Selected topics in Multivariate Statistical Analysis

A thesis Submitted
in Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

by
Tanikella Padma Ragaleena

to the
School of Mathematical Sciences
National Institute of Science Education and Research
Bhubaneswar
13 July 2021

DECLARATION

I hereby declare that I am the sole author of this thesis in partial fulfillment of the requirements for a postgraduate degree from National Institute of Science Education and Research (NISER). I authorize NISER to lend this thesis to other institutions or individuals for the purpose of scholarly research.

T. Padma Ragaleena

Signature of the Student
Date: 13-07-21

The thesis work reported in the thesis entitled.. was carried out under my supervision, in the school of .... at NISER, Bhubaneswar, India.

Signature of the thesis supervisor
School:
Date:

Abstract

As the title suggests, the thesis covers selected topics from multivariate statistical analysis. The primary reference used while learning different topics was “Modern Multivariate Statistical Techniques” by Alan J. Izenman.

The first two chapters focus on introductory topics. Multivariate gaussian distribution and the properties associated with the distribution have been mentioned in detail. The third chapter deals with “Principal component analysis” which is a dimension reduction technique where we attempt to a new set of variables which are not only uncorrelated with each other, but also much fewer in number than the original set of variables that we start with. Construction of these new variables is such that these new variables capture most of the variation information present in the data.

Lastly, the three remaining chapters deal with different kinds of classification techniques. Statistical classification techniques attempt to predict the true class label of a new observation (i.e. an observation not used in the construction of the classifier) based on a set of observations whose class labels are known beforehand. The thesis discusses the following classification methods - linear discriminant analysis, quadratic discriminant analysis, Fisher discriminant analysis, decision tree classifier, and artificial neural network classifier. Underlying assumptions of each of these techniques along with how each of these methods work has been discussed.

Acknowledgment

I would first like to thank Dr. Nabin Kumar Jana, my thesis supervisor, and Dr. Shyamal Krishna De without whose help and co-operation, a thesis in statistics would have been impossible. I will always be grateful for their help. They consistently allowed this project to be my own work, but steered me in the right the direction whenever they thought I needed it.

Finally, I must express my very profound gratitude to my parents and to my friends for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of writing this thesis. This accomplishment would not have been possible without them. Thank you.

Tanikella Padma Ragaleena
School of Mathematical Sciences
NISER, Bhubaneswar

Contents

Chapter 1

Multivariate Gaussian distribution

The content of this chapter has been taken from [Ize08], [And03], [SL03], [RTSH08, Appendix A.12. A.13], [Ana07], [Lee14], [Mas11] and [Bar06]. The figures in the chapter have been taken from [Lee14]

The normal distribution, commonly known as the bell curve, plays a crucial role in the field of statistics. Therefore, it is worth noting that the univariate normal density was first discovered by De Moivre in the year 1733 while attempting to approximate the binomial sum. Interestingly, this is nothing but the "normal approximation to binomial" that most of us learn during our introductory course in statistics. Later, in the year 1783, Laplace used normal density curve to describe the distribution of errors which was subsequently used by Gauss to analyze astronomical data in 1809.

The fundamental properties of the multivariate normal distribution are discussed in the current chapter.

1.1 Multivariate normal density

Definition 1.1. The $p \times 1$ random vector $\tilde{X}$ is said follow a (non-singular) multivariate normal distribution if its density function is as follows:

$$f_{\tilde{X}}(\tilde{x}) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left[\frac{-1}{2}(\tilde{x} - \tilde{\mu})^T |\Sigma|^{-1}(\tilde{x} - \tilde{\mu})\right] \tag{1.1}$$

where $\tilde{\mu}$ is a $p \times 1$ vector while Σ is a $p \times p$ symmetric positive definite matrix.

Observe that the univariate normal distribution can be written as

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

It is clear that we obtain the univariate normal distribution when we replace the $p \times p$ symmetric positive definite matrix Σ^{-1} by the 1×1 symmetric positive definite matrix $\frac{1}{\sigma^2}$.

Remark 1.1.

1. Observe that Σ^{-1} is also a symmetric positive definite matrix as the inverse of a symmetric positive definite matrix is also symmetric and positive definite.

2. Observe that Σ has to be an invertible matrix for the multivariate normal distribution to be well defined. However, if Σ is non-invertible, one can define a **singular normal distribution**. The density in this case will involve the generalized inverse of Σ in place of the inverse mentioned in (1.1).

3. The assumption that Σ is positive definite ensures that the multivariate density is a bounded function on $\mathbb{R}^p$. This ensures that the integral of the normal density is finite.

In the reminder of the section, I will verify the fact that the normal density is indeed a mathematically well-defined density function.

The exponential part of the multivariate normal density function in (1.1) ensures that the density is positive for all $\tilde{x} \in \mathbb{R}^b$. Now, we show that the integral of the density function over $\mathbb{R}^p$ integrates to one.

$$\text{Let } K^* = \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \exp\left\{ \frac{-1}{2} (\tilde{x} - \tilde{\mu})^T \mathbf{A} (\tilde{x} - \tilde{\mu}) \right\} \mathrm{d}x_p \mathrm{d}x_{p-1} \cdots \mathrm{d}x_1 \text{ where } \mathbf{A} = \Sigma^{-1} \tag{1.2}$$

Using the spectral decomposition theorem for the symmetric matrix $\mathbf{A}$, we can claim that there exists an orthogonal matrix $\mathbf{P}$ such that $\mathbf{A} = \mathbf{PDP^T}$ where $\mathbf{D}$ is the diagonal matrix containing the eigen values of $\mathbf{A}$. Now, write $\mathbf{D}$ as $\mathbf{D} = \mathbf{D}^{\frac{1}{2}}\mathbf{D}^{\frac{1}{2}}$. Taking $\mathbf{C} = \mathbf{PD}^{\frac{-1}{2}}$, we can show that, for every $\mathbf{A}$, there exists an invertible matrix $\mathbf{C}$ such that $\mathbf{C^T A C} = \mathbf{I}$.

Since $\mathbf{C}$ is invertible, the following transformation on $\tilde{x}$ can be defined.

$$\tilde{y} = \mathbf{C}^{-1}(\tilde{x} - \tilde{\mu}) \tag{1.3}$$

Using the transformation (1.3) in (1.2) and then applying the change of variable formula gives the following integral:

$$K^* = |\det \mathbf{C}| \times \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \exp\left\{ \frac{-\tilde{y}^T \tilde{y}}{2} \right\} \mathrm{d}x_p \mathrm{d}x_{p-1} \cdots \mathrm{d}x_1 \tag{1.4}$$

where $|\det \mathbf{C}|$ is the modulus of the Jacobian of the transformation (1.3). Since $\tilde{y}^T\tilde{y} = \sum_{i=1}^{p} y_i^2$ and the univariate normal density integrates to 1, the equation (1.4) can be written as

$$K^* = |\det \mathbf{C}| \times \prod_{i=1}^{p} \left\{ \int_{-\infty}^{+\infty} \exp\left\{ \frac{-y_i^2}{2} \right\} \right\} = |\det \mathbf{C}| \times \prod_{i=1}^{p} (\sqrt{2\pi}) = |\det \mathbf{C}| (\sqrt{2\pi})^p$$

Finaly, using $\mathbf{C^T A C} = \mathbf{I}$ we can show that $|\det\mathbf{C}| = 1/\sqrt{\det\mathbf{A}}$.

$$K^* = (\sqrt{\det\mathbf{A}})^{-1} \times (\sqrt{2\pi})^p = (\sqrt{\det\Sigma}) \times (\sqrt{2\pi})^p \tag{1.5}$$

Therefore, (1.5) proves that the density in (1.1) integrates to one. This proves that the multivariate normal density is a well defined density function.

1.2 Expectation and Variance

Definition 1.2. An $m \times n$ matrix $\mathbf{M} = (M_{ij})$ with random variables M_{ij} as its entries is called a random matrix.

Definition 1.3. If $\mathbf{M} = (M_{ij})$ is an $m \times n$ random matrix, then the expected value of the matrix is defined as

$$\mathbb{E}[\mathbf{M}] = (\mathbb{E}[M_{ij}]) \text{ for all } (i,j) \in \{1, 2, \cdots m\} \times \{1, 2, \cdots, n\}$$

Since a vector is a special kind of a matrix, this definition can be used to define the expectation of a random vector $\tilde{X}$ as well. $\mathbb{E}[\tilde{X}]$ is usually denoted by $\tilde{\mu}$ in this report.

Definition 1.4. The variance or covariance matrix of a $p \times 1$ random vector is defined as the expectation of the $p \times p$ matrix $(\tilde{X} - \tilde{\mu})(\tilde{X} - \tilde{\mu})^T$ where $\tilde{\mu} = \mathbb{E}[\tilde{X}]$.

$$\mathcal{V}[\tilde{X}] = \mathbb{E}[(\tilde{X} - \tilde{\mu})(\tilde{X} - \tilde{\mu})^T] = \mathbb{E}[\tilde{X}\tilde{X}^T] - \tilde{\mu}\tilde{\mu}^T$$

Lemma 1.1. *If* $\mathbf{Z}$ *is an* $m \times n$ *random matrix,* $\mathbf{D}$ *is an* $l \times m$ *real matrix,* $\mathbb{E}$ *is an* $n \times q$ *real matrix, and* $\mathbf{F}$ *is an* $l \times q$ *real matrix, then*

$$\mathbb{E}[\mathbf{DZE} + \mathbf{F}] = \mathbf{D}(\mathbb{E}[\mathbf{Z}])\mathbf{E} + \mathbf{F} \tag{1.6}$$

Proof. Let $\mathbf{Z} = (z_{ij})$, $\mathbf{D} = (d_{ij})$, $\mathbf{E} = (e_{ij})$, and $\mathbf{F} = (f_{ij})$ where $\mathbf{M} = (m_{ij})$ for a $p \times q$ matrix $\mathbf{M}$ implies that that the matrix $\mathbf{M}$ consists of the elements m_{ij} for all $(i.j) \in \{1, 2, \cdots, p\} \times \{1, 2, \cdots, q\}$. Using this notation, we can show that that general (i,j)th element on both the sides of (1.6) is equal. This proves that the lemma is true. □

