experiments using real-world data sets such as METRIC and NKI show the accuracy of the proposed method for classification and predicting certain events, like recurrence and metastasis.

Acknowledgements

This project was funded by McMaster Centre for Software Certification. Many thanks go to my advisor, Dr. Ned Nedialkov, for his advising on this research; his dedication to my work is very much appreciated. I am thankful to the examination committee members Dr. Antoine Deza and Dr. Franya Franek for their helpful suggestions. Thanks to the Synapse Teams for providing the METABRIC dataset.

Notation and Abbreviations

BN (Bayesian Network)

CPD (Conditional Probability Distributaion)

PVD (Posterior Vitreous Detachment)

MLE (Maximum Likelihood Estimation)

EM (Expectation Maximization)

IID (Independent and Identically Distributed)

PCA (Principal Component Analysis)

Isomap (Isometric feature mapping)

LPP (Locality Preserving Projections)

LLE (Locally Linear Embedding)

SVMs (Support Vector Machines)

k-NN (k-Nearest Neighbor)

WOBC (Wisconsin Original Breast Cancer dataset)

WDBC (Wisconsin Diagnostic Breast Cancer dataset)

NKI (Netherlands Cancer Institute breast cancer dataset)

Contents

List of Figures

Chapter 1

Introduction

One in nine women is expected to be diagnosed with breast cancer during her life. In 2013, an estimated $23,800$ Canadian women will be diagnosed with breast cancer and $5,000$ will die of it. Making decisions about the treatment for a patient is difficult since it depends on various clinical features, genomic factors, and pathological and cellular classification of a tumor.

In this research, we propose a probabilistic graphical model for prognosis and diagnosis of breast cancer. Probabilistic graphical models are suitable for making decisions under uncertainty from data with missing attributes and noisy evidence. Using this model, we may enter the results of different tests (e.g. estrogen and progesterone receptor test and HER2/neu test), gene expression profile (microarray data), and clinical traits (e.g. womans age, general health, menopausal status, stage of cancer, and size of the tumor) to the model and answer to following questions.

- What is the chance of survival?
- How likely is it that the cancer will extend in the body (distant metastasis)?

- How likely is that the cancer comes back (local or regional recurrence)?
- How promising is a treatment? For example, how likely metastasis is, and how likely recurrence is for a new patient if certain treatment e.g. surgical removal, radiation therapy, hormone therapy, or chemotherapy is applied.
- What is the type of the tumor (classification)?

The structure of the chapter is as follows. Sectin 1.1 presents an introduction to Bayesian networks (BN). Section 1.2 discusses reasoning patterns in BNs by an example. In Sectin 1.3, we discuss various Conditional Probability Distributaions (CPDs) that can be assigned to a node in a BN. Section 1.4 describes related work. Finally, in Section 1.5, we give the outline of this thesis.

1.1 Bayesian Networks

A Bayesian network (BN) is a probabilistic model that represents a set of random variables and their conditional dependencies by a directed acyclic graph. In such a network, each node is a continuous or a discrete random variable. There is an edge between two nodes, if they are conditionally dependent. Nodes that are not connected correspond to variables that are conditionally independent of each other. For each node, there is a probability density function that takes as input a set of values for the node's parent variables and gives the probability of the variable represented by this node.

For instance, a BN can represent the probabilistic connections between symptoms and diseases. Having symptoms, the BN can be applied to compute the probabilities of the existence of a disease.

Formally, a Bayesian network is a directed acyclic graph G, whose nodes represent random variables $X_1, X_2, ..., X_n$, and for each node X_i, a Conditional Probability Distribution (CPD) is assigned, which defines the probability distribution of a node given its parents. That is $P(X_i \mid U(X_i))$, where $U(X_i)$ are parents of node X_i in graph G. The BN represents a joint distribution via the chain rule for BNs [3, 11]:

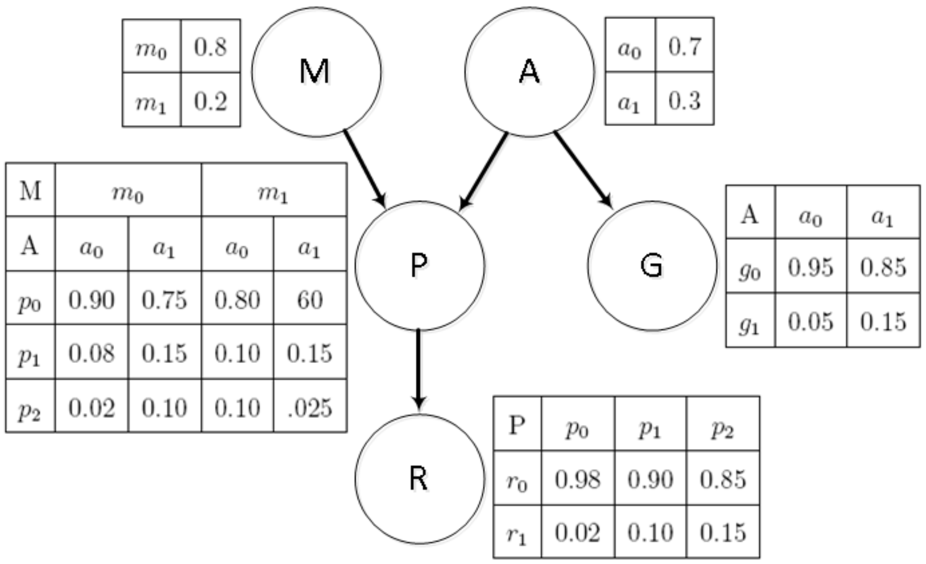
$$P(X_1, ..., X_n) = \prod_{i=1}^{n} P\big(X_i \mid U(X_i)\big).$$

Further, BN is a legal probability distributaion. That is,

$$P(X_1 = x_1, ..., X_n = x_n) = P(x_1, ..., x_n) \geq 0 \quad \text{and} \quad \sum_{x_1} ... \sum_{x_n} P(x_1, ..., x_n) = 1.$$

Example 1.1.1. To explain the inference and learning in BN, we use a real example through this text. A posterior vitreous detachment (PVD) occurs in the eye, when the vitreous separates from the retina. Age and myopia (near sightedness) are two causes and risk factors for PVD. It is rare in young people with normal vision. However, people with myopia (greater than five diopters) are at higher risk of PVD at all ages. PVD does not usually cause sight loss. Although it can cause some problems, such as floaters and little flashes of light, in many cases, PVD does not lead to serious vision problems. However, there is a small risk that PVD may lead to a retinal tear or detachment, which needs urgent attention. Age is also a risk factor for glaucoma, an eye disease, in which the optic nerve is damaged in a certain pattern. Glaucoma usually affects old people.

Figure 1.1 shows a simple BN for this example. We construct the structure (nodes



m_0	0.8
m_1	0.2

a_0	0.7
a_1	0.3

M	m_0		m_1	
A	a_0	a_1	a_0	a_1
p_0	0.90	0.75	0.80	60
p_1	0.08	0.15	0.10	0.15
p_2	0.02	0.10	0.10	.025

A	a_0	a_1
g_0	0.95	0.85
g_1	0.05	0.15

P	p_0	p_1	p_2
r_0	0.98	0.90	0.85
r_1	0.02	0.10	0.15

Figure 1.1: A simple Bayesian network for Example 1.1.1

and the dependencies) of the network using the above facts, although there are algorithms to extract the structure of a BN from data automatically [8]. We briefly explain some of these algorithms in Chapter 2. In this example, all nodes are discrete. We use two states for each of the nodes myopia (M), age (A), glaucoma (G), and retinal tear (R). Myopia has two states m_0 = (no myopia), and m_1 = (with myopia). Age has two states a_0 = (young), and a_1 = (old). Glaucoma has two states g_0 = (no glaucoma), and g_1 = (with glaucoma). Retinal tear has also two states r_0 = (no retinal tear), and r_1 = (with retinal tear). For PVD (P), we use three states p_0 = (no PVD), p_1 = (moderate PVD), and p_2 = (chronic PVD). For example, $(m_0, a_1, p_2, g_0, r_1)$ is an old person without myopia and glaucoma, but with chronic PVD and retinal tear. As another example $(m_1, a_0, p_0, g_1, r_0)$ represents a young person without PVD and retinal tear, but with myopia and glaucoma.

For each node a CPD is assigned. Each CPD takes as input a set of values for the node's parent variables and gives the probability of the variable represented by the corresponding node. For example, the first value in the CPD of node R shows that the propability of having no retinal tear, given the sample has no PVD is, $P(r_0 \mid p_0) = 0.98$. As another example, we may see from the CPD of node P that the probability of having chronic PVD, given the sample is old and has myopia, is $P(p_2 \mid a_1, m_1) = 0.25$.

The paprameters of the network, that is CPDs, can be extracted from a data set of samples. There are efficient algorithms for this purpose, and we discuss them in Chapter 2, but for now we want to show how this network can be useful by obtaining some reasoning patterns.

1.2 Reasoning Patterns

The most common reasoning pattern is causal reasoning, which goes from top to bottom in the BN. For example, we may compute the effect of the age on retinal tear. For this purpose, we first compute the probabilty of having retinal tear, $P(r_1)$, for a randomly selected example. Using the conditional probability formula and CPD of node R in Figure 1.1:

$$\begin{aligned} P(r_1) &= P(r_1 \mid p_0)P(p_0) + P(r_1 \mid p_1)P(p_1) + P(r_1 \mid p_2)P(p_2) \\ &= 0.02P(p_0) + 0.1P(p_1) + 0.15P(p_2). \end{aligned}$$

Here, we need to compute $P(p_0)$, $P(p_1)$, and $P(p_2)$. To compute $P(p_0)$, we use the conditional probability formula and the independency of nodes M and A:

$$\begin{aligned}P(p_0) = {} & P(p_0 \mid m_0, a_0)P(m_0)P(a_0) + P(p_0 \mid m_0, a_1)P(m_0)P(a_1) \\ & + P(p_0 \mid m_1, a_0)P(m_1)P(a_0) + P(p_0 \mid m_1, a_1)P(m_1)P(a_1) \\ \approx {} & 0.832.\end{aligned}$$

Thus, the the probabilty of having no PVD for a randomly selected sample is 0.832. Similarly, we obtain $P(p_1) = 0.1038$ and $P(p_2) = 0.0642$. Therefore,

$$P(r_1) = 0.02P(p_0) + 0.1P(p_1) + 0.15P(p_2) \approx 0.03665.$$

Thus, the probabilty of having retinal tear for a randomly selected sample is 0.03665.

Now, suppose that we wish to see how age (cause) affects the probability of having retinal tear. We want to compute the probability of retinal tear given the sample is a young person (evidence):

$$\begin{aligned}P(r_1 \mid a_0) &= P(r_1 \mid p_0)P(p_0 \mid a_0) + P(r_1 \mid p_1)P(p_1 \mid a_0) + P(r_1 \mid p_2)P(p_2 \mid a_0) \\ &= 0.02 \times 0.88 + 0.1 \times 0.084 + 0.15 \times 0.036 \\ &= 0.0314.\end{aligned}$$

As a result, the probability of retinal tear not surprisingly goes down compared with 0.03665, since now we know that the sample is not an old person, and age is one of the risk factors of retinal tear.

As another example of causal reasoning with additional evidence, we may compute the probability of retinal tear given the sample is a young person, but with myopia. In this case, we have two evidences: the sample is a young person, and he or she has myopia. Formally, we should compute $P(r_1 \mid a_0, m_1)$. Using the conditional probability formula and CPD of nodes P and R, we have:

$$\begin{aligned} P(r_1 \mid a_0, m_1) &= P(r_1 \mid p_0)P(p_0 \mid a_0, m_1) + P(r_1 \mid p_1)P(p_1 \mid a_0, m_1) \\ &\quad + P(r_1 \mid p_2)P(p_2 \mid a_0, m_1) \\ &= 0.02 \times 0.8 + 0.1 \times 0.1 + 0.15 \times 0.1 = 0.041. \end{aligned}$$

As we can see, the probabilty increases compared with 0.0314, because now we know that the sample has myopia.

We can also do evidential reasoning. It goes from bottom to top in the BN. For example, from the CPD of node M, we know that the probabilty of having myopia for a randomly selected sample is 0.2, that is $P(m_1) = 0.2$. Now, we want to compute the probablity of having myopia given the sample has chronic PVD. Using the conditional probability formula and CPD of nodes P, M, and A we have:

$$\begin{aligned} P(m_1 \mid p_2) &= \frac{P(m_1, p_2)}{P(p_2)} = \frac{P(m_1, p_2, a_0) + P(m_1, p_2, a_1)}{P(p_2)} \\ &= \frac{P(p_2 \mid m_1, a_0)P(m_1)P(a_0) + P(p_2 \mid m_1, a_1)P(m_1)P(a_1)}{P(p_2)} \\ &= 0.4517. \end{aligned}$$

The probabiltity of having myopia, given the sample has chronic PVD, not surprisingly increases compared with the probabilty of having myopia for a randomly selected sample which is 0.2 (see Figure 1.1), since we know that the sample has chronic PVD.

1.3 CPD Types

There are various ways to represent CPDs, based on whether the corresponding node and its parents are continuous or discrete. Here, we discuss three types of CPDs, which are more useful than the others in applications: table CPD, Gaussian CPD and softmax CPD [11].

A table CPD is a multinomial distribution, and it can be represented by a table. The number of parameters in this type of CPD is exponential with respect to the number of parents. This may lead to poor results for predicting and classification, especially if the size of the traning dataset is small. In many applications, a simple BN with few edges is more efficient (for correct predicting and classification) than a complex one with many edges, even if the structure of the BN is not completely correct. Table CPDs are used for discrete nodes with discrete parents.

A Gaussian CPD is suitable for a continous node with discrete and continous parents. Suppose that Y is a contious node. If it has no parents, its CPD is simply a Gaussian distribution with mean μ and variance σ, that is $Y \sim N(\mu, \sigma^2)$. If Y has a continous parent X, the CPD of it is defined as:

$$Y \mid X = x \sim N(\mu + wx, \sigma^2),$$

where w is a weight. We may extend this definition, if there are more than one continous parents, by assigning different weights to each parent as:

$$Y \mid (X_1 = x_1, ..., X_n = x_n) \sim N(\mu + \sum_j w_j x_j, \sigma^2).$$

If Y has a discrete parent D, the CPD of it is defined as $Y \mid D = d_i \sim N(\mu_i, \sigma_i^2)$, where μ_i and σ_i^2 are the mean and variance associated with d_i, ith assignment of discrete variable D. If Y has a continous parent X and a discrete parent D, the CPD of it is defined as folows:

$$Y \mid (X = x, D = d_i) \sim N(\mu_i + w_i x, \sigma_i^2),$$

where μ_i, σ_i^2 and w_i, are mean, variance, and weight associated with d_i, respectively.

A discrete node with a continous parent can be represented by a CPD, which is called softmax CPD defined as follows:

$$P(D = d_i \mid X = x) = \frac{\exp{(w_i x + b_i)}}{\sum_j \exp{(w_j x + b_j)}}$$

Here b_i and w_i, the parameters of the softmax node D, are bias and weight of x associated with d_i, respectively. It ensures the values are between 0 and 1, which is necessary for a probability density function.

1.4 Related Work

Previous work mostly relies on clinical data [12, 19, 17]. In our opinion, since cancer is a genetic disease, the integration of the genomic (microarray) and clinical data can improve the accuracy of the model for prognosis and diagnosis. However, the microarray data is high dimensional. It consists of around $25,000$ variables per patient. Moreover, structure learning and parameter learning for probabilistic graphical models require a significant amount of computations. The number of possible structures is also super-exponential with respect to the number of variables. For instance, as we will discuss in the next chapter, there are 4.2×10^{18} possible structures with just 10 variables [11, 6]. Another difficulty is missing data, a common problem in medical datasets. A good solution to this problem is the Expectation Maximization algorithm which estimates the value of missing attributes and then re-estimates the parameters using complete data. We discuss this algorithm in the next chapter.

Recently, several works reported using microarray technology for studying various types of cancer. Bhattacharjee et al. [2], Singh et al. [16], and Vijver et al. [18] used microarray technology for cancer study. Gevaert et al. [9] proposed a simple BN for prognosis of breast cancer by integrating clinical and genomic data. They selected a few genes (variables) based on the correlation between the expression values of the genes with the classification outcomes. This is due to the fact that applying all genes will lead to curse of dimensionality and small sample size problem for structure and parameter learning algorithms. Then, they integrated these variblaes with the clinical data to construct a BN. However, selecting a few genes may lead to loss of genomic information, which is useful for classification and prediction.

In our method, we avoid this problem by appling an efficient dimensionality reduction (manifold learning) algorithms to the genomic data. Figure 1.2 illustrates the proposed method. The number of clinical features in the real-world datasets is usually less than 30. Thus, there is no need to apply dimensionality reduction to the

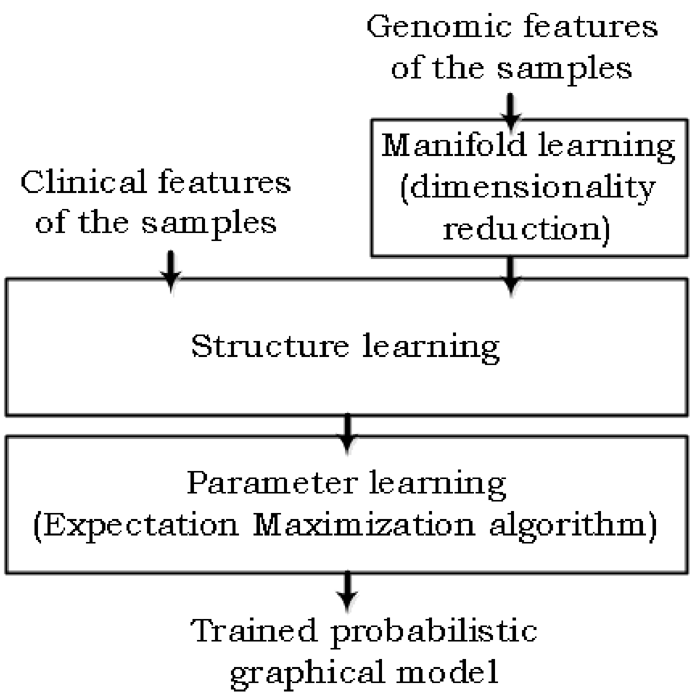

Genomic features of the samples
Manifold learning (dimensionality reduction)
Clinical features of the samples
Structure learning
Parameter learning (Expectation Maximization algorithm)
Trained probabilistic graphical model

Figure 1.2: The proposed method

clinical data. After applying the dimensionality reduction algorithm to the genomic data, we have d genomic features for each sample, where $d << 25,000$ (in our experiments $d = 30$). Then, we aggregate the clinical and the features resulted from the dimensionality reduction algorithm to create a BN. We use a node for each feature. To create the BN, we need to extract the structure and parameters of the model. An efficient algorithm for parameter learning that can tackle the missing data problem is the Expectation Maximization (EM) algorithm [11]. We discuss this algorithm in Chapter 2.

1.5 Thesis Outline

The structure of this thesis is as follows. In Chapter 2 we discuss, given a dataset, how to extract the parameters and structure of a BN. Chapter 3 explaines manifold learning and dimensionality reduction techniques. Chapter 4 reports our experimental results, and explaination of the datasets. Finally, Chapter 5 discusses conclusions and future research directions.

Chapter 2

Parameter and Structure Learning in Bayesian Networks

In this chapter we explain, given a dataset and the structure of a BN, how to extract the parameters of CPDs. As we discussed in the previous chapter, there are various types of CPD. For table CPDs, we must extract entries of the table. For Gaussian and soft-max CPDs, we must compute the parameters of the conditional probability density function like mean, variance, and weights.

The structure of this chapter is as follows. Section 2.1 explains the principle of Maximum Likelihood Estimation (MLE) and applies it to extract the parameters of a BN given a data set [11]. Section 2.2 discusses Bayesian parameter estimation [11]. Finally, in Section 2.3, we explain the Expectation Maximization (EM) algorithm to deal with the missing data problem, which occurs when no data value is available for some traits of a sample, a typical challenge in many applications [11]. For example, when we work with a medical dataset, the value of some traits of one or more patients may not be available. Finally, in Section 2.4, we briefly explain common structure learning algorithms for BNs [6].

2.1 Maximum Likelihood Estimation

Suppose that $D = \{x_1, x_2, ..., x_n\}$ is a sample set of n independent and identically distributed (IID) observations coming from a distribution with an unknown probability density function $f = f(\theta)$ that belongs to a given family of distributions, where θ is a vector of parameters for this family. The goal of Maximum Likelihood Estimation (MLE) [11] is to find θ that predicts D well. The quality of the prediction is the likelihood of D, for a given θ. Formally in MLE, the goal is to maximize the following *likelihood function*:

$$L(\theta : D) = f(D \mid \theta) = \prod_{i=1}^{n} f(x_1, ..., x_n \mid \theta) = \prod_{i=1}^{n} f(x_i \mid \theta). \tag{2.1}$$

Example 2.1.1. Let P be a Bernoulli distribution:

$$P(X = x) = \begin{cases} \theta & \text{if } x = 1 \\ 1 - \theta & \text{if } x = 0, \end{cases}$$

where θ is the probability of success. Suppose that $D = \{x_1, x_2, ..., x_n\}$ is a sample set of n IID tosses of a coin coming from P. Namely, tosses are independent and coming from the same distribution. We want to estimate the parameter θ which is the probability of obtaining a head (H) as the outcome. Thus, $1-\theta$ is the probability of obtaining a tail (T).