Lemma 1.2. *Let* $\tilde{Y}$ *be an* $p \times 1$ *random vector, such that it follows the following equation for a random vector* $\tilde{X}$ *of size* $k \times 1$*:*

$$\tilde{Y} = \mathbf{D}\tilde{X} + \tilde{F} \tag{1.7}$$

Note that the matrices $\mathbf{D}$ *and* $\mathbf{F}$ *are non-random matrices of appropriate sizes. Then*

$$\mathcal{V}[\tilde{Y}] = \mathbf{D}\mathcal{V}[\tilde{X}]\mathbf{D^T} \tag{1.8}$$

Proof.

$$\mathcal{V}[\tilde{Y}] = \mathbb{E}[\mathbf{D}(\tilde{X} - \tilde{\mu})(\tilde{X} - \tilde{\mu})^T \mathbf{D}^T] = \mathbf{D}\mathbb{E}[(\tilde{X} - \tilde{\mu})(\tilde{X} - \tilde{\mu})^T]\mathbf{D}^T$$

The first equality is obtained by using the definition of the variance of a random vector. The second equality follows from the Lemma 1.1. □

Let $\tilde{X}$ be a multivariate normal random vector. As mentioned earlier, its density will be given as follows:

$$f_{\tilde{X}}(\tilde{x}) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left[\frac{-1}{2}(\tilde{x} - \tilde{\mu})^T |\Sigma|^{-1}(\tilde{x} - \tilde{\mu})\right] \tag{1.9}$$

Now consider the transformation (1.3) that we used earlier. Then density $g_{\tilde{Y}}$ of $(Y_1, Y_2, \cdots, Y_p)$ is given by

$$g_{\tilde{Y}}(y_1, y_2, \cdots, y_p) = f_{\tilde{X}}(x_1(\tilde{y}), \cdots, x_p(\tilde{y})) \times |\mathcal{J}| \tag{1.10}$$

where $|\mathcal{J}|$ is the modulus of the Jacobian of the transformation (1.3). Using (1.10), we can show that the density of $(Y_1, Y_2, \cdots, Y_p)$ is

$$g_{\tilde{Y}}(\tilde{y}) = \frac{1}{(2\pi)^{\frac{p}{2}}} \exp\left\{\frac{-\tilde{y}^T\tilde{y}}{2}\right\} = \frac{1}{(2\pi)^{\frac{p}{2}}} \prod_{i=1}^{p} \exp\left\{\frac{-y_i^2}{2}\right\}$$

It is easy to see from the calculations below that the expected value of Y_i, for any $i \in \{1, 2, \cdots, p\}$, is 0 since the integral of $y_i \exp\left\{\frac{-y_i^2}{2}\right\}$, an odd function, over $\mathbb{R}$ is 0.

$$\mathbb{E}[Y_i] = \frac{1}{(2\pi)^{\frac{p}{2}}} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} y_i \exp\left\{\prod_{i=1}^{p} \exp\left\{\frac{-y_i^2}{2}\right\}\right\} \mathrm{d}x_p \mathrm{d}x_{p-1} \cdots \mathrm{d}x_1 = 0 \tag{1.11}$$

Equation (1.11), implies that $\mathbb{E}[\tilde{Y}] = \tilde{0}$. Now, applying Lemma 1.7 to the transformation (1.3), we can show that $\mathbb{E}[\tilde{X}] = \tilde{\mu}$. Similarly, we can compute the variance of the vector $\tilde{X}$ by first finding the variance of $\tilde{Y}$ and then using Lemma 1.7.

From the definition of covariance matrix,

$$\mathcal{V}[\tilde{Y}] = \mathbb{E}[\tilde{Y}\tilde{Y}^T] = (\mathbb{E}[Y_iY_j])\ (i, j) \in \{1, 2, \cdots, p\} \times \{1, 2, \cdots, p\}$$

So finding the variance of $\tilde{X}$ is equivalent to calculating the expectation of $\mathbb{E}[Y_iY_j]$ for all $(i, j) \in \{1, 2, \cdots, p\} \times \{1, 2, \cdots, p\}$.

$$\mathbb{E}[Y_iY_j] = \frac{1}{(2\pi)^{\frac{p}{2}}} \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} y_iy_j \exp\left\{\prod_{i=1}^{p} \exp\left\{\frac{-y_i^2}{2}\right\}\right\} \mathrm{d}x_p \mathrm{d}x_{p-1} \cdots \mathrm{d}x_1 = 0 \tag{1.12}$$

$$= \begin{cases} \int_{-\infty}^{+\infty} y_i \exp\{\frac{-y_i^2}{2}\}\mathrm{d}y_i \times \int_{-\infty}^{+\infty} y_j \exp\{\frac{-y_j^2}{2}\}\mathrm{d}y_j \times (2\pi)^{\frac{-p}{2}} \prod_{k\neq i,j} \int_{-\infty}^{+\infty} \exp\{\frac{-y_k^2}{2}\} = 0 & i \neq j \\ \int_{-\infty}^{+\infty} y_i^2 \exp\{\frac{-y_i^2}{2}\} \times \prod_{k\neq i} \int_{-\infty}^{+\infty} \exp\{\frac{-y_k^2}{2}\} = 1 & i = j \end{cases} \tag{1.13}$$

It is clear from (1.13), that the covariance matrix of $\tilde{Y}$ is the $p \times p$ identity matrix denoted by $\mathcal{I}_p$. Now, using Lemma 1.7, we can deduce that the covariance matrix of $\tilde{X} = \mathbf{C}\tilde{Y} + \mu$ is $\mathbf{C}\mathbf{C}^T = \mathbf{A}^{-1} = \Sigma$.

Therefore,we proved that if $\tilde{X}$ follows a multivariate normal distribution with a density function as shown in (1.1), then the expected value of the vector is $\tilde{\mu}$ and the covariance matrix is Σ. When $\tilde{X}$ is a multivariate normal random vector with mean $\tilde{\mu}$ and variance σ, we denote it as $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$. The p in the notation is sometimes ignored in the report if the dimension of the random vector $\tilde{X}$ is clear. Additionaly, the density of a multivariate normal distribution is denoted as $n(\tilde{x} \mid \tilde{\mu}, \Sigma)$.

Finally, the section can be concluded with the following theorem that summarizes all the results discussed so far.

Theorem 1.1. *If $\tilde{X}$ is a p-dimensional vector corresponding to the density function $f_{\tilde{X}}$ given below, then the expected value and variance of the vector are $\tilde{b}$ and $\mathbf{A}^{-1}$ respectively.*

$$f_{\tilde{X}}(\tilde{x}) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left[\frac{-1}{2}(\tilde{x} - \tilde{b})^T \mathbf{A}(\tilde{x} - \tilde{\tilde{b}})\right]$$

Conversely, given a vector $\tilde{b} \in \mathbb{R}^p$ and a symmetric positive definite matrix Σ, then there exists a multivariate density function

$$\frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left[\frac{-1}{2}(\tilde{x} - \tilde{b})^T \mathbf{A}(\tilde{x} - \tilde{\tilde{b}})\right]$$

such that the vector corresponding to this density has mean $\tilde{\mu}$ and variance Σ.

1.3 Marginal distribution and independence of normal random vectors

In this section, we prove that the marginal density of any $q < p$ components of the p-dimensional Gaussian random vector $\tilde{X}$ is also a multivariate normal vector. Let the components of $\tilde{X}$ be re-ordered in such a way that the q components of our interest are written as the first q components of the vector $\tilde{X}$. Let this sub-vector be called $\tilde{X}^{(1)}$. Denote the remaining $(p - q) \times 1$ subvector by $\tilde{X}^{(2)}$. Before proving the statement claimed, it is important to go through some notations that I will use quite often in the remainder of the chapter.

$$\tilde{X}^{(1)} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_q \end{bmatrix} \text{ and } \tilde{X}^{(2)} = \begin{bmatrix} x_{q+1} \\ x_{q+2} \\ \vdots \\ x_p \end{bmatrix}$$

Denote the expected values of $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ by $\tilde{\mu}^{(1)}$ and $\tilde{\mu}^{(2)}$ respectively. In addition to that, the following notations will also be used often.

$$\begin{aligned}
\Sigma_{11} &= \mathbb{E}[(\tilde{X}^{(1)} - \tilde{\mu}^{(1)})(\tilde{X}^{(1)} - \tilde{\mu}^{(1)})^T] \\
\Sigma_{22} &= \mathbb{E}[(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})^T] \\
\Sigma_{12} &= \mathbb{E}[(\tilde{X}^{(1)} - \tilde{\mu}^{(1)})(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})^T]
\end{aligned}$$

It is easy to observe that

$$\tilde{\mu} = \begin{bmatrix} \tilde{\mu}^{(1)} \\ \tilde{\mu}^{(2)} \end{bmatrix} \text{ and } \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$$

where $\Sigma_{21} = \Sigma_{12}^T$

Lemma 1.3. *Let $\tilde{X}$ and Σ be partitioned as mentioned above. If $\Sigma_{12} = \Sigma_{21}^T = \mathbf{0}_{q\times(p-q)}$. Then $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ are independent and normally distributed random vectors.*

Proof. Let the quadratic form in $n(\tilde{x} \mid \tilde{\mu}, \Sigma)$ be denoted by $\mathcal{Q}$ i.e. $\mathcal{Q} = (\tilde{x} - \tilde{\mu})^T \Sigma (\tilde{x} - \tilde{\mu})$. If $\Sigma_{12} = \Sigma_{21}^T = \mathbf{0}_{q\times(p-q)}$, as given in the statement of the lemma, then it is easy to show that:

$$\mathcal{Q} = (\tilde{x}^{(1)} - \tilde{\mu}^{(1)})^T \Sigma_{11} (\tilde{x}^{(1)} - \tilde{\mu}^{(1)}) + (\tilde{x}^{(2)} - \tilde{\mu}^{(2)})^T \Sigma_{22} (\tilde{x}^{(2)} - \tilde{\mu}^{(1)}) := \mathcal{Q}_1 + \mathcal{Q}_2 \quad (1.14)$$

Σ is now a block matrix with $\mathbf{0}$ as off-diagonal matrices.

$$\text{Therefore, } \det(\Sigma) = \det(\Sigma_{11}) \times \det(\Sigma_{22}) \quad (1.15)$$

Using (1.14) and (1.15) in the multivariate normal density function $n(\tilde{x} \mid \tilde{\mu}, \Sigma)$, one can show that

$$n(\tilde{x} \mid \tilde{\mu}, \Sigma) = n(\tilde{x}^{(1)} \mid \tilde{\mu}^{(1)}, \Sigma_{11}) \times n(\tilde{x}^{(2)} \mid \tilde{\mu}^{(2)}, \Sigma_{22}) \quad (1.16)$$

It is clear from (1.16), that the random vectors $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ are independent random vectors. Now, we are left to prove that the random vectors $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ are multivariate normal random variables. This can be proved by showing that both the subvector have a marginal normal density.

$$f_{\tilde{X}^{(1)}} = \int_{\forall x_{q+1}} \cdots \int_{\forall x_p} n(\tilde{x}^{(1)} \mid \tilde{\mu}^{(1)}, \Sigma_{11}) \times n(\tilde{x}^{(2)} \mid \tilde{\mu}^{(2)}, \Sigma_{22}) \mathrm{d}x_p \cdots \mathrm{d}x_{q+1} = n(\tilde{x}^{(1)} \mid \tilde{\mu}^{(1)}, \Sigma_{11}) \tag{1.17}$$

Similarly, it can be shown that the marginal density of $\tilde{X}^{(2)}$ is $n(\tilde{x}^{(2)} \mid \tilde{\mu}^{(2)}, \Sigma_{22})$. This proves the lemma that $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ are independent and normally distributed random vectors. □

This proves that the marginal density of any subset of components of the multivariate normal random vector $\tilde{X}$ is again normally distributed **provided** $\Sigma_{12} = \Sigma_{21}^T = \mathbf{0}$.

Remark 1.2. Note that $\Sigma_{12} = \Sigma_{21}^T = \mathbf{0}$ is true when it is given beforehand that $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ are any two independently distributed random vectors. However, the converse of this is not always true. The converse has been proved to be true when $\tilde{X}$ follows a multivariate normal distribution.

The results proved so far have been summarized in the theorem below:

Theorem 1.2.

1. *Let $\tilde{X} = ([\tilde{X}^{(1)}]^T, [\tilde{X}^{(2)}]^T)$ follow a multivariate normal distribution i.e. $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$. Assume that $cov(X_i^{(1)}, X_j^{(2)}) = 0$ where $[\tilde{X}^{(1)}]^T = (X_1^{(1)}, \cdots, X_q^{(1)})$ and $[\tilde{X}^{(2)}]^T = (X_{q+1}^{(2)}, \cdots, X_p^{(1)})$. Then, according to the notation mentioned earlier,*
   - *$\tilde{X}^{(1)} \sim \mathcal{N}_p(\tilde{\mu}^{(1)}, \Sigma_{11})$ and $\tilde{X}^{(2)} \sim \mathcal{N}_p(\tilde{\mu}^{(2)}, \Sigma_{22})$.*
   - *$\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ are independent random vectors i.e. $\tilde{X}^{(1)} \perp \tilde{X}^{(2)}$.*
2. *Let $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$, then a necessary and sufficient condition for one subset of random variables, say $\tilde{X}^{(1)}$, and the subset consisting of the remaining variables $\tilde{X}^{(2)}$ to be independent is $cov(X_i^{(1)}, X_j^{(2)}) = 0$ for all $X_i^{(1)} \in \{X_1^{(1)} \cdots, X_q^{(1)}\}$ and $X_i^{(2)} \in \{X_{q+1}^{(2)} \cdots, X_p^{(1)}\}$*

Remark 1.3. However, it is possible for two random vectors $\tilde{X} = (X_1, \cdots, X_m)^T$ and $\tilde{Y}(Y_1, \cdots, Y_n)^T$ to have the following set of properties:

- $(\tilde{X}^T, \tilde{Y}^T)$ are not jointly Gaussian.
- $cov(X_i, Y_j) = 0$ for all $X_i \in \{X_1 \cdots, X_m\}$ and $Y_j \in \{Y_1 \cdots, Y_n\}$
- $\tilde{X}$ and $\tilde{Y}$ are individually Gaussian random vectors
- $\tilde{X}$ and $\tilde{Y}$ are **not** independent.

We have proved that the marginal distribution of multivariate normal distribution is again normally distributed in a very particular case. We now aim to prove the statement without any specific assumptions on the matrix Σ. Before doing that, a couple of results have been recalled or proved below.

The matrices mentioned are block matrices

$$\det\begin{bmatrix}\mathbf{A}_{r\times r} & \mathbf{B}_{r\times s}\\ \mathbf{0}_{s\times r} & \mathbf{C}_{s\times s}\end{bmatrix} = \det\begin{bmatrix}\mathbf{A}_{r\times r} & \mathbf{0}_{r\times s}\\ \mathbf{B}_{s\times r} & \mathbf{C}_{s\times s}\end{bmatrix} = \det\mathbf{A}\times\det\mathbf{C} \tag{1.18}$$

Theorem 1.3. *Let $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma_x)$. Then $\tilde{Y} = \mathbf{C}\tilde{X}$ for some non-singular $\mathbf{C}$ is also normally distributed. In fact, $\tilde{Y} \sim \mathcal{N}_p(\mathbf{C}\tilde{\mu}, \mathbf{C}\Sigma_x\mathbf{C}^T)$*

Proof. Since $\mathbf{C}$ is an invertible matrix, the following transformation is well-defined:

$$\tilde{Y} = \mathbf{C}^{-1}\tilde{X} \tag{1.19}$$

The Jacobian of the transformation 1.19 is $\det\mathbf{C}^{-1}$. This imples the following equations:

$$|\mathcal{J}| = |\det\mathbf{C}^{-1}| = \frac{1}{|\det\mathbf{C}|} = \sqrt{\frac{1}{|\det\mathbf{C}|^2}} = \sqrt{\frac{1}{|\det\mathbf{C}|\times|\det\mathbf{C}^T|}} = \sqrt{\frac{|\det\Sigma_x|}{|\det\mathbf{C}||\det\Sigma_x||\det\mathbf{C}^T|}} \tag{1.20}$$

Now, the density of $\tilde{Y}$ can be calculated as follows:

$$f_{\tilde{Y}}(y_1,\cdots,y_p) = |\mathcal{J}|\times f_{\tilde{X}}(x_1(\tilde{y})\cdots,x_p(\tilde{y})) \tag{1.21}$$

Using 1.19, 1.20 and 1.21 it can be proved that $\tilde{Y} \sim \mathcal{N}_p(\mathbf{C}\tilde{\mu}, \mathbf{C}\Sigma_x\mathbf{C}^T)$. □

Theorem 1.4. *Let $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$, then the marginal distribution of any subset of components of $\tilde{X}$ is multivariate normal with means and variances obtained by taking corresponding components of $\tilde{\mu}$ and $\tilde{\Sigma}$ respectively.*

Proof. Without loss of generality, we can assume that the components of $\tilde{X}$ whose marginal density we wish to calculate are the last $p-q$ components of $\tilde{X}$. Denote this sub-vector by $\tilde{X}^{(2)}$ while the first q components form the subvector $\tilde{X}^{(1)}$. Then it is easy to show that there exists an invertible matrix $\mathbf{B}$ such that $cov(X^{(1)} + \mathbf{B}X^{(2)}, X^{(2)}) = 0$. In particular, $\mathbf{B} = -\Sigma_{12}\Sigma_{22}^{-1}$. Now, let $Y^{(1)} = X^{(1)} + \mathbf{B}X^{(2)} = X^{(1)} - \Sigma_{12}\Sigma_{22}$ and $Y^{(2)} = X^{(1)}$. This can be written in the block-matrix form as follows:

$$\begin{bmatrix}Y^{(1)}_{q\times 1}\\ Y^{(2)}_{(p-q)\times 1}\end{bmatrix} = \begin{bmatrix}\mathcal{I}_q & -\Sigma_{12}\Sigma_{22}^{-1}\\ \mathbf{0}_{(p-q)\times q} & \mathcal{I}_{p-q}\end{bmatrix}\begin{bmatrix}X^{(1)}_{q\times 1}\\ X^{(2)}_{(p-q)\times 1}\end{bmatrix} \text{ or } \tilde{Y} = \mathbf{A}\tilde{X} \text{ in short.}$$

Since $\tilde{Y} = \mathbf{A}\tilde{X}$ and $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$, we can use Theorem 1.3 to claim that $\tilde{Y} \sim \mathcal{N}_p(\mathbf{A}\tilde{\mu}, \mathbf{A}\Sigma\mathbf{A}^T)$. Simple block matrix calculations prove that

$$\mathbb{E}[\tilde{Y}] = \begin{bmatrix} \tilde{\mu}^{(1)} - \Sigma_{12}\Sigma_{22}^{-1}\tilde{\mu}^{(2)} \\ \tilde{\mu}^{(2)} \end{bmatrix} \text{ and } \mathcal{V}[\tilde{Y}] = \begin{bmatrix} \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} & \mathbf{0} \\ \mathbf{0} & \Sigma_{22} \end{bmatrix}$$

Now, using Theorem 1.2, it is clear that $\tilde{X}^{(2)} \sim \mathcal{N}(\tilde{\mu^{(2)}}, \Sigma_{22})$ Since the components of interest in $\tilde{X}$ can be written as the subvector $\tilde{X}^{(2)}$, we have shown that if $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$, then any subset of the elements of $\tilde{X}$ also follow the Gaussian distribution. □

Corollary 1.1. *Let $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$, then any component X_i of $\tilde{X}$ follows the univariate normal distribution.*

Proof. This follows directly from the Theorem 1.4. □

Remark 1.4. We just proved that if $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$, then each component X_i of $\tilde{X}$ is also normally distributed. However, the converse of this is not always true. It is possible for random variables $X_1, X_2 \cdots, X_n$ to exist such that each one of them is a Gaussian random variable while their joint density is not Gaussian.