For example, if $D = \{H, T, H, T, T\}$ then the likelihood function is as follows:

$$L(\theta : D) = \prod_{i=1}^{n} f(x_i \mid \theta) = \theta^2(1-\theta)^3.$$

In general, for the Bernoulli distribution with a parameter θ:

$$L(\theta : D) = \theta^{N_H}(1-\theta)^{N_T}, \tag{2.2}$$

where N_H and N_T, are number of heads and tails. Often, it is more convenient to maximize the logarithm of the likelihood function, called log-likelihood function. For (2.2) we obtain

$$l(\theta : D) = \log(L(\theta : D)) = N_H \log(\theta) + N_T \log(1-\theta).$$

By differentiating and setting the derivative to zero, we have:

$$\frac{N_H}{\theta} - \frac{N_T}{1-\theta} = 0.$$

Thus, as an estimation of θ, the MLE gives:

$$\widehat{\theta} = \frac{N_H}{N_H + N_T}.$$

Since $N_H = 2$ and $N_T = 3$,

$$\widehat{\theta} = \frac{N_H}{N_H + N_T} = \frac{2}{5} = 0.4,$$

is a value for θ, where the likelihood function achieves its maximum (see Figure 2.1).

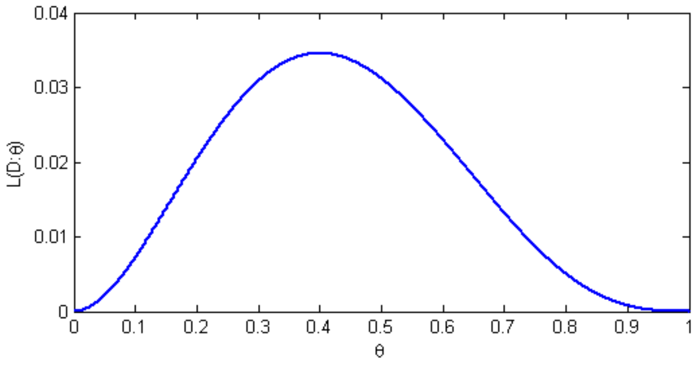


Figure 2.1: Likelihood function $L(\theta : D) = \theta^2(1-\theta)^3$ versus θ

As a result, the MLE gives $\widehat{\theta} = 0.4$, as an estimation of θ using the above sample set D.

2.1.1 Sufficient Statistics

An important concept in the context of MLE is the notion of sufficient statistics. In the coin example, as we can see from (2.2), the likelihood function depends only on N_H and N_T. That is, N_H and N_T are sufficient to define the likelihood function, and these two parameters are sufficient statistics for Bernoulli distribution with a parameter θ. More formally, the definition of sufficient statistics is as follows.

Definition 2.1.1. A function $s(D) = \sum_{x_i \in D} s(x_i)$ is a sufficient statistic from instances to a vector in $\mathbb{R}^k$ if, for any two datasets D and D' and any θ, we have

$$s(D) = s(D') \Rightarrow L(\theta : D) = L(\theta : D').$$

Example 2.1.2. In the coin example, we define $s(x) = (1, 0)$ if x is head, otherwise $s(x) = (0, 1)$. Thus, $s(D) = \sum_{x_i \in D} s(x_i) = (N_H, N_T)$ are sufficient statistics.

Example 2.1.3. Suppose that $D = \{x_1, x_2, ..., x_n\}$ is a set of n IID samples, coming from a multinomial distribution with parameters $(\theta_1, ..., \theta_k)$, where θ_j is the probability of occurring of the jth outcome.

We may simply verify that, with $s(x) = e_j$ if x be the jth outcome, $s(D) = \big(N_1, ..., N_k\big)$ are the sufficient statistics, where N_j is the number of times that x_i's are the jth outcome in D, and e_j is the jth unit vector of dimension k.

Namely, $N_1, ..., N_k$ are the sufficient statistics for multinomial distribution with parameters $(\theta_1, ..., \theta_k)$. By computing the likelihood function we have:

$$L\big((\theta_1, ..., \theta_k) : D\big) = f\big(D \mid (\theta_1, ..., \theta_k)\big) = \prod_{i=1}^{n} f\big(x_i \mid (\theta_1, ..., \theta_k)\big) = \prod_{j=1}^{k} {\theta_j}^{N_j}.$$

By applying MLE, we obtain $\widehat{\theta_j} = N_j / \sum_{i=1}^{k} N_i$ as an estimator for θ_j.

Example 2.1.4. A random variable X has Gaussian distribution with mean μ and variance σ^2, denoted by $X \sim N(\mu, \sigma^2)$, if its probability density function has the form:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

This can be rewritten as

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-x^2\frac{1}{2\sigma^2} + x\frac{\mu}{\sigma^2} - \frac{\mu^2}{\sigma^2}\right).$$

Suppose that $D = \{x_1, x_2, ..., x_n\}$ is a set of n IID samples coming from a Gaussian distribution with mean μ and variance σ^2. According to Definition 2.1.1, we can simply verify that, with $s(x) = (1, x, x^2)$, $s(D) = \big(n, \sum_i x_i, \sum_i x_i^2\big)$ is the sufficient statistic for Gaussian distribution. By computing the log-likelihood function and setting its derivative to zero, MLE gives

$$\widehat{\mu} = \frac{1}{n}\sum_i x_i \quad \text{and} \quad \widehat{\sigma} = \sqrt{\frac{1}{n}\sum_i (x_i - \hat{\mu})^2}$$

as estimators for the mean and variance of a Gaussian distribution.

2.1.2 Maximum Likelihood Estimation for BN

In Section 2.1, we defined the principle of maximum likelihood estimation for a single parameter. We now consider a more complex case for a given BN structure.

First, consider the simple case in which we have only two nodes X and Y, and variable X is the parent of random variable Y. Suppose that

$$D = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$$

is a sample set of n IID observations. According to (2.1),

$$\begin{aligned} L(\theta : D) &= \prod_{i=1}^{n} P(x_i, y_i : \theta) = \prod_{i=1}^{n} P(x_i : \theta) P(y_i \mid x_i : \theta) \\ &= \prod_{i=1}^{n} P(x_i : \theta) \prod_{i=1}^{n} P(y_i \mid x_i : \theta) = \prod_{i=1}^{n} P(x_i : \theta_x) \prod_{i=1}^{n} P(y_i \mid x_i : \theta_{y|x}), \end{aligned}$$

where θ_x, and $\theta_{y|x}$ are likelihood of variables X and X given Y, respectively. The two products in the last equation are called *local likelihood*.

More generally, for m nodes $X_1, ..., X_m$ of a given BN structure, we have

$$\begin{aligned} L(\theta : D) = \prod_{i=1}^{n} P(x_i : \theta) = \prod_{i=1}^{n}\prod_{j=1}^{m} P\big(x_{j_i} \mid y_{j_i} : \theta_j\big) \\ = \prod_{i=1}^{n}\prod_{j=1}^{m} P\big(x_{j_i} \mid y_{j_i} : \theta_j\big) = \prod_{j=1}^{m} L_j(D : \theta_j), \end{aligned}$$

where $x_i = (x_{1i}, ..., x_{mi})$, x_{j_i} is ith sample of random variable X_j, and Y_j is the parent of X_j, that is $Y_j = U\big(X_j\big)$.

If we have table CPDs, that is all variables are discrete, we may further decompose the previous equation into some buckets as follows:

$$\begin{aligned} L(\theta : D) = \prod_{i=1}^{n} f(x_i \mid y_i : \theta_{x|y}) = \prod_{x,y} \prod_{i:x_i=x,u:u_i=u}^{m} P\big(x_i \mid y_i : \theta_{x|y}\big) \\ = \prod_{x,y} \prod_{i:x_i=x,u:u_i=u}^{m} \theta_{x|y} = \prod_{x,y} {\theta_{x|y}}^{N(x,y)}, \end{aligned}$$

where $N(x, y)$ is the number of (x, y) in the dataset. By applying MLE we obtain

$$\widehat{\theta}_{x|y} = \frac{N(x,y)}{\sum_z N(z,x)} = \frac{N(x,y)}{N(y)} \tag{2.3}$$

For instance, in Example 1.1.1, if we have 100 samples with chronic PVD in total, and 15 of them have retinal tear too, we have $N(p_2) = 100$ and $N(r_1, p_2) = 15$. Then, by (2.3), we obtain $\widehat{\theta}_{r_1|p_2} = P(r_1 \mid p_2) = 0.15$, as an estimation for parameter $P(r_1 \mid p_2)$ in CPD of node R.

2.2 Bayesian Parameter Estimation

In the previous section, we discussed Maximum Likelihood Estimation (MLE) for parameter estimation in BNs. The MLE method supposes that given a fixed parameter vector θ, the n observations are independent and identically distributed (IID). For instance, in the coin example, we suppose that given a fixed θ, tosses are independent.

In this section, we explain an alternative approach called *Bayesian parameter estimation* [11]. This method treats the parameters as random variables. For instance, suppose that in the coin example, the parameter θ is a random variable such that $\theta \in [0, 1]$. Since θ is unknown, tosses are not marginally independent and each new toss give some information about θ. As a result, using the chain rule for BNs,

$$P\big(X_1, ..., X_n, \theta\big) = P\big(X_1, ..., X_n \mid \theta\big)P\big(\theta\big) = P(\theta)\prod_{i=1}^{n} P(X_i \mid \theta) = P\big(\theta\big)\theta^{N_H}(1-\theta)^{N_T},$$

The probability of θ, $P\big(\theta\big)$, is called *prior*. By a simple application of the Bayes rule, we may compute the so called *posterior* over parameter θ given the data:

$$P\big(\theta \mid X_1, ..., X_n\big) = \frac{P\big(X_1, ..., X_n \mid \theta\big)P\big(\theta\big)}{P\big(X_1, ..., X_n\big)}. \tag{2.4}$$

The probability of the data in the denominator of (2.4) is just a constant since it does not depend on θ. A common situation is when the likelihood, $P\big(X_1, ..., X_n \mid \theta\big)$, is a multinomial distribution over k values and the prior, $P\big(\theta\big)$, is a *Dirichlet distribution* with a set of hyperparameters $(\alpha_1, ..., \alpha_k)$. Intuitively, α_i represents the number of the data with values x_i which we have seen so far. In this case, we have the following form for $P(\theta)$:

$$P(\theta) = \frac{1}{c}\prod_{i=1}^{k} {\theta_i}^{\alpha_i - 1}, \quad \text{where} \quad c = \frac{\prod_{i=1}^{k}\Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{k}\alpha_i)}. \tag{2.5}$$

Here, $\Gamma(x) = \int_0^\infty t^{(x-1)}\mathrm{e}^{-t}\,\mathrm{d}t$, is the gamma function. By (2.4) and (2.5), we see that if prior is a Dirichlet distribution with a set of hyperparameters $(\alpha_1, ..., \alpha_k)$ and likelihood is multinomial, then the posterior is also a Dirichlet distribution with a set of hyperparameters $(N_1 + \alpha_1, ..., N_k + \alpha_k)$, where $N_1, ..., N_k$ are data counts corresponding to the k values.