1.4 Conditional distributions

In this section, we prove the fact that the conditional distributions derived from joint normal distributions are also normal. Let us assume, once again, that the vector $\tilde{X}$ is divided into two subvectors $\tilde{X}^{(1)}$ and $\tilde{X}^{(1)}$ of sizes $q \times 1$ and $(p-q) \times 1$ respectively. An important and useful theorem is first proved before proving the main theorem of interest.

Theorem 1.5. *If $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$ and let $\tilde{Z} = \mathbf{D}\tilde{X}$ for some $q \times p$ matrix $\mathbf{D}$ of rank $q \leqslant p$. Then,*

$$\tilde{Z} \sim \mathcal{N}_p(\mathbf{D}\tilde{\mu}, \mathbf{D}\Sigma\mathbf{D}^T)$$

Proof. First of all, if rank$(\mathbf{D}) = q = p$, then $\mathbf{D}$ is a full rank matrix. Then the theorem mentioned will be true according to the Theorem 1.3 proved earlier.

Now, assume that rank$(\mathbf{D}) = q < p$. Then, $\mathbf{D}$ is a singular matrix with q linearly independent row vectors $\{\tilde{d_1}, \tilde{d_2} \cdots, \tilde{d_q}\}$. These q vectors span a subspace in $\mathbb{R}^p$. Since every vector space has a basis, there exist vectors $\{\tilde{e}_1, \cdots, \tilde{e}_{p-q}\}$ in $\mathbb{R}^p$ such that the set $\{\tilde{d_1}, \tilde{d_2} \cdots, \tilde{d_q}, \tilde{e}_1, \cdots, \tilde{e}_{p-q}\}$ form a basis for the real vector space $\mathbb{R}^p$. Let $\mathbf{E}$ be the $(p-q) \times p$ matrix with the vectors $\tilde{e}_1, \cdots, \tilde{e}_{p-q}\}$ as rows.

$$\mathbf{C} = \begin{bmatrix} \mathbf{D} \\ \mathbf{E} \end{bmatrix} \text{ will be a } p \times p \text{ full rank matrix. Additionally, } \begin{bmatrix} \tilde{Z} \\ \tilde{W} \end{bmatrix} = \begin{bmatrix} \mathbf{D} \\ \mathbf{E} \end{bmatrix} \tilde{X}$$

Therefore for every rank deficient matrix $\mathbf{D}_{q\times p}$, there exists an invertible full rank matrix $\mathbf{C}$ of size $p \times p$. Clearly, $(\tilde{Z}^T, \tilde{W}^T)^T$ is a non-singular transformation of $\tilde{X}$. Since, $\tilde{X}$ follows a multivariate normal distribution, the vector $(\tilde{Z}^T, \tilde{W}^T)^T$ is Gaussian as well according to Theorem 1.3. Then according to the Theorem 1.4, $\tilde{Z}$ has a marginal normal density. In addition to that, Theorem 1.4 also implies that

$$\mathbb{E}\left\{ \begin{bmatrix} \tilde{Z} \\ \tilde{W} \end{bmatrix} \right\} = \begin{bmatrix} \mathbf{D} \\ \mathbf{E} \end{bmatrix} \mathbb{E}[\tilde{X}] \text{ and } \mathcal{V}\left\{ \begin{bmatrix} \tilde{Z} \\ \tilde{W} \end{bmatrix} \right\} = \begin{bmatrix} \mathbf{D}\Sigma\mathbf{D}^T & \mathbf{D}\Sigma\mathbf{E}^T \\ \mathbf{E}\Sigma\mathbf{D}^T & \mathbf{E}\Sigma\mathbf{E}^T \end{bmatrix} \tag{1.22}$$

Therefore, $\tilde{Z} \sim \mathcal{N}_p(\mathbf{D}\tilde{\mu}, \mathbf{D}\Sigma\mathbf{D}^T)$ □

Theorem 1.6. *Let $\tilde{X}$ be partitioned into $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ as usual. The notations $\tilde{\mu}^{(1)}$, $\tilde{\mu}^{(2)}$, Σ_{11}, Σ_{22}, Σ_{12}, and Σ_{21} mean the same as the way they were defined earlier. Let $\tilde{X} \sim \mathcal{N}_p(\tilde{\mu}, \Sigma)$. Then the conditional distribution of $\tilde{X}^{(1)}$ given $\tilde{X}^{(2)} = \tilde{x}^{(2)}$ is a q-variate normal density. In particular, it is:*
$n(\tilde{x}^{(1)}|\ \tilde{\mu}^{(1)} + \Sigma_{12}\Sigma_{22}^{-1}(\tilde{x}^{(2)} - \tilde{\mu}^{(2)}),\ \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$

Proof. Consider the transformations we defined in Theorem 1.4. We also proved that $\tilde{Y}^{(1)} \sim \mathcal{N}(\tilde{\mu}^{(1)} - \Sigma_{12}\Sigma_{22}^{-1}\tilde{\mu}^{(2)}, \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$, $\tilde{Y}^{(2)} \sim \mathcal{N}(\tilde{\mu}^{(2)}, \Sigma_{22})$ and $\tilde{Y}^{(1)} \perp \tilde{Y}^{(2)}$. These transformations have been re-defined here for convenience.

$$Y^{(1)} = X^{(1)} + \mathbf{B}X^{(2)} = X^{(1)} - \Sigma_{12}\Sigma_{22} \text{ and } Y^{(2)} = X^{(1)} \tag{1.23}$$

Then the joint density of $\tilde{Y}^{(1)}$ and $\tilde{Y}^{(2)}$ is given by:

$$f_{\tilde{Y}^{(1)},\tilde{Y}^{(2)}}(\tilde{y}^{(1)}, \tilde{y}^{(2)}) = n(\tilde{y}^{(1)}|\tilde{\mu}^{(1)} - \Sigma_{12}\Sigma_{22}^{-1}\tilde{\mu}^{(2)}, \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}) \times n(\tilde{y}^{(2)}|\tilde{\mu}^{(2)}, \Sigma_{22})$$

The joint density of $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ can be obtained using the joint density of $\tilde{Y}^{(1)}$ and $\tilde{Y}^{(1)}$. The Jacobian of the transformation (1.23) is 1. Therefore, the joint density of $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ can be obtained by replacing $\tilde{y}^{(1)}$ by $\tilde{x}^{(1)} - \Sigma_{12}\Sigma_{22}^{-1}\tilde{x}^{(2)}$ and replacing $\tilde{y}^{(2)}$ by $\tilde{x}^{(2)}$. Simple calculations will show that the joint density of $\tilde{x}^{(1)}$ and $\tilde{x}^{(2)}$ is:

$$f_{\tilde{X}^{(1)},\tilde{X}^{(2)}}(\tilde{x}^{(1)}, \tilde{x}^{(2)}) = n(\tilde{x}^{(1)}|\tilde{\mu}^{(1)} + \Sigma_{12}\Sigma_{22}^{-1}(\tilde{x}^{(2)} - \tilde{\mu}^{(2)}), \Sigma_{11.2}) \times n(\tilde{x}^{(2)}|\tilde{\mu}^{(2)}, \Sigma_{22})$$

where $\Sigma_{11.2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$. Since $\tilde{X}^{(2)}$ has marginal normal density with mean $\tilde{\mu}$ and variance Σ_{22}, we can use the definition of conditional distribution to come to the conclusion that:

$$f(x^{(1)}|X^{\tilde{(2)}} = \tilde{x}^{(2)}) = n(\tilde{x}^{(1)}|\tilde{\mu}^{(1)} + \Sigma_{12}\Sigma_{22}^{-1}(\tilde{x}^{(2)} - \tilde{\mu}^{(2)}), \Sigma_{11.2})$$

□

Observation

1. $\mathbb{E}[\tilde{X}^{(1)}|\tilde{X}^{(2)} = \tilde{x}^{(2)}] = \tilde{\mu}^{(1)} + \Sigma_{12}\Sigma_{22}^{-1}(\tilde{x}^{(2)} - \tilde{\mu}^{(2)})$ depends linearly on the components held fixed i.e. $\{\tilde{x}_1^{(2)}, \cdots, \tilde{x}_{p-q}^{(2)}\}$

2. $\mathcal{V}[\tilde{X}^{(1)}|\tilde{X}^{(2)} = \tilde{x}^{(2)}] = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$. Since $\tilde{X}^{(2)}$ does not depend on the fixed value taken $\tilde{X}^{(2)}$

1.5 Characteristic Functions

Definition 1.5. The characteristic function of a $p \times 1$ random vector $\tilde{X}$ is given by $\phi : \mathbb{R}^p \longrightarrow \mathbb{C}$ such that

$$\boxed{\phi(\tilde{t}) = \mathbb{E}[\exp i\tilde{t}^T\tilde{X}] \text{ for all } \tilde{t} \in \mathbb{R}^p} \text{ and } \; i \text{ here denotes the imaginary number} \tag{1.24}$$

In addition to that,

$$\phi(\tilde{t}) = \mathbb{E}[\cos(\tilde{t}^T\tilde{X})] + i\mathbb{E}[\sin(\tilde{t}^T\tilde{X})] \text{ for all } \tilde{t} \in \mathbb{R}^p$$

Clearly, the Definition 1.24 is very similar to that of moment generating function.

Definition 1.6. Let $\tilde{X}$ be a random vector. Then the function $M_{\tilde{X}}(\tilde{t}) = \mathbb{E}[\exp \tilde{t}^T\tilde{X}]$ in some neighbourhood of origin.

One of the most useful properties of the moment generating function (MGF) is that the existence of an MGF uniquely determines a distribution function. However, the major drawback of the MGF is that it may not always exist for a given random vector. This is because of the strong requirement that $M_{\tilde{X}}(\tilde{t})$ has to exist in some neighbourhood of origin. So, we wish to find a generating function that exists for all random vectors and also uniquely corresponds to a particular distribution function. The function with these desirable properties is the characteristic function.

1.5.1 Properties of the characteristic function

1. Characteristic function always exists for $p \times 1$ random vector $\tilde{X}$ i.e. $\mathbb{E}[\exp i\tilde{t}\tilde{X}] < \infty$ for all $\tilde{t} \in \mathbb{R}^p$.
   This is because:
   $$\mathbb{E}[\exp\{i\tilde{t}\tilde{X}\}] = \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} \exp\{i\tilde{t}\tilde{X}\} f_{\tilde{X}}(\tilde{x}) \mathrm{d}x_1 \cdots \mathrm{d}x_p \leqslant \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} |\exp\{i\tilde{t}\tilde{X}\}||f_{\tilde{X}}(\tilde{x})| \mathrm{d}x_1 \cdots \mathrm{d}x_p =$$
   In other words $\phi(\tilde{X}) \leqslant 1$ for all $\tilde{t} \in \mathbb{R}^p$.
   However, in the case of a moment generating function, it cannot always exist because:
   $$\mathbb{E}[e^{tx}] \leqslant \int_{-t_0}^{t_0} e^{tx} f_X(x) \mathrm{d}x \leqslant \int_{-t_0}^{t_0} e^{tt_0} f_X(x) \mathrm{d}x \longrightarrow \infty \text{ as } t \longrightarrow \infty$$

2. $\phi_{-\tilde{X}}(\tilde{t}) = \overline{\phi_{\tilde{X}}(\tilde{t})}$ where the bar indicates complex conjugate. The reason is given below:
   $$\phi_{-\tilde{X}}(\tilde{t}) = \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} \exp\{-i\tilde{t}\tilde{X} f_{\tilde{X}}(\tilde{x})\} \mathrm{d}x_1 \cdots \mathrm{d}x_p = \overline{\int_{\mathbb{R}} \cdots \int_{\mathbb{R}} \exp\{i\tilde{t}\tilde{X} f_{\tilde{X}}(\tilde{x})\} \mathrm{d}x_1 \cdots \mathrm{d}x_p} = \overline{\phi_{\tilde{X}}(\tilde{t})}$$
   This implies that the characteristic function of $-\tilde{X}$ is the complex conjugate of the characteristic function of $\tilde{X}$.

3. Let X be any random variable with characteristic function $\phi_X(t)$ for all $t \in \mathbb{R}$. Then ϕ_X is uniformly continuous on $\mathbb{R}$. Without loss of generality, one can assume that $t > s$ and $t = s + h$ for some $h > 0$. Then simple calculations can show that:

   $$|\phi_X(t) - \phi_X(s)| \leqslant \int_{\mathbb{R}} |e^{isx}(e^{ihx} - 1) f_X(x)| \mathrm{d}x \leqslant \int_{\mathbb{R}} |e^{ihx} - 1| f_X(x) \mathrm{d}x \leqslant \int_{\mathbb{R}} 2 f_X(x) \mathrm{d}x$$

   Using the Dominated convergence theorem shows that:

   $$\lim_{h\to 0} \int_{\mathbb{R}} |e^{ihx} - 1| f_X(x) \mathrm{d}x = \int_{\mathbb{R}} \lim_{h\to 0} |e^{ihx} - 1| f_X(x) \mathrm{d}x = 0$$

   Therefore, ϕ_X is uniformly continuous on $\mathbb{R}$.

4. Let $\tilde{X}$ and $\tilde{Y}$ be independent random vectors. Then, $\boxed{\phi_{\tilde{X}+\tilde{Y}}(\tilde{t}) = \phi_{\tilde{X}}(\tilde{t}) \times \phi_{\tilde{Y}}(\tilde{t}) \forall \tilde{t} \in \mathbb{R}^p}$ This can easily be proved using the definitions of characteristic function and independence of random vectors.

Definition 1.7. Let g be a complex valued function i.e. $g(\tilde{x}) = g_1(\tilde{x}) + i g_2(\tilde{x})$ for some real valued functions g_1 and g_2. Then,

$$\mathbb{E}[g(\tilde{x})] = \mathbb{E}[g_1(\tilde{x})] + i\mathbb{E}[g_2(\tilde{x})] \tag{1.25}$$

Lemma 1.4. *Let $\tilde{X}^T = (\{\tilde{X}^{(1)}\}^T, \{\tilde{X}^{(2)}\}^T)$. If $\tilde{X^{(1)}}$ and $\tilde{X^{(1)}}$ are independent and $g(\tilde{x}) = g^{(1)}(\tilde{x}^{(1)}) g^{(2)}(\tilde{x}^{(2)})$, then*

$$\mathbb{E}[g(\tilde{x})] = \mathbb{E}[g^{(1)}(\tilde{x}^{(1)})] \times \mathbb{E}[g^{(2)}(\tilde{x}^{(2)})]$$

Proof. The lemma can be proved using simple manipulations, definition 1.7, and the definition of independence. □

Certain special cases of this theorem provide us with quite useful results:

1. Let $g(\tilde{x}) = \exp\{i\tilde{t}^T \tilde{X}\}$ in the above lemma. Then the lemma now is equivalent to:

   $$\tilde{X}^{(1)} \perp \tilde{X}^{(2)} \implies \phi_{\tilde{X}^{(1)}+\tilde{X}^{(2)}}(\tilde{t}) = \phi_{\tilde{X}^{(1)}}(\tilde{t}) \times \phi_{\tilde{X}^{(2)}}(\tilde{t})$$

2. Now, let $\tilde{X}$ have independent components $\{X_1, \cdots, X_p\}$ i.e. $f_{\tilde{X}}(\tilde{x}) = \prod_{i=1}^p f_{X_i}(x_i)$. Then the lemma above can be used to prove that:

   $$\phi_{\tilde{X}}(\tilde{t}) = \prod_{i=1}^{p} \phi_{X_j}(t_j) \tag{1.26}$$

Theorem 1.7. *Let $\tilde{X} \sim \mathcal{N}_p(\mu, \Sigma)$. Then the characteristic function of $\tilde{X}$ is given by:*

$$\phi_{\tilde{X}}(\tilde{t}) = \exp\left\{i\tilde{t}^T\tilde{\mu} - \frac{1}{2}\tilde{t}^T\Sigma\tilde{t}\right\} \; \forall\tilde{t} \in \mathbb{R}^p \tag{1.27}$$

Proof. Σ is a symmetric, positive definite matrix $\implies$ there exists an invertible matric $\mathbf{C}$ such that $\mathbf{C}^T\mathbf{\Sigma}^{-1}\mathbf{C} = \mathcal{I}_p$. Consider the following transformation that we considered earlier:

$$\tilde{Y} = \mathbf{C}^{-1}(\tilde{X} - \tilde{\mu}) \tag{1.28}$$

The Jacobian of the above transformation is $\mathcal{J} = |\det\mathbf{C}|$. Then the density of $\tilde{Y}$ is given by:

$$f_{\tilde{Y}}(\tilde{y}) = f_{\tilde{x}(\tilde{y})} \times |\mathcal{J}| = \frac{1}{(2\pi)^{\frac{p}{2}}} \exp\{\frac{-1}{2}\tilde{y}^T\tilde{y}\}$$

That is, $\tilde{Y} \sim \mathcal{N}(\tilde{0}, \mathcal{I})$. Since $cov(Y_i, Y_j) = 0 \forall i \neq j$, $Y_i \perp Y_j$ for all $i \neq j$. Then, using 1.26 we can write that:

$$\phi_{\tilde{Y}}(\tilde{u}) = \mathbb{E}[\exp\{i\tilde{u}^T\tilde{Y}\}] = \prod_{j=1}^{p}\mathbb{E}[\exp\{iu_iY_j\}] \text{ where } Y_i \sim \mathcal{N}(0,1) \text{ for all } j \in \{1, 2, \cdots, p\} \tag{1.29}$$

We know that the normal univariate characteristic function of Y_j is $\exp\{\frac{-u_j^2}{2}\}$. Therefore, using 1.29 the characteristic function pf $\tilde{Y}$ is:

$$\phi_{\tilde{Y}}(u_1, \cdots, u_p) = \exp\{\frac{-1}{2}\tilde{u}^T\tilde{u}\}$$

Now, using the transformation 1.28, the characteristic function of $\tilde{X}$ can be written as:

$$\phi_{\tilde{X}}(\tilde{t}) = \mathbb{E}[\exp\{i\tilde{t}^T(\mathbb{C}\tilde{Y} + \tilde{\mu})\}] = \mathbb{E}[\exp\{i\tilde{t}^T(\mathbf{C}\tilde{Y})\}] \times \mathbb{E}[\exp\{i\tilde{t}^T\tilde{\mu}] \tag{1.30}$$

$\mathbb{E}[\exp\{i\tilde{t}^T\tilde{\mu}]$ is a constant.Since $\mathbf{C}$ is invertible and $\tilde{t} \in \mathbb{R}^p$, the term $\tilde{t}^T\mathbb{C}$ also spans $\mathbb{R}^p$. Therefore,

$$\mathbb{E}[\exp\{i\tilde{t}^T(\mathbf{C}\tilde{Y})\}] = \phi_{\tilde{Y}}(\mathbf{C}^T\tilde{t}) = \exp\left\{\frac{-1}{2}\tilde{t}^T\Sigma\tilde{t}\right\} \tag{1.31}$$

Using (1.31) in (1.30), we can prove that the characteristic function of $\tilde{X}$ is

$$\phi_{\tilde{X}}(\tilde{t}) = \exp\left\{i\tilde{t}^T\tilde{\mu} - \frac{1}{2}\tilde{t}^T\Sigma\tilde{t}\right\} \; \forall\tilde{t} \in \mathbb{R}^p$$

□

Theorem 1.8. *Real valued random variables $X_1, X_2, \cdots, X_p$ are jointly Gaussian*

if and only if

$\tilde{a}^T\tilde{X} = \sum_{i=1}^{n} a_iX_i$ *is Gaussian for all* $\tilde{a} \in \mathbb{R}^p$

Proof. ($\Rightarrow$) Assume that $(X_1, X_2, \cdots, X_p)^T$ is jointly Gaussian. Therefore,

$$\phi_{\tilde{X}}(\tilde{t}) = \exp\left\{i\tilde{t}^T\tilde{\mu} - \frac{1}{2}\tilde{t}^T\Sigma\tilde{t}\right\} \ \forall\tilde{t} \in \mathbb{R}^p$$