We may also make a prediction over the value of a random variable X that depends on parameter θ:

$$P(X) = \int_\theta P(X \mid \theta)P(\theta)\mathrm{d}\theta.$$

Using integration by part and the properties of the integral of polynomials, we obtain the following equation for the probability of obtaining value x_i given parameter θ:

$$P(X = x_i \mid \theta) = \frac{1}{c}\int_\theta \theta_i \prod_j {\theta_j}^{\alpha_j - 1}\mathrm{d}\theta = \frac{\alpha_i}{\sum_j \alpha_j}. \tag{2.6}$$

Since we know the posterior is also a Dirichlet distribution with a set of hyperparameters $(N_1 + \alpha_1, ..., N_k + \alpha_k)$, by (2.6), we obtain:

$$P(X_{N+1} = x_i \mid \theta, X_1, ..., X_N) = \frac{\alpha_i + N_i}{\sum_{i=1}^{k}\alpha_i + \sum_{i=1}^{k}N_i}. \tag{2.7}$$

In Example 2.1.1, if $D = \{H, T, H, T, T\}$, then $N_1 = N_H = 2$ and $N_2 = N_T = 3$. Moreover, if θ has a uniform distribution on $[0, 1]$, then $\alpha_1 = 1$ and $\alpha_2 = 1$, since the uniform distribution is the same as a Dirichlet distribution with a set of hyperparameters $(\alpha_1, \alpha_2) = (1, 1)$. Thus, by (2.7), we can compute the probability of getting a head in 6th toss given the data as follows:

$$P\big(X_6 = H \mid X_1 = H, X_2 = T, X_3 = H, X_4 = T, X_5 = T\big) = \frac{\alpha_1 + N_1}{\alpha_1 + \alpha_2 + N_1 + N_2} = \frac{3}{7}.$$

However, as we discussed before (see Figure 2.1), with the MLE method, the probability of obtaining a head is 0.4.

2.3 Expectation Maximization for Missing Data

In many real-world applications, we may not have the value of some features for some samples. For instance, in a breast cancer dataset, we may not have the value of some tests for a patient. In this section, we address the problem of missing data by explaining the Expectation Maximization (EM) algorithm [11].

The intuition behind the EM algorithm is that the parameter estimation is a well-defined problem, if we have complete set of data. Conversely, if we have a full set of parameters, then we may compute the probability of missing data given the parameters, by performing inference as we discussed in Chapter 1. As a result, we have two sets of things that we need to estimate and each of them gives the other: the parameters and the value of missing data.

The EM algorithm is basically an iterative algorithm, which starts with some set of parameters and try to estimate the value of the missing data. Then, it uses the data to re-estimate the parameters. This process continues until it converges.

More precisely, the EM algorithm selects a starting point for parameters, and then it iterates the following two steps:

- Expectation step: estimate (inference) the value of missing data using current parameters.
- Maximization step: re-estimate the parameters using the complete data by the MLE method.

2.4 Structure Learning

Structure learning algorithms can be divided into two main methods: constraint-based methods and search-and-score methods. The constraint-based methods start with a complete graph and try to remove edges, if conditional independencies are detected. However, in the search-and-score methods, which are more common, the algorithm searches among possible solutions to find an approximation to the best directed acyclic graph.

Due to the fact that the number of directed acyclic graphs is super-exponential with respect to the number of nodes, it is impractical to search through the whole space of the solutions. For instance, there are 4.2×10^{18} directed acyclic graphs with just 10 nodes. As a result, we have to use a local search algorithm such as the K2 algorithm [3, 11].

The K2 algorithm is basically a greedy search algorithm that starts with no parents for each node. It then adds the parents one by one. The parent that lead to the most increase in the score of the resulting structure will add first. The algorithm stops when adding the next parent cannot increase the score [11]. For structure learning, we applied a structure learning software developed by Cassio P. de Campos, Zhi Zeng, and Qiang Ji [8], which can be downloaded from `http://www.ecse.rpi.edu/`

`~cvrl/structlearning.html`. They used a branch and bound algorithm that uses structural constraints with the data.

Chapter 3

Dimensionality Reduction and Manifold Learning

As we discussed in the previous chapter, parameter and structure learning algorithms may lead to poor results, if the number of variables is high with respect to the number of samples. Since we want to use genomic features (around $25,000$ features for each patient) to model breast cancer, the number of variables is too high with respect to number of samples (around 2000 samples) in our application.

In this chapter, we discuss dimensionality reduction and manifold learning algorithms. The most common dimensionality reduction algorithm is Principal Component Analysis (PCA) [10]. We explain PCA in Section 3.1. Then, in Section 3.2, we explain more advanced manifold learning techniques to reduce the dimensionality of the samples, while keeping useful information [5, 13].

3.1 Principal Component Analysis

Suppose that $D = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ is a dataset of n samples, where $\boldsymbol{x}_i$'s are m-dimensional vectors, and we wish to reduce the dimensionality of the samples of this dataset from m to d ($d < m$). Without loss of generality, we may assume that $\sum \boldsymbol{x}_i = \mathbf{0}$, since we can always center the data by subtracting the mean of the samples. Moreover, suppose that X is an $n \times m$ matrix, where its ith row is $\boldsymbol{x}_i$.

The goal of PCA is to find directions that maximize the variance of the projected dataset. That is, we have the objective function

$$\max_{V} \operatorname{var}(XV), \tag{3.1}$$

where V is an orthogonal $m \times d$ matrix, and d is the dimensionality of the new subspace. If $d = 1$, the problems is to find a unit-length vector $\boldsymbol{v}$ that maximizes the variance of the projected dataset:

$$\max_{\|\boldsymbol{v}\|=1} \operatorname{var}(X\boldsymbol{v}) = \max_{\|\boldsymbol{v}\|=1} \sum_{i=1}^{n} \boldsymbol{v}^\top \boldsymbol{x}_i \boldsymbol{x}_i^\top \boldsymbol{v} = \max_{\|\boldsymbol{v}\|=1} \boldsymbol{v}^\top \Big(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top \Big) \boldsymbol{v} = \max_{\|\boldsymbol{v}\|=1} \boldsymbol{v}^\top X^\top X \boldsymbol{v}.$$

The unit-length vector $\boldsymbol{v}$ that maximizes the above optimization problem is the eigenvector corresponding to the largest eigenvalue of the $m \times m$ matrix $X^\top X$ (for more details see [10]).

If $d > 1$, V which maximize the objective function (3.1) is an $m \times d$ matrix whose columns are the d eigenvectors corresponding to the largest eigenvalues of $X^\top X$. The samples can be projected onto the new subspace using $\boldsymbol{y}_i = V^\top \boldsymbol{x}_i$, where $\boldsymbol{y}_i$ is a d-dimensional vector representing $\boldsymbol{x}_i$ in the new subspace.

3.2 Manifold Learning

PCA is a linear dimensionality reduction algorithm. That is, it supposes that the data approximately lie on a linear subspace. However, if the data do not have such a property, as we can see in many applications, after projecting the data to the new subspace, the information for classification may be lost. Manifold learning algorithms are non-linear extension of PCA and try to solve the limitation of linearity. They suppose that the data lie on a non-linear manifold rather that a linear subspace.

Definition 3.2.1. A compact set $M \subseteq \mathbb{R}^n$ is a smooth d-dimensional manifold, if there exists a differentiable function with a differentiable inverse $f : M \to \mathbb{R}^d$. This smooth map f is called a coordinate chart, which is an embedding of manifold M into the lower-dimensional space $\mathbb{R}^d$.

Figure 3.1 illustrates a one-dimensional manifold, since we can map the points on the curve into an interval in $\mathbb{R}^1$ by a coordinate chart like $f(x, y, z) = t$, where $x = \sin(t), y = \cos(t)$ and $z = t$. The coordinate chart f is a differentiable with a differentiable inverse function and maps the points on the curve into the interval $[0, 18]$ which is a subset of $\mathbb{R}^1$. Similarly, Figure 3.2 shows a Swiss roll, which is a two-dimensional manifold, since we can map it into a region in $\mathbb{R}^2$ by a coordinate chart f, which unravels it by opening the Swiss roll and mapping it to a subset of $\mathbb{R}^2$.

Given a training dataset $D = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$, where $\boldsymbol{x}_i$'s are m-dimensional vectors, but lie on a d-dimensional manifold M described by $f : M \to \mathbb{R}^d$, manifold learning algorithms try to find $D' = \{\boldsymbol{y}_1, ..., \boldsymbol{y}_n\}$, where $\boldsymbol{y}_i$'s are d-dimensional vectors and $\boldsymbol{y}_i = f(\boldsymbol{x}_i)$. We will describe three of these algorithms in the next subsections.

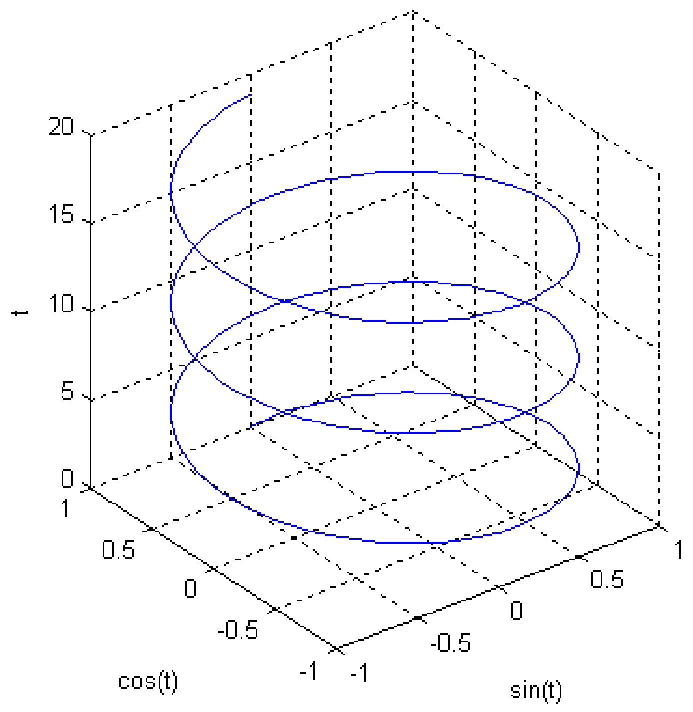


Figure 3.1: A one-dimensional manifold

3.2.1 Isomap

Isometric feature mapping (Isomap) is based on two main assumptions. First, there exists a coordinate chart $f : M \to \mathbb{R}^d$, such that the distances along the manifold M, which are called geodesic distances, are preserved in the new space $\mathbb{R}^d$. That is, for all pairs of training samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ we have:

$$d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\|_2,$$

where, $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the geodesic distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ along M in $\mathbb{R}^n$. Second, Isomap assumes that the geodesic distance between close points along manifold M in the original space is approximately equal to the Euclidean distance between them in

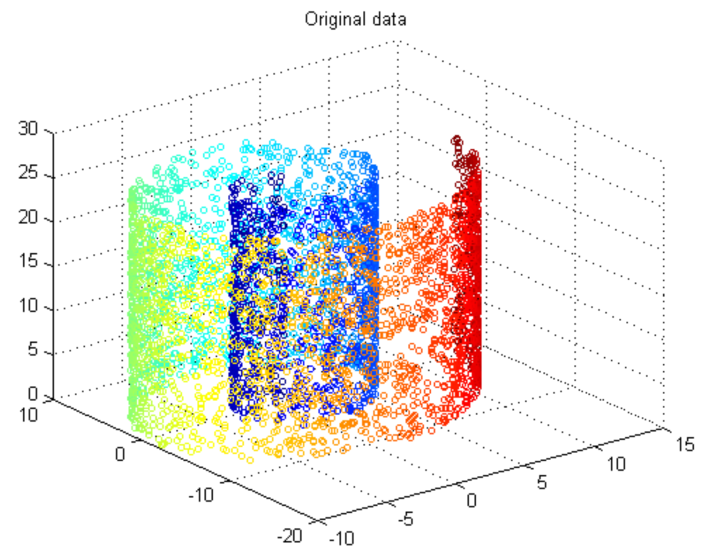


Figure 3.2: A two-dimensional manifold

the original space $\mathbb{R}^n$.

Isomap is composed of two stages: estimating geodesic distances and finding d-dimensional samples, whose inter-point distances (the distances between each pair of points) are compatible with the estimated geodesic distances. To estimate the geodesic distances, Isomap creates a graph with n vertices corresponding to the n training samples $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$.

In this graph, each vertex is connected to its k nearest neighbors, and the weights on the edges of the graph are based on Euclidean distance between samples in $\mathbb{R}^n$. This is due to the fact that according to the second assumption, the geodesic distance between close samples is almost linear and is approximately equal to the Euclidean distance between them. Then, for the vertices which are not connected (are not close to each other), we may use algorithms such as Dijkstra's or Floyd's to estimate the distances between them. This gives us a $n \times n$ matrix G of estimated inter-point geodesic distances.

Given a matrix G of geodesic distances from the first stage, the goal of the second stage is to find a set $D' = \{\boldsymbol{y_1}, ..., \boldsymbol{y}_n\}$ of d-dimensional samples whose inter-point distances is compatible with G. This is due to the fact that Isomap assumes geodesic distances are preserved in the new space $\mathbb{R}^d$. To perform this, there are various algorithms [5]. We explain a common algorithm, which is called *Multidimensional Scaling*.

Multidimensional Scaling first computes $A = -\frac{1}{2}BGB$, where $B = I - \frac{1}{n}D$ and D is a matrix of ones. Matrix A is positive semidefinite if and only if, G is an Euclidean distance matrix. That is, there is a set of points whose inter-point distances are compatible with G. Then, Multidimensional Scaling computes eigenvalue decomposition of A as $A = C\Lambda C^\top$.

If G is exactly a Euclidean distance matrix, then all entries of the diagonal matrix Λ are non-negative, and the desired configuration would be $Y = C\Lambda^{\frac{1}{2}}$. However, since G is an approximation of a Euclidean distance matrix, some of the entries of the diagonal matrix Λ may not be non-negative. Thus, we compute the $m \times d$ matrix $\widehat{\Lambda}$ by selecting d largest non-negative eigenvalues of Λ where $d < m$. Then, we can compute the desired configuration via $Y = \widehat{\Lambda}^{\frac{1}{2}}$. The inter-point distances of rows of $n \times d$ matrix Y, $\boldsymbol{y}_i$'s, are compatible with G.

3.2.2 Locality Preserving Projections

Locality Preserving Projections (LPP) is another popular manifold learning algorithm. It first constructs an adjacency graph using neighborhood information of the training database. Then, LPP computes a transformation matrix, such that the locality information of the original space is preserved in the new subspace. That is, if two samples are close to each other according to a certation criteria, they must be close to each other after the transformation matrix applied.

To construct the adjacency graph, LPP builds a graph with n nodes. Each node represents a training sample. There are two ways to construct the adjacency graph: ϵ-neighborhoods and k-nearest neighbors. In the ϵ-neighborhoods method, nodes i and j are connected, if and only if ${\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2}^2 < \epsilon$, where ϵ is a positive real number. This number is a parameter of the algorithm, which must be determined using the dataset. In the k-nearest neighbors method, however, we put an edge between nodes i and j, if j is among k nearest neighbors of i or vice versa.

There are two method for choosing the weights: heat-kernel and simple-mined. In the heat-kernel method, if there is an edge between nodes i and j, then the corresponding weight is

$$W_{ij} = \exp\left(-\frac{{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2}^2}{t}\right),$$

where $t > 0$ is a parameter of the algorithm. If there is no edge between two nodes, the corresponding weight is 0. The weight W_{ij} shows how much samples i and j are close to each other. In the simple-mined method $W_{ij} = 1$, if nodes i and j are connected, otherwise $W_{ij} = 0$.

After constructing the adjacency graph, to find the transformation matrix, LPP solves the following generalized eigenvector problem:

$$X^\top LX\boldsymbol{v} = \lambda X^\top LXS\boldsymbol{v}, \tag{3.2}$$

where S is a diagonal matrix with diagonal values the row sums of W. That is, $S_{ii} = \sum_j W_{ij}$ and $L = S - W$. The matrix X is an $n \times m$ matrix, whose ith row is

$\boldsymbol{x}_i$ as before.

Suppose that the vectors $\boldsymbol{v}_1, ..., \boldsymbol{v}_d$ are d eigenvector solutions of (3.2), associated with d largest eigenvalues $\lambda_1 > ... > \lambda_d$. Then, the ith columns of desired transformation matrix V is $\boldsymbol{v}_i$, and the low-dimensional representation can be obtained via $\boldsymbol{y}_i = V^\top \boldsymbol{x}_i$.

3.2.3 Locally Linear Embedding

The main idea of Locally Linear Embedding (LLE) algorithm is to see a manifold as a set of overlapping approximately linear patches. We also assume that for every patch, there is a linear coordinate chart from the manifold to $\mathbb{R}^d$. Thus, if the manifold is smooth, we can see it as a set of neighboring small patches that are approximately linear.

Suppose that $N(i)$ is the of set of k-nearest neighbors of the sample $\boldsymbol{x}_i$. The LLE algorithm consists of two steps. In the first step, it attempts to represent the sample $\boldsymbol{x}_i$ as a weighted sum of its k-nearest neighbors. The weights must be determined such that the following objective function be minimized.

$$\|\boldsymbol{x}_i - \sum_{j \in N(i)} W_{ij} \boldsymbol{x}_j\|_2^{\,2}. \tag{3.3}$$

There are two constraints on the weights in (3.3). For each i, $W_{ij} = 0$ if $j \notin N(i)$, and $\sum_{j \in N(i)} W_{ij} = 1$. The first constraint shows that LLE is a local algorithm. The second constraint makes the weights invariant to translation, scale, and rotation. For instance, we can see from the following equation that the weights are invariant to an arbitrary translation $\boldsymbol{\beta}$.

$$\|\boldsymbol{x}_i + \boldsymbol{\beta} - \sum_{j \in N(i)} W_{ij}(\boldsymbol{x}_j + \boldsymbol{\beta})\|_2^2 = \|\boldsymbol{x}_i - \sum_{j \in N(i)} W_{ij}\boldsymbol{x}_j\|_2^2.$$

By Lagrange multipliers, we can find a close form solution to minimize (3.3). The reconstruction weights for each $\boldsymbol{x}_i$ are as follows.

$$W'_i = \frac{\sum_k C^{-1}_{jk}}{\sum_l \sum_m C^{-1}_{lm}},$$

Here, $C_{ij} = (\boldsymbol{x}_i - \boldsymbol{z}_j)^\top(\boldsymbol{x}_i - \boldsymbol{z}_k)$, where $\boldsymbol{z}_j$ and $\boldsymbol{z}_k$ are neighbors of $\boldsymbol{x}_i$. The weight $W_{ij} = 0$ if $j \notin N(i)$, otherwise $W_{ij} = W'_{il}$, where $\boldsymbol{x}_j$ is the lth neighbor of $\boldsymbol{x}_i$.

The second step of the LLE algorithm determines a set of d-dimensional points $D' = \{\boldsymbol{y}_1, ..., \boldsymbol{y}_n\}$ whose geometry is indicated by W. This is like the second step of Isomap, but here d is unknown. To find the d-dimensional points, LLE solves the optimization problem

$$\min \sum_{i=1}^{n} \|\boldsymbol{y}_i - \sum_{j=1}^{n} W_{ij}\boldsymbol{y}_j\|_2^2. \tag{3.4}$$

The solution of (3.4), is an $n \times d$ matrix $Y = [\boldsymbol{y}_1 ... \boldsymbol{y}_n]^\top$, whose columns are the d eigenvectors of $(I - W)^\top(I - W)$, corresponding to the d nonzero eigenvalues [5].

Figures 3.3, 3.4, and 3.5 show different embedding of the Swiss roll in Figure 3.2 using two manifold learning algorithms. In Figure 3.3, Isomap is used to project the Swiss roll into a region in $\mathbb{R}^2$. In Figure 3.4, LPP is applied to map the the Swiss roll into a region in $\mathbb{R}^2$. Finally, in Figure 3.5 LLE is applied to map the the Swiss roll into a region in $\mathbb{R}^2$.