Let $U = \sum_{i=1}^{p} a_iX_i$, then it can be shown that:

$$\phi_U(u) = \mathbb{E}[\exp\{i\tilde{a}(u\tilde{X})\}] = \phi_{u\tilde{X}}(\tilde{a}) = \exp\left\{i\tilde{a}^T(u\tilde{\mu}) - \frac{1}{2}\tilde{a}^T(u^2\Sigma)\tilde{a}\right\} \ \forall\tilde{t} \in \mathbb{R}^p$$

It is clear from the characteristic function of U that $U \sim \mathcal{N}_p(\tilde{a}^T\tilde{\mu}, \tilde{a}^T\Sigma\tilde{a})$

($\Leftarrow$) Conversely, let $\Sigma_{i=1}^{p}a_iX_i$ be Gaussian for all $\tilde{a} \in \mathbb{R}^p$. Then, each component X_i is Gaussian as well as $X_i = \tilde{e}_i^T\tilde{X}$. Assume that $X_i \sim \mathcal{N}_p(\mu_i, \sigma_i^2)$ such that μ_i and $\sigma_i^2 < \infty$ for all $i = 1, 2, \cdots, p$. Now, from Cauchy-Schwarz inequality,

$$\mathbb{E}[X_iX_j] \leqslant (\mathbb{E}[X_i^2]\mathbb{E}[X_j^2])^{\frac{1}{2}} \tag{1.32}$$

Now variance of $\tilde{X}$ is the matrix with $cov(X_i, X_j)$ as the $(i, j)th$ entry.$cov(X_i, X_j) < \infty \implies \mathcal{V}[X_i] < \infty \implies \mathbb{E}[X_i^2] < \infty$. Then, from (1.32), it is clear that variance and mean of $\tilde{X}$ are well defined. When $U = \sigma_{i=1}^{p}u_iX_i = \tilde{u}^T\tilde{X}$, then $\mathbb{E}[U] = \tilde{u}^T\tilde{X}$ and $\mathcal{V}[U] = \tilde{u}^T\Sigma\tilde{u}$ where $\tilde{u}^T = (u_1, u_2, \cdots, u_p)$. Then it can be shown that:

$$\phi_{\tilde{U}}(t) = \mathbb{E}[\exp\{i(t\tilde{u})^T\tilde{X}\}] = \text{ characteristic function of a normal distribution}$$

Since characteristic function uniquely determines the corresponding density function, the claim that $\tilde{X}$ is jointly Gaussian is proved. □

Remark 1.5. The above theorem proves that the multivariate Gaussian is the only distribution with the property that every linear combination of its components is normally distributed.

1.6 Multiple correlation coefficient

We earlier proved that the conditional density of $\tilde{X}^{(1)}$ given $\tilde{X}^{(1)} = \tilde{x}^{(2)}$, when $\tilde{X}$ follows a multivariate Gaussian distribution, is also normal with mean $\tilde{\mu}^{(1)} + \Sigma_{12}\Sigma_{22}^{-1}(\tilde{x}^{(2)} - \tilde{\mu}^{(2)})$ and variance $\Sigma_{11.2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$. In this section, we denote the matrix $\Sigma_{12}\Sigma_{22}^{-1}$ by $\mathbf{B}$. The general $(i, j)^{th}$ element of $\mathbf{B}$ is denoted by $b_{ij} = \sum_{k=1}^{p-q} cov(X_i^{(1)}, X_k^{(2)})cov(X_k^{(1)}, X_j^{(2)})$.

Definition 1.8. Let the (i, j)th element of the matrix $\Sigma_{11.2}$ be given by $\sigma_{ij.q+1,\cdots,p}$. Then:

1. $\sigma_{ij.q+1,\cdots,p}$ for $i \neq j$ is called the **partial covariance**.

2. $\sigma_{ij.q+1,\cdots,p}$ (i.e. i = j) is called **partial variance**.

Note that the letters that follow a dot in the symbol $\sigma_{ij.q+1,\cdots,p}$ are the components of $\tilde{X}$ that are given to us.

Definition 1.9. The **partial correlation** between X_i and X_j while holding $X_{q+1}, \cdots, X_p$ fixed is given by:

$$\rho_{ij.q+1,\cdots,p} = \frac{\sigma_{ij.q+1,\cdots,p}}{\sqrt{\sigma_{ii.q+1,\cdots,p}}\sqrt{\sigma_{jj.q+1,\cdots,p}}}$$

The matrix $\mathbf{B} = \Sigma_{12}\Sigma_{22}^{-1}$ is called the **matrix of regression coefficients of $\tilde{\mathbf{X}}^{(\mathbf{1})}$ on $\tilde{\mathbf{X}}^{(\mathbf{2})}$**. The function $\tilde{\mu}^{(1)} + \mathbf{B}(\tilde{x}^{(2)} - \tilde{\mu}^{(2)})$ is called **regression function**. Additionally, the **vector of residuals of $\tilde{\mathbf{X}}^{(1)}$ from its regression on $\tilde{\mathbf{X}}^{(2)}$** is given by:

$$\tilde{X}^{(1.2)} = \tilde{X}^{(1)} - \tilde{\mu}^{(1)} - \mathbf{B}(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})$$

The reason why they are called that way can only be understood after going through this section.

Let $\tilde{X}$ be partitioned into $\tilde{X}^{(1)}$ and $\tilde{X}^{(2)}$ as usual. In the reminder of the section, our aim to study the properties of the vector $\mathbf{B}\tilde{X}^{(2)}$.

Theorem 1.9. *The components of $\tilde{X}^{(1.2)}$ are uncorrelated with components of $\tilde{X}^{(2)}$.*

Proof. Observe that $\tilde{X}^{(1.2)} = \tilde{Y}^{(1)} - \mathbb{E}[\tilde{Y}^{(1)}]$ where $\tilde{Y}^{(1)} = \tilde{X}^{(1)} - \mathbf{B}\tilde{X}^{(2)}$. Then according to Theorem 1.4, the covariance matrix of Y is given by:

$$\mathcal{V}[\tilde{Y}] = \begin{bmatrix} \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} & \mathbf{0} \\ \mathbf{0} & \Sigma_{22} \end{bmatrix} \tag{1.33}$$

Using (1.33),

$$cov(\tilde{X}^{(1.2)}, \tilde{X}^{(2)}) = cov(\tilde{Y}^{(1)}, \tilde{X}^{(2)}) = cov(\tilde{X}^{(1)}, \tilde{X}^{(2)}) - \mathbf{B}cov(\tilde{X}^{(2)}, \tilde{X}^{(2)}) = -\Sigma_{12} = \tilde{0}$$

□

Intuitively, the effect of $\tilde{X}^{(2)}$ has been removed from $\tilde{X}^{(1.2)}$

Theorem 1.10. *For every vector $\tilde{\alpha} \in \mathbb{R}^q$,*

$$\mathcal{V}[X_i^{(1.2)}] \leqslant \mathcal{V}[X_i - \tilde{\alpha}^T\tilde{X}^{(2)}]$$

Proof. Let $\sigma_{(i)}^T$ denote the i^{th} row of Σ_{12} while $\tilde{\beta}_i^T$ denotes the i^{th} row of $\mathbf{B}$ such that they follow $\beta_i^T = \sigma_{(i)}^T \Sigma_{22}^{-1}$. Using the Theorem 1.9, we can show that:

$$\mathcal{V}[X_i - \tilde{\alpha}^T \tilde{X}^{(2)}] = \mathbb{E}[X_i - \mu_i - \tilde{\alpha}^T(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})]^2 = \mathbb{E}[\tilde{X}_i^{(1.2)}]^2 + \mathbb{E}[(\beta_{(i)}^T - \alpha^T)(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})]^2 \tag{1.34}$$

Then the term $\mathbb{E}[(\beta_{(i)}^T - \alpha^T)(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})]^2$ in (1.34) can be simplified to give the following result:

$$\mathbb{E}[(\beta_{(i)}^T - \alpha^T)(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})]^2 = (\tilde{\beta}_{(i)} - \tilde{\alpha})^T \Sigma_{22} (\tilde{\beta}_{(i)} - \tilde{\alpha})$$

All covariance matrices are positive semi-definite matrices. Therefore, $(\tilde{\beta}_{(i)} - \tilde{\alpha})^T \Sigma_{22} (\tilde{\beta}_{(i)} - \tilde{\alpha}) \geqslant 00$. Hence, one of the points at which (1.34) is minimized is at $\tilde{\alpha} = \tilde{\beta}_{(i)}$. Additionally, $\mathbb{E}[\tilde{X}_i^{(1.2)}]^2 = \mathcal{V}[\tilde{X}_i^{(1.2)}]$. These facts together imply that $\mathcal{V}[X_i^{(1.2)}] \leqslant \mathcal{V}[X_i - \tilde{\alpha}^T \tilde{X}^{(2)}]$. □

Remark 1.6.

1. If Σ_{22} is the covariance matrix associated with a normal density, then, minimum is attained at the unique vector $\bar{\alpha} = \tilde{\beta}_{(i)}$. However, if Σ_{22} is positive semi definite and not positive definite, then the maximizing vector is not unique.

2. It follows from the theorem just proved that $X_i^{(1)} = \tilde{X}_i^{\;(1)} - \tilde{\mu}_i^{(1)} - \tilde{\beta}_{(i)}^T(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})$. Therefore, $\tilde{\mu}_i^{(1)} + \tilde{\beta}_{(i)}^T(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})$ is the best estimator in the sense that it has the least variance among all linear estimators.