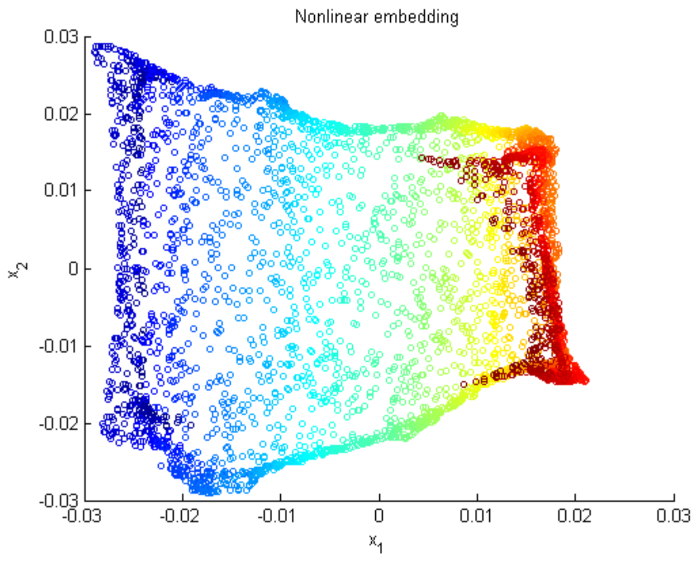


Figure 3.3: Embedding of a Swiss roll into a region in $\mathbb{R}^2$ by Isomap

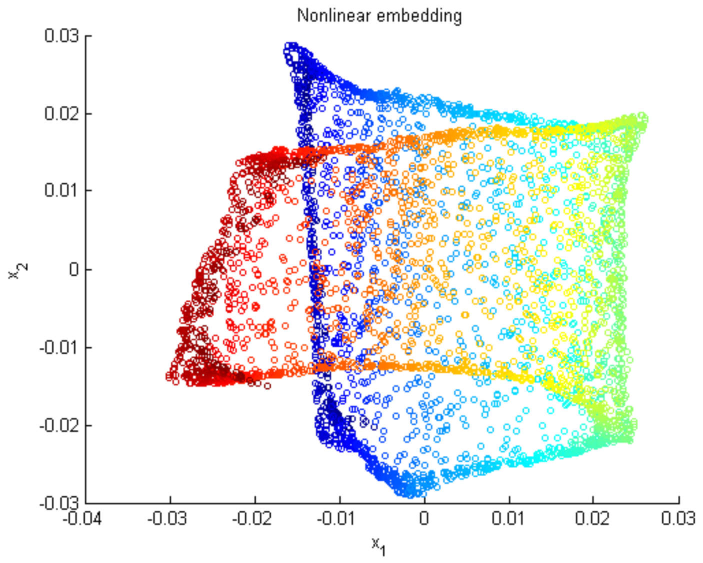


Figure 3.4: Embedding of a Swiss roll into a region in $\mathbb{R}^2$ by LPP

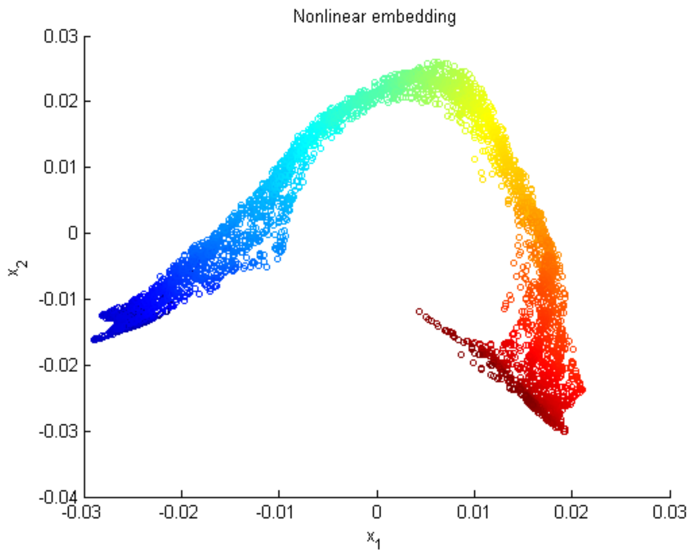


Figure 3.5: Embedding of a Swiss roll into a region in $\mathbb{R}^2$ by LLE

Chapter 4

Experimental Results

This chapter presents our experimental results. The structure of the chapter is as follows. Section 4.1 explains the datasets which we used to train the probabilistic graphical models. The details of the implementation and experimental setup is presented in Section 4.2. Then, in Section 4.3, we explain the methodology and two baseline methods to compare the BN method with some typical classification approaches. Finally, in Section 4.4, we discuss the results.

4.1 Datasets

To evaluate the accuracy and performance of the proposed method, we tested it using several breast cancer datasets: the Netherlands Cancer Institute (NKI) dataset [18], the METABRIC dataset [7], the Ljubljana Breast Cancer dataset, the Wisconsin

Original Breast Cancer (WOBC) dataset from the University of Wisconsin Hospitals [12], and the Wisconsin Diagnostic Breast Cancer (WDBC) dataset from University of Wisconsin, Clinical Sciences Center. The last three datasets can be downloaded from the machine learning repository at University of California, Irvine [1] `http://archive.ics.uci.edu/ml/`.

4.1.1 Netherlands Cancer Institute (NKI) Dataset

This dataset includes expression profiles and clinical traits extracted from 295 breast cancer tumors collected from the tissue bank of the Netherlands Cancer Institute [18]. These data can be downloaded from the Bioinformatics and Statistics Division of Molecular Carcinogenesis, Netherlands Cancer Institute (`bioinformatics.nki.nl/data.php`). This dataset has been used to develop a gene expression signature that is highly predictive for good versus poor prognosis in patients with stage I or stage II breast cancer. Each individual has $24,496$ genes and 19 clinical traits. Each gene is represented by a real number between -1 and 1. The clinical traits are showed in Table 4.1. Some clinical traits are discrete, like regional recurrence at any time during follow-up, while some others are continuous, like tumor diameter.

4.1.2 METABRIC Breast Cancer Dataset

The METABRIC dataset contains clinical traits, expression profiles, copy number variation profiles, and single nucleotide polymorphism genotypes extracted from 1981 breast tumors collected from participants of the METABRIC trial [7]. These data were accessed through Synapse (`synapse.sagebase.org`). Each patient has 24 clinical features. Some of them are age at diagnosis, size, lymph nodes positive, grade,

Trait ID	Trait Description
NKI-TR1	Regional recurrence at any time during follow-up; (0=no, 1=yes)
NKI-TR2	Local recurrence at any time during follow-up; (0=no, 1=yes)
NKI-TR3	Distant metastasis at any time during follow-up; (0=no, 1=yes)
NKI-TR4	Death; (0=no, 1=yes)
NKI-TR5	Survival interval in years between first data of treatment and last date of follow-up
NKI-TR6	Disease free interval in years between first date of treatment and last date of followup
NKI-TR7	Metastasis free interval in years between first date of treatment and date of diagnosis of distant metastisit
NKI-TR8	Interval in years between first date of treatment and date of regional recurrence
NKI-TR9	Interval in years between first data of treatment and local recurrence
NKI-TR10	Chemotherapy; (0=did not receive, 1=received)
NKI-TR11	Hormonal therapy (0=did not receive, 1=received)
NKI-TR12	Surgical removal; (0=no, 1=yes)
NKI-TR13	Tumor diameter in mm
NKI-TR14	Lymph node status from pathology report
NKI-TR15	Histopathology tumor grade
NKI-TR16	Angio invasion
NKI-TR17	Degree of lymphocytic infiltration
NKI-TR18	Age at diagnosis in years
NKI-TR19	Estrogen receptor alpha expression measurement (0=no, 1=yes)

Table 4.1: Description of the clinical trait of the Netherlands breast cancer dataset

histological type, treatment, menopausal status inferred, group, stage, mutation status, and various molecular pathologic classifications.

4.1.3 Ljubljana Breast Cancer Dataset

This breast cancer dataset was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia, provided by M. Zwitter and M. Soklic, `http://archive.ics.uci.edu/ml/datasets/Breast+Cancer`. The instances are described by 9 attributes plus the class label (no-recurrence/recurrence) for 286 instances. There are 9 instances with missing values. In this dataset, 85 instances have recurrence, while 201 instances do not have recurrence. The description of the attributes are presented in Table 4.2.

Trait ID	Trait name	Domain
A0	Class	No-recurrence, recurrence
A1	Age	10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99.
A2	Menopause	Less than 40, greater than 40, premeno
A3	Tumor size	0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59.
A4	Invasive nodes	0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39.
A5	Node-caps	Yes, no
A6	Degrees of malignancy	1, 2, 3
A7	Breast	Left, right
A8	Breast-quad	Left-up, left-low, right-up, right-low, central
A9	Irradiate	Yes, no

Table 4.2: Attributes of the Ljubljana Breast Cancer dataset

4.1.4 Wisconsin Original Breast Cancer (WOBC) Dataset

This breast cancer databases was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg [12]. It contains 699 instances. The number of attributes is 10 plus the class attribute, which is benign or malignant. In this dataset, 458 instances are benign, while 241 instances are malignant. There are 16 instances that contain a missing attribute value. The description of the attributes are presented in Table 4.3.

Trait ID	Trait name	Domain
A0	Clump Thickness	$1-10$
A1	Uniformity of cell size	$1-10$
A2	Uniformity of cell shape	$1-10$
A3	Marginal adhesion	$1-10$
A4	Single epithelial cell size	$1-10$
A5	Bare nuclei	$1-10$
A6	Bland chromatin	$1-10$
A7	Normal nucleoli	$1-10$
A8	Mitoses	$1-10$
A9	Class	Benign, malignant

Table 4.3: Attributes of the Wisconsin Original Breast Cancer (WOBC) dataset

4.1.5 Wisconsin Diagnostic Breast Cancer (WDBC) Dataset

This breast cancer databases was obtained from the University of Wisconsin, Dr. William H. Wolberg, W. Nick Street, and Olvi L. Mangasarian. It can be downloaded from the machine learning repository at University of California, Irvine [1] `http://archive.ics.uci.edu/ml/`. It contains 569 instances. There are ten real value attributes. The description of the attributes are presented in Table 4.4. The mean, standard error, and ”worst” or largest (mean of the three largest values) of these features were computed resulting in 30 attributes. The class attribute is benign or malignant. In this dataset, 357 instances are benign, while 212 instances are malignant. There is no instance containing a missing attribute value.

4.2 Experimental Setup

For dimensionality reduction and manifold learning, we used a free subspace learning library [4], [13], which can be downloaded from `http://www.cad.zju.edu.cn/home/dengcai/Data/DimensionReduction.html`. To create the BN, we need to extract the

Trait ID	Trait name	Domain
A0	Radius	Real value
A1	Texture (standard deviation of gray-scale values)	Real value
A2	Perimeter	Real value
A3	Area	Real value
A4	Smoothness (local variation in radius lengths)	Real value
A5	Compactness (Perimeter^2/Area $-$ 1.0)	Real value
A6	Concavity (severity of concave portions of contour)	Real value
A7	Concave points (number of concave portions of contour)	Real value
A8	Symmetry	Real value
A9	Fractal dimension ("coastline approximation" - 1)	Real value

Table 4.4: Attributes of the Wisconsin Diagnostic Breast Cancer (WDBC) dataset

structure and parameters of the model. For structure learning, we applied a strcture learning software developed by Cassio P. de Campos, Zhi Zeng, and Qiang Ji [8], which can be downloaded from `http://www.ecse.rpi.edu/~cvrl/structlearning.html`. They used a branch and bound algorithm that uses structural constraints with the data.

As we discussed in Chapter 2, an efficient algorithm for parameter learning that can tackle the missing data problem is the EM algorithm [11]. We used the EM algorithm implemented in the Bayes Net Toolbox for Matlab written by Kevin Murphy (`https://code.google.com/p/bnt/`) to extract the parameters of the proposed BN from the breast cancer datasets.

There are 24,496 genes in Netherlands Breast Cancer dataset and 18,538 copy number variation profiles plus 49,576 expression profiles in METARBRIC dataset. Since the dimensionality of the genomic features for each instance in these two datasets is high compared to the number of samples of the dataset, structure and parameter learning would lead to poor results, in terms of accuracy of the prognosis, due to the curse of dimensionality dilemma and small sample size problem. Therefore, we first apply the dimensionality reduction algorithm described in Chapter 3, to reduce the number of genomic features in these two datasets.

More specifically, for the Netherlands Breast Cancer Dataset, we first apply Principal Component Analysis (PCA) to the data to reduce the dimensionality from $24,496$ to 130, by keeping 0.99 percent of the energy of the eigenvalues (sum of the absolute of the eigenvalues). Then, we used Isomap described in Chapter 3, to reduce the dimensionality from 130 to d which must be determined from the data by doing several experiments. We used $d = 30$. In this way, we have 59 features for each sample: 30 continuous genomic features plus 19 clinical traits.

For the METABRIC dataset, we first computed the correlation between class label (e.g. tumor grade or tumor stage) and $18,538$ copy number variation features plus $49,576$ expression features and kept the features which for them $\mid \rho \mid > 0.3$, where ρ is the correlation coefficient. Then, we applied dimensionality reduction algorithm to the remained features. The best performance was obtained by Isomap algorithm. In this way, we got 54 features: 30 continuous genomic features plus 24 clinical traits. For the other datasets, we did not used dimensionality reduction algorithms since the number of the traits is not high.

To create a BN, we represent each feature by a node. Then we apply the structure learning algorithm [8] to extract the structure of the BN. We suppose that each genomic node has no income link, while each clinical node may have income links from some genomic nodes. This is due to the fact that each clinical trait is related to some specific genes. We may easily set these constraints for the structure identification algorithm (see Figure 4.1).

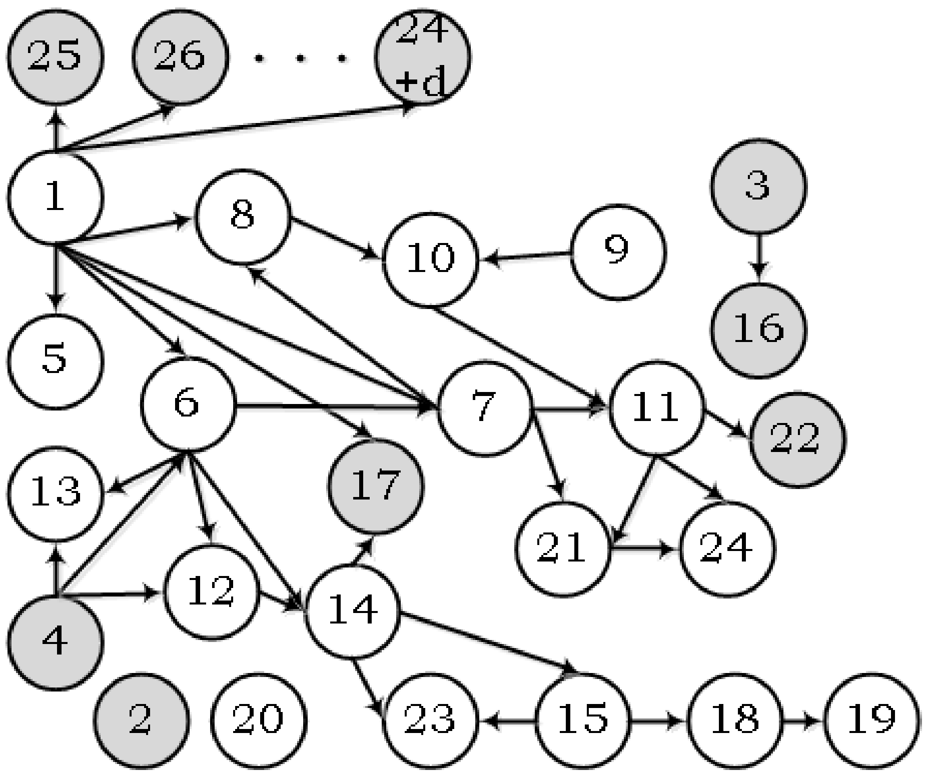


Figure 4.1: Structure Learning for the METABRIC Dataset. Nodes 1 to 24 are clinical variables, while nodes 25 to $24 + d$ are genomic variables. The number d is the dimensionality of the manifold. Node 1 denotes the tumor grade. The white nodes are discrete, while the gray nodes are continous.

Having the structure of the BN, we represent each genomic feature by a Gaussian node. Thus, we suppose that the microarray data has a Gaussian distributaion. For a clinical trait, we use a table CPD or a Gaussian node depends on the type of the trait and its parents (if there is any parents). Then, we apply the parameter learning algorithm described in Chapter 2 to extract the parameters of the BN.

4.3 Methodology

We use a techniques called *cross validation* to estimate the accuracy of our predictive BN. In an n-fold cross validation method, the samples of the data set are divided into n parts. Then, the samples of $n-1$ parts are used for training purpose and samples of the remaining part are used for testing. By repeating this process for each fold as the test fold, we obtain n values as the accuracy, and the average of them is used to estimate the accuracy of the predictive BN.

More specifically, we select first an interesting discrete clinical feature for which we anticipate its value. For examples, regional recurrence at any time during follow-up (0=no, 1=yes) may be used for this purpose. Then, we create a BN using samples of $n-1$ folds and the learning algorithm described in Chapters 2 and 3. This gives us two matrices M_{pca} and M_{iso} for dimensionality reduction using PCA and Isomap, plus structure and parameters of the BN.

For each sample of the remaining part, we first reduce its dimensionality using M_{pca} and M_{iso}, then we remove the value of the query trait, which is regional recurrence at any time during follow-up in this case, and compute its probability of being 0 or 1 using the BN that we created.

The maximum value is used for anticipating the label (0=no, 1=yes). For instance, if the probability of recurrence at any time during a follow-up is 0.3 (and the probability of no recurrence at any time during follow-up is 0.7), we can anticipate that no recurrence will occur for this sample at any time during follow-up. In the meantime, we can see whether the anticipated label is correct or not. In this way, we can compute the ratio of correctly classified samples for the test part.

We also implemented two baselines to compare the BN method with some typical classification methods. We used Support Vector Machines (SVMs) [14] and k-nearest neighbor (k-NN) [15] classifiers with cross validation technique to evaluate the BN method. The SVMs method applies an optimization technique to find a hyperplane with maximum margin from samples of two given linearly separable classes. After the model is created, new examples are mapped into the same space and anticipated to belong to a class based on which side of the hyperplane they fall on. There are nonlinear and multi-class extensions for this method (see Figure 4.2).

The k-NN classifier find k nearest samples in the training set by computing the Euclidean distance between a test sample and all training samples. Then, it assigns the class-label to the test sample based on the majority of these k training samples (see Figure 4.3).

We compare these methods with the proposed method by choosing various clinical traits as the query trait. For each query trait, we compare the accuracy of all methods. Also, we may change the number of genomic features (by changing the target dimensionality in dimensionality reduction algorithm) or other parameters of the network to obtain better performance.

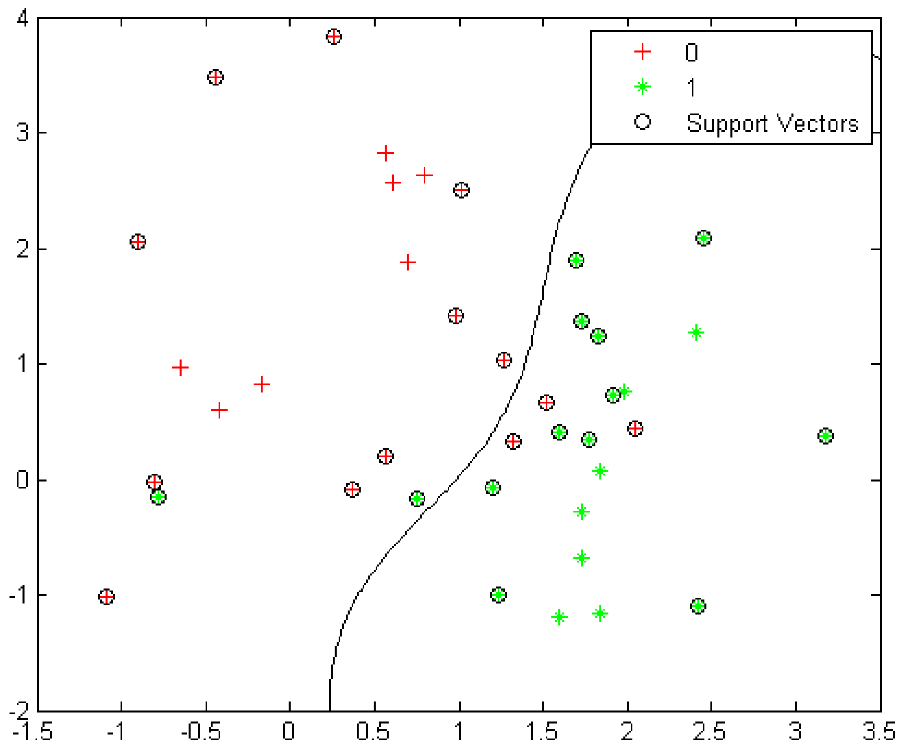


Figure 4.2: An SVMs classifier with two classes: the circled samples, which called support vectors, are the training samples that are effective for computing the hyperplane with maximum margin from samples of two classes. A new example can be classified based on which side of the hyperplane it fall on.

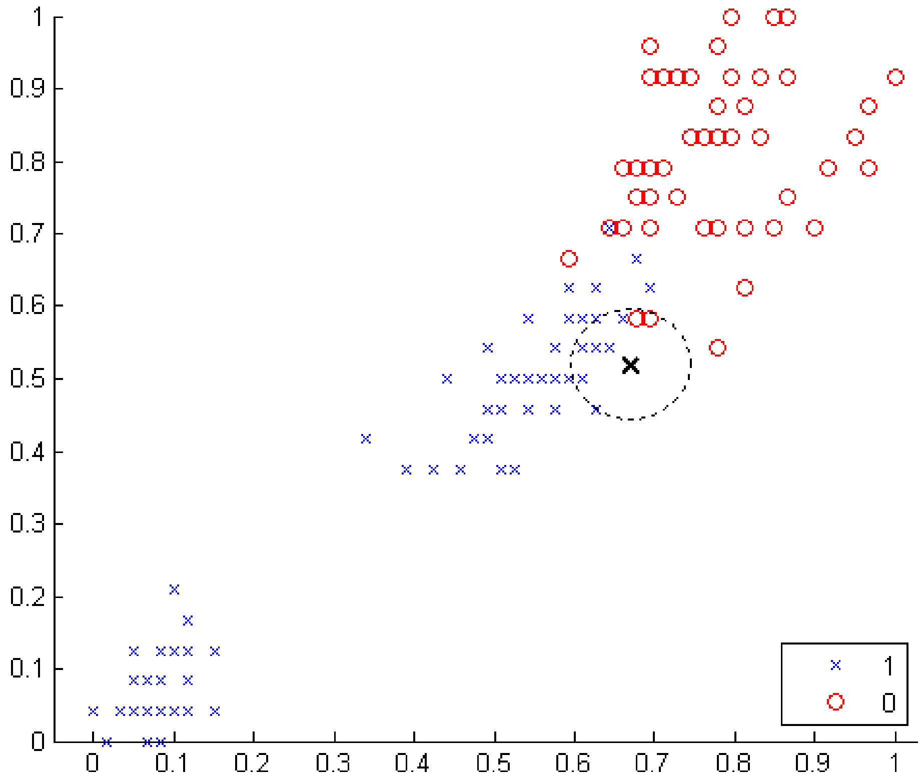


Figure 4.3: A 10-nearest neighbor (10-NN) classifier with two classes: a new sample (black cross) and its 10 nearest neighbors are represented inside a circle. The new sample is classified to the blue class since 8 samples of 10 nearest training samples belong to the blue class.