Corollary 1.2. *$corr(X_i, \tilde{\beta}_{(i)}^T \tilde{X}^{(2)}) \geqslant corr(X_i, \tilde{\alpha}_{(i)}^T \tilde{X}^{(2)})$ for all vectors $\tilde{\alpha} \in \mathbb{R}^{p-q}$*

Proof. Let $Y = \tilde{\alpha}^T(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})$. Correlation is defined as follows:

$$corr(X, Y) = \frac{cov(X, Y)}{\sqrt{\mathcal{V}[X]}\sqrt{\mathcal{V}[Y]}}$$

The right and left hand side of $corr(X_i, \tilde{\beta}_{(i)}^T \tilde{X}^{(2)}) \geqslant corr(X_i, \tilde{\alpha}_{(i)}^T \tilde{X}^{(2)})$ have the terms $\sqrt{\mathcal{V}[\alpha^T \tilde{X}^{(2)}]}$ and $\sqrt{\mathcal{V}[\beta_{(i)}^T \tilde{X}^{(2)}]}$ respectively, both of which have some finite variance value. But the vectors $\tilde{\alpha}^T \tilde{X}^{(2)}$ and $\tilde{\beta}_{(i)}^T \tilde{X}^{(2)}$ can be re-scaled to make them have unit variance. Rescaling does not change the correlation values, therefore, the following can be assumed:

$$\mathbb{E}[\tilde{\alpha}^T(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})]^2 = \mathbb{E}[\tilde{\beta}_{(i)}^T(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})]^2$$

We have proved in Theorem 1.10 that $\mathcal{V}[X_i^{(1.2)}] = \mathcal{V}[X_i - \tilde{\beta}_{(i)}^{\;T} \tilde{X}^{(2)}]$. Using the formula $\mathcal{V}[X + Y] = \mathcal{V}[X] + \mathcal{V}[Y] + 2cov(X, Y)$, we can simplify $\mathcal{V}[X_i^{(1.2)}] \leqslant \mathcal{V}[X_i - \tilde{\alpha}^T \tilde{X}^{(2)}]$ as follows:

$$\sigma_{ii} - 2\mathbb{E}[(X_i - \mu_i)\tilde{\beta}_{(i)}^T(\tilde{X}_{(2)} - \tilde{\mu}^{(2)})] + \mathcal{V}[\tilde{\beta}_{(i)}^T \tilde{X}^{(2)}] \leqslant \sigma_{ii} - 2\mathbb{E}[(X_i - \mu_i)\tilde{\alpha}^T(\tilde{X}_{(2)} - \tilde{\mu}^{(2)}) + \mathcal{V}[\tilde{\alpha}^T \tilde{X}^{(2)}].$$

The statement we wish to prove follows from the above equation. □

Remark 1.7. The theorem above states that the linear combination of $X_i^{(2)}$s that "best" linearly approximate X_i is $\tilde{\beta}_{(i)}^T \tilde{X}^{(2)}$ in the sense that it has least variance among all other linear estimators $\tilde{\alpha}^T \tilde{X}^{(2)}$.

Definition 1.10. The maximum correlation between $X_i and \alpha^T \tilde{X}^{(2)}$, which we have proved as $corr(X_i, \tilde{\beta}_{(i)}^T \tilde{X}^{(2)})$, is called the **multiple correlation coefficient** between X_i and $\tilde{X}^{(2)}$. From the definition, it is understood that the multiple correlation coefficient measures the maximum linear association between X_i and $\tilde{X}^{(2)}$.

Multiple correlation coefficient between X_i and $\{\tilde{X}^{(2)}\}^T = (X_{p+1}^{(2)}, \cdots, X_q^{(2)})$ is denoted by $\boxed{\bar{R}_{i.q+1,\cdots,p}}$. This coefficient can be simplied as follows:

$$\bar{R}_{i.q+1,\cdots,p} = \frac{\mathbb{E}[(X_i - \mu_i)\tilde{\beta}_{(i)}^T(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})]}{(\sigma_{ii}\mathcal{V}[\tilde{\beta}_{(i)}^T])^{\frac{1}{2}}} = \frac{\sqrt{\tilde{\sigma}_{(i)}^T \Sigma_{22}^{-1} \tilde{\sigma}_{(i)}}}{\sqrt{\sigma_{ii}}} \tag{1.35}$$

Lemma 1.5. *Let* $\mathbf{A}$ *be a* $p \times p$ *square matrix partitioned as given below:*

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$$

Let $\mathbf{A}_{11}$ *be an invertible* $q \times q$ *matrix. Then,*

$$\det(\mathbf{A}) = \det(\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}) \times \det(\mathbf{A}_{22})$$

Proof. Define the following block matrices:

$$\mathbf{B} = \begin{bmatrix} \mathcal{I}_q & -\mathbf{A}_{12}\mathbf{A}_{22}^{-1} \\ \mathbf{0} & \mathcal{I}_{p-q} \end{bmatrix} \text{ and } \mathbf{C} = \begin{bmatrix} \mathcal{I}_q & \mathbf{0} \\ -\mathbf{A}_{12}\mathbf{A}_{22}^{-1} & \mathcal{I}_{p-q} \end{bmatrix}$$

One can verify that the matrix multiplication of matrices $\mathbf{B}, \mathbf{A}$ and $\mathbf{C}$, in the order mentioned is:

$$\mathbf{BAC} = \begin{bmatrix} \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{22} \end{bmatrix} \implies \det(\mathbf{BAC}) = \det(\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}) \times \det(\mathbf{A}_{22}) \tag{1.36}$$

However, the way the matrices $\mathbf{B}$ and $\mathbf{C}$ have been defined, it is clear that $\det(\mathbf{B})$ and $\det(\mathbf{C})$ is 1. This along with (1.36) proves the lemma. □

Theorem 1.11. *The partial variance of any component of* $\tilde{X}$ *cannot be greater than its corresponding variance.*

Proof. From equation (1.35), we can show that:

$$1 - \bar{R}^2 = \frac{\sigma_{ii} - \sigma_{(i)}^T \Sigma_{22}^{-1} \sigma_{(i)}}{\sigma_{ii}} \tag{1.37}$$

where we sometimes write $\bar{R}$ in short for $\bar{R}_{i.q+1,\cdots,p}$

Definition 1.11. Define a block matrix Σ_i as follows:

$$\begin{bmatrix} \sigma_{ii} & \tilde{\sigma}_{(i)}^T \\ \tilde{\sigma}_{(i)} & \Sigma_{22} \end{bmatrix}$$

Then, according to the Lemma 1.5,

$$\det\Sigma_i = \det(\sigma_{ii} - \tilde{\sigma}_{(i)}^T \Sigma_{22}^{-1} \tilde{\sigma}_{(i)}) \times \det\Sigma_{22} \tag{1.38}$$

Using (1.37) and (1.38), we can show that:

$$1 - \bar{R}^2 = \frac{\det(\Sigma_i)}{\sigma_{ii} \times \det(\Sigma_{22})} \tag{1.39}$$

We know that $\sigma_{ii.q+1,\cdots p}$ is the $(i,i)^{th}$ element of the matrix $\Sigma_{11.2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$. Then:

$$\sigma_{ii.q+1,\cdots p} = \sigma_{ii} - \tilde{\sigma}_{(i)}^T \Sigma_{22}^{-1} \tilde{\sigma}_{(i)}$$

This implies that

$$\sigma_{ii.q+1,\cdots,p} = \sigma_{ii}(1 - \bar{R}^2)$$

The term $\sigma_{ii.q+1,\cdots,p}$ is the variance of $\tilde{X}^{(1)}$ given that we know $\tilde{x}^{(2)}$ and therefore non-negative. This implies that $\bar{R}^2 \leqslant 1$. This in turn implies the claim we wished to prove.

$\square$

Remark 1.8. Some observations have been noted below as remarks

1. Now let us look closely at the equation $\sigma_{ii.q+1,\cdots,p} = \sigma_{ii}(1-\bar{R}^2)$. When $\bar{R}$ is closer to one, it implies that X_i is being approximated well using $\tilde{X}_{(2)}$. Since, X_i is being "well-explained" by $\tilde{X}^{(2)}$, it is only natural to assume that the variance of X_i when $\tilde{X}^{(2)}$ is known is low. This is what we predict from the above equation as well.

2. We earlier showed that $\tilde{X}_i^{(1)} = \tilde{X}_i^{(1)} - \tilde{\mu}_i^{(1)} - \tilde{\beta}_{(i)}^T(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})$ where $\tilde{\beta}_{(i)}^T$ is the i^{th} row of $\mathbf{B}$. This, in matrix form, can be written as $\tilde{X}^{(1)} = \tilde{\mu}^{(1)} + \Sigma_{12}\Sigma_{22}^{-1}(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})$. Thus, it is the best linear predictor of $\tilde{X}^{(2)}$ in the sense that it has least variance among all linear estimators of $\tilde{X}^{(2)}$. The proof that this is the best linear predictor of $\tilde{X}^{(1)}$ does not depend on the distribution that $\tilde{X}$ follows. Since $\tilde{X}^{(1)}$ can be estimated using $\tilde{\mu}^{(1)} + \Sigma_{12}\Sigma_{22}^{-1}(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})$, the term which is a function of $\tilde{X}^{(2)}$ is called the **regression funtion of $\tilde{\mathbf{X}}^{(\mathbf{1})}$ on $\tilde{\mathbf{X}}^{(\mathbf{2})}$**.

3. We know that the vector of residuals for $\tilde{X}^{(1)}$ will be $\tilde{X}^{(1)} - \widehat{\tilde{X}}^{(1)}$. Since, $\widehat{\tilde{X}}^{(1)} = \tilde{\mu}^{(1)} + \Sigma_{12}\Sigma_{22}^{-1}(\tilde{X}^{(2)} - \tilde{\mu}^{(2)})$ predicts $\tilde{X}^{(1)}$, we can call the vector

$$\tilde{X}^{(1)} - \tilde{\mu}^{(1)} - \Sigma_{12}\Sigma_{22}^{-1}(\tilde{X}^{(2)} - \tilde{\mu}^{(2)}).$$

is called the **vector of residuals**.

4. It can be shown that varaince of the residual vector defined above is $\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$. If $\tilde{X}$ is assumed to be a normal random vector, then the variance of the residual vector is same as the variance of $\tilde{X}^{(1)}$ provided $\tilde{X}^{(2)}$ is known. However, if $\tilde{X}$ is not normally distributed, then $\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$ can simply be regarded as the variance of the residual vector.

1.7 Moments and Cumulants

Definition 1.12. The k^{th} order moment of a random vector $\tilde{X}^T = (x_1, X_2, \cdots, X_p)$ is defined as $\mathbb{E}[X_{i_1}X_{i_2}\cdots X_{i_k}]$ where $\{i_1, i_2 \cdots, i_k\} \subseteq \{1, 2, \cdots, p\}$.