4.4 Results and Discussion

Table 4.5 represents the accuracy of the classification for different datasets. For NKI and METABRIC datasets, we implemented two baseline methods (SVMs and k-NN) to compare the proposed method (BN method) with them. For the other datasets, we compared the accuracy of the BN method with some best published results. For Ljubljana dataset, we compared the BN method with a method which uses *weighted networks* for representing the classification knowledge [17]. For WOBC dataset, we compared the BN method with a method which uses a *multisurface* technique for pattern seperation [19], and for WDBC dataset, we compared the proposed method with a method which applies a linear programming technique [12].

Dataset	Query trait	BN method	Baseline method
NKI	Recurrence	95%	93% (SVMs), 91% (k-NN)
	Metastasis	96%	94% (SVMs), 94% (k-NN)
	Death	91%	88% (SVMs), 87% (k-NN)
METABRIC	Tumor grade	83%	78% (SVMs), 78% (k-NN)
	Tumor stage	68%	62% (SVMs), 61% (k-NN)
Ljubljana	Recurrence	74%	74% (Weighted networks [17])
WOBC	Benign or not	97%	96% (Multisurface [19])
WDBC	Benign or not	96%	97% (Linear programming [12])

Table 4.5: Experimental results

As we can see from these results, the BN (with manifold learning for microarray data) method is promising especially for NKI and METABRIC datasets. This is due to the following advantages:

1. Modeling using BN is a powerful method for making decisions under uncertainty from data with noisy evidence.

2. The BN method can efficiently combine discrete and continous features.

3. The BN method can tackle the missing data problem by applying Expectation Maximization (EM) algorithm.
4. For the NKI and METABRIC datasets, using manifold learning techniques and by finding the optimum structure of the BN, we efficiently aggregated clinical and genomic (microarray) data.

Chapter 5

Conclusion and Future Research Directions

In this chapter, we briefly discuss conclusion and some future research directions. By applying structure and parameter learning algorithms, we trained a probabilistic graphical model for prognosis and diagnosis of breast cancer using real-world data. We applied manifold learning to reduce the dimensionality of the genomic features. Our experiments using different datasets shows promising results for classification of the tumors and predicting certain events, like recurrence and metastasis. The results obtained from our methodology can help medical doctors make better decisions about the best treatment for a patient. We may enter the results of the various tests to the model and predict the probability of events such as recurrence and metastasis.

Some future research directions are as follows:

1. Treatment selection: how promising is a certain treatment? For example, what is the survival interval, how likely metastasis is, and how likely recurrence is

for a new patient, if certain treatment (e.g. surgical removal, radiation therapy, hormone therapy, or chemotherapy) is applied.

2. Extracting new features directly from the MRI, CT and bone scans using manifold learning. This automatic feature extraction may improve the accuracy of the model, since these features can represent some information about the tumor that may not have been described by the clinical features, which are extracted by human. By this technique, manifold learning treats each pixel as a feature, and then it reduces the number of the features. Doctors cannot use these features for making decision since they are just numbers and do not represent semantic information for the doctors, but we may use them in the model.

3. Using dynamic probabilistic graphical model [11] for genomic features.

Appendix A

MATLAB Code

```
%_______________LOADING THE DATA__________________
clear all; close all; clc; format long;
M = 1981; % number of samples in METABRIC dataset
Data1 = load('Data1'); % microarray data
Data2 = load('Data2'); % clinical data
dnodes = [1 5 6 7 8 9 10 11 12 13 14 ...
         15 18 19 20 21 23 24]; % discrete nodes
%_______________MANIFOLD LEARNING__________________
%center data by subtracting the mean of the samples:
Data1 = Data1 - repmat(mean(Data1), M, 1);
options.Metric = 'Euclidean';
options.PCARatio = 0.99;
options.ReducedDim = 30;
% Isometric feature map (Isomap):
```

```
[eigvector, eigvalue] = IsoP(options, Data5);
% projection (reducing the dimensionality):
fea = [];
for i = 1 : M
    fea = [fea; Data1(i, :) * eigvector];
end
% scaling the features to [-1 1]:
ck = 1;
for k = 1 : size(fea, 2)
    max0(k) = max(fea(:, k));
    min0(k) = min(fea(:, k));
    if (min0(k) < max0(k))
        cmx(:, ck) = 2 * (fea(:, k) - min0(k)) /...
                      (max0(k) - min0(k)) - 1;
        ck = ck + 1;
    end
end
fea = cmx;
clear cmx;
d = size(fea, 2);
% d is the dimensionlity of the manifold (number of
% features after applying dimensionality reduction)
Data = [Data2 Data1]; %aggregate clinical and microarray
n = 24 + d;  % 24 is the number of clinical features, so
```

```
% n is total number of features (number of nodes in BN)
%__________________STRUCTURE LEARNING_________________
dag = zeros(n, n); % the adjacency matrix of the BN
%calling the structure learning software via DOS:
[a, b] = dos('sl.win64␣sl.txt␣out.txt');
% the output (adjacency matrix) will be in out.txt
fileID = fopen('out.txt');
dag = read_matrix(fileID);
%if there is an edge from node i to the node j, then
%dag(i, j) = 1, otherwise dag(i, j) = 0.
% dag(1:24, 1:24) = [
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0;
    %   0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0;
    %   0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0]';
%____________________CREATING THE BN________________
ns = ones(1, n);
% ns includes the number of states for each node
% ns(i) = 1 if i is a continous(Gaussian) node
ns(1) = 3; % node 1 is the tumor grade
ns(5) = 12;  ns(6) = 2;  ns(7) = 2;  ns(8) = 2;
ns(9) = 4; ns(10) = 3; ns(11) = 2;  ns(12) = 8;
ns(13) = 2; ns(14) = 4; ns(15) = 5; ns(18) = 3;
ns(19) = 2; ns(20) = 4; ns(21) = 6; ns(23) = 5;
ns(24) = 4;
onodes = ones(1, n); % all nodes are observation nodes
% make Bayesian network:
bnet = mk_bnet(dag, ns, 'discrete', dnodes, 'observed' ...
    , onodes);
% Conditional Probability Distributaion (CPDs):
%nodes 2, 3, 4, 16, 17, and 22 are continous (Gaussian
%CPD). Other clinical nodes are discrete (table CPD)
for j = 1 : 24
    if (j==2 || j==3 || j==4 || j==16 || j==17 || j==22)
        bnet.CPD{j} = gaussian_CPD(bnet, j);
    else
        bnet.CPD{j} = tabular_CPD(bnet, j);
    end
end
for i = 24 + 1 : n  % all genomic nodes are continous
    bnet.CPD{i} = gaussian_CPD(bnet, i);
end
%____EVALUATION OF THE BN (CLAASSIFICATION ACCURACY)___
max_iter = 20; sum0 = 0;
engine = jtree_inf_engine(bnet);
fold = 20; % number of folds for cross-validation
av = 0; % number of correctly classified samples
for test = 1 : fold
   st = (test - 1) * floor(M / fold) + 1;
   if (test == fold)
        en = M;
   else
```

```
      en = st + floor(M / fold) - 1;
  end
  D = [Data(1: st - 1, :); Data(en + 1 :M, :)];
  % parameter learning and the Expectation
  % Maximization (EM) algorithm
  [bnet0, ll] = learn_params_em(engine, D', max_iter);
   engine0 = jtree_inf_engine(bnet0);
   for i = st : en
     if (size(Data{i, 1}, 2) > 0)
       for j = 2 : n
           evidence{j} = Data{i, j};
       end
       % computing the probabilities for the new sample
       [engine1, loglik] = enter_evidence ...
                                    (engine0, evidence);
       mar = marginal_nodes(engine1, 1);
       % predicting the class
       [max1, index]= max([mar.T(1) mar.T(2) mar.T(3)]);
       % checking if the class label is correct or not
       if (Data(i, 1) == index)
          av = av + 1;
       end
       clear evidence;
       sum0 = sum0 + 1;
```

```
        **end**
    **end**
**end**
accuracy = av / sum0
```

Bibliography

[1] K. Bache and M. Lichman, *UCI machine learning repository*, 2013.

[2] A. Bhattacharjee, W. G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, et al., *Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses*, Proceedings of the National Academy of Sciences, 98 (2001), pp. 13790–13795.

[3] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, vol. 1, Springer New York, 2006.

[4] D. Cai, X. He, and J. Han, *Isometric projection*, 22 (2007), p. 528.

[5] L. Cayton, *Algorithms for manifold learning*, University of California, San Diego, Tech. Rep. CS2008-0923, (2005).

[6] G. F. Cooper and E. Herskovits, *A Bayesian method for the induction of probabilistic networks from data*, Machine learning, 9 (1992), pp. 309–347.

[7] C. Curtis, S. P. Shah, S.-F. Chin, G. Turashvili, O. M. Rueda, M. J. Dunning, D. Speed, A. G. Lynch, S. Samarajiwa, Y. Yuan, et al.,

The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups, Nature, 486 (2012), pp. 346–352.

[8] C. P. de Campos and Q. Ji, *Efficient structure learning of Bayesian networks using constraints.*, Journal of Machine Learning Research, 12 (2011), pp. 663–689.

[9] O. Gevaert, F. De Smet, D. Timmerman, Y. Moreau, and B. De Moor, *Predicting the prognosis of breast cancer by integrating clinical and microarray data with Bayesian networks*, Bioinformatics, 22 (2006), pp. e184–e190.

[10] I. Jolliffe, *Principal component analysis*, Wiley Online Library, 2005.

[11] D. Kollar and N. Friedman, *Probabilistic graphical models: principles and techniques*, The MIT Press, 2009.

[12] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, *Breast cancer diagnosis and prognosis via linear programming*, Operations Research, 43 (1995), pp. 570–577.

[13] X. Niyogi, *Locality preserving projections*, 16 (2004), p. 153.

[14] B. Schölkopf, C. J. Burges, and A. J. Smola, *Advances in kernel methods: support vector learning*, The MIT press, 1999.

[15] G. Shakhnarovich, T. Darrell, and P. Indyk, *Nearest-neighbor methods in learning and vision*, vol. 19, 2008.

[16] D. Singh, P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A. V. D'Amico, J. P. Richie, et al., *Gene expression correlates of clinical prostate cancer behavior*, Cancer cell, 1 (2002), pp. 203–209.

[17] M. Tan and L. J. Eshelman, *Using weighted networks to represent classification knowledge in noisy domains.*, (1988), pp. 121–134.

[18] M. J. Van De Vijver, Y. D. He, L. J. van't Veer, H. Dai, A. A. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, et al., *A gene-expression signature as a predictor of survival in breast cancer*, New England Journal of Medicine, 347 (2002), pp. 1999–2009.

[19] W. H. Wolberg and O. L. Mangasarian, *Multisurface method of pattern separation for medical diagnosis applied to breast cytology.*, Proceedings of the national academy of sciences, 87 (1990), pp. 9193–9196.