The following relationship between the moments and characteristic function $\phi_{\tilde{X}}(t_1, t_2, \cdots, t_k)$ of $\tilde{X}$ can easily be verified.

$$\mathbb{E}[X_{i_1}X_{i_2}\cdots X_{i_k}] = \frac{1}{i^k}\left.\frac{\partial^k \phi_{\tilde{X}}}{\partial t_1 \partial t_2 \cdots \partial t_k}\right|_{t=0} \tag{1.40}$$

Definition 1.13. Assuming all the moments of $\tilde{X}$ exist, the cumulants of $\tilde{X}$ are defined as coefficients, κ, in the multinomial Taylor series expansion (centered at origin) of $\ln \phi_{\tilde{X}}(t_1, t_2, \cdots, t_p)$.

In order to state the k^{th} order multinomial Taylor expansion in a simple manner, some notation needs to be mentioned.

- $\tilde{\alpha}^T = (\alpha_1, \alpha_2, \cdots, \alpha_p)$ is a p-index if it is p-tuple with $\alpha_i \geqslant 0$ $\forall i \in \{1, 2, \cdots, p\}$.
- $|\tilde{\alpha}| = \alpha_1 + \alpha_2 + \cdots + \alpha_p$
- $\tilde{\alpha}! = \alpha_1!\alpha_2!\cdots\alpha_p!$
- $\tilde{x}^{\tilde{\alpha}} = x_1^{\alpha_1}x_2^{\alpha_2}\cdots x_p^{\alpha_p}$
- Let f be a class $\mathcal{C}^{k+1}$ function on $\mathbb{R}^p$. Then,

$$\partial^{\tilde{\alpha}} f = \frac{\partial^{|\tilde{\alpha}|}}{\partial x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_p^{\alpha_p}}$$

Using the notations defined so far, the multinomial Taylor series expansion is mentioned below:

Let f be a real valued class $\mathcal{C}^{k+1}$ function on $\mathbb{R}^p$. Then the k^{th} order several variable expansion of f centered at the vector $\tilde{a}$ is given by:

$$f(\tilde{x}) = \sum_{|\tilde{\alpha}|\leqslant k} \frac{\partial^{\tilde{\alpha}} f(\tilde{a})}{\tilde{\alpha}!}(\tilde{x}-\tilde{a})^{\tilde{\alpha}} + R_k^f(\tilde{x},\tilde{a}) \tag{1.41}$$

where $R_k^f(\tilde{x},\tilde{a})$ denotes the reminder for the k^{th} order expansion.

If we use (1.41) to write an expansion for $\ln \phi_{\tilde{X}}(t_1, t_2, \cdots, t_p)$, one can show that for some function g:

$$\ln \phi_{\tilde{X}}(t_1,\cdots,t_p) = g(it_1,\cdots,it_p) = \sum_{|\tilde{\alpha}|<\infty} \frac{\partial^{|\tilde{\alpha}|} f(\tilde{0})}{\partial(it_1)^{\alpha_1}\partial(it_2)^{\alpha_2}\cdots\partial(it_p)^{\alpha_p}} \frac{(it_1)^{\alpha_1}(it_2)^{\alpha_2}\cdots(it_p)^{\alpha_p}}{\alpha_1!\alpha_2!\cdots\alpha_p!} \tag{1.42}$$

The the coefficients $\frac{\partial^{|\tilde{\alpha}|} f(\tilde{0})}{\partial(it_1)^{\alpha_1}\partial(it_2)^{\alpha_2}\cdots\partial(it_p)^{\alpha_p}}$ denoted by $\kappa_{\alpha_1,\alpha_2\cdots,\alpha_p}$ are called as the **cumulants** of $\tilde{X}$.

1.8 Geometry of the Gaussian density

Consider a finite dimensional vector space $\mathbf{V}$. Let $\mathcal{B}$ and $\mathcal{C}$ be two possible bases for the vector space $\mathbf{V}$. Let $\mathcal{B} = \{\tilde{v}_1, \tilde{v}_2, \cdots, \tilde{v}_n\}$ and $\mathcal{C} = \{\tilde{w}_1, \tilde{w}_2, \cdots, \tilde{w}_n\}$. Then there exists an invertible matrix $\mathbf{P}$ such that

$$\mathbf{P}[\tilde{v}]_B = [\tilde{v}]_C \tag{1.43}$$

Note: $[\tilde{v}]_B$ implies that the vector $\tilde{v}$, with respect to the basis $\mathcal{B}$, can be written as $\tilde{v} = \sum_{i=1}^n \alpha_i \tilde{v}_i$.

$$[\tilde{v}]_B \underset{\times\mathbf{P}^{-1}}{\overset{\times\mathbf{P}}{\rightleftarrows}} [\tilde{v}]_C \tag{1.44}$$

The result (1.43) is the main result we use to understand the geometry of the normal distribution. We start with the simpler case of a circle and then move towards an ellipse.

1.8.1 n-dimensional sphere: A special case of an ellipse

Consider the 1-sphere $\mathcal{S}^1 = \{\tilde{x} \in \mathbb{R}^2 | x_1^2 + x_2^2 = r^2\}$. Then, this equation can be written as:

$$\begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = r^2$$

Or more generally, if we consider an $(n-1)$-dimensional sphere $\mathcal{S}^{n-1} = \{\tilde{x} \in \mathbb{R}^n | x_1^2 + x_2^2 \cdots x_n^2 = r^2\}$, then it can be written as

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = r^2 \text{ or in short, we can write it as } \tilde{x}^T \mathcal{I}^{-1} \tilde{x} = r^2 \tag{1.45}$$

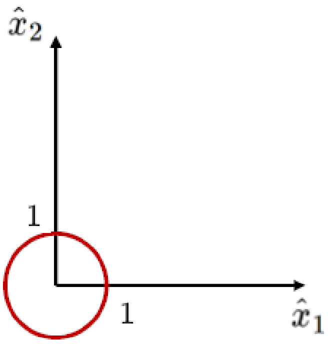


Figure 1.1: A 1-sphere in $\mathbb{R}^2$ - special case of an ellipse

1.8.2 Un-rotated ellipse with center at (0,0)

Consider the following ellipse (also shown in Fig. 1.2) with its center at the origin.

$$\boxed{\frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} = 1} \text{ which can also be written as } \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} \frac{1}{a^2} & 0 \\ 0 & \frac{1}{b^2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \text{ or } \boxed{\tilde{x}\Sigma_x^{-1}\tilde{x} = 1}$$

In short, this equation can be written as $\tilde{x}\Sigma_x^{-1}\tilde{x} = 1$ where $\tilde{x}^T = (x_1, x_2)$ and $\Sigma_x = \begin{bmatrix} a^2 & 0 \\ 0 & b^2 \end{bmatrix}$.

More generally, an n-dimentional "un-rotated" ellipse would be

$$\frac{x_1^2}{\alpha_1^2} + \frac{x_2^2}{\alpha_2^2} + \cdots \frac{x_n^2}{\alpha_n^2} = 1, \text{ where } \alpha_i \neq 0 \ \forall i$$

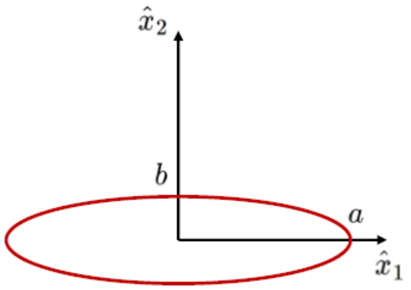


Figure 1.2: Unrotated ellipse with its center at the origin

This can also be written as follows:

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} \frac{1}{\alpha_1^2} & 0 & \cdots & 0 \\ 0 & \frac{1}{\alpha_2^2} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\alpha_n^2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \text{ or } \tilde{x}^T \Sigma_x^{-1} \tilde{x}$$

Here, Σ_x denotes the the matrix $\begin{bmatrix} \alpha_1^2 & 0 & \cdots & 0 \\ 0 & \alpha_2^2 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \alpha_n^2 \end{bmatrix}$

1.8.3 Un-rotated ellipse with a translated center

Here, we consider an un-rotated ellipse with its centre at $(c, d) \neq (0, 0)$. Since the centre (c, d) is obtained through translations from the point $(0, 0)$, this ellipse will be called an unrotated ellipse with translated origin. The two dimensional version of such an ellipse will be:

$$\frac{(x_1 - c_1)^2}{a^2} + \frac{(x_2 - c_2)^2}{b^2} = 1$$

Its generalized version in the n-dimensional world will be:

$$\frac{(x_1 - c_1)^2}{\alpha_1^2} + \frac{(x_2 - c_2)^2}{\alpha_2^2} + \cdots \frac{(x_n - c_n)^2}{\alpha_n^2} = 1 \text{ or } \boxed{(\tilde{x} - \tilde{\alpha})^T \Sigma_x^{-1} (\tilde{x} - \tilde{\alpha})}$$

where the matrix Σ_x is given by $\begin{bmatrix} \alpha_1^2 & 0 & \cdots & 0 \\ 0 & \alpha_2^2 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \alpha_n^2 \end{bmatrix}$, $\alpha_i > 0 \ \forall i \in \{1, 2, \cdots, n\}$

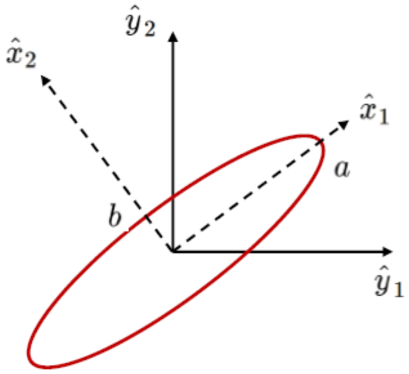


Figure 1.3: Unrotated ellipse with its center at the origin

1.8.4 Rotated ellipse with centre at origin

Consider a 2-dimensional ellipse in variables y_1 and y_2 which has its center at the origin. Such an ellipse will be called a rotated ellipse if its major and minor axes are not parallel to the directions $\widehat{y}_1$ and $\widehat{y}_2$ as shown in the figure. However, it is always possible to obtain a new set of variables $\widehat{x}_1$ and $\widehat{x}_2$ such that $\mathbf{P}\tilde{x} = \tilde{y}$ for some invertible matrix $\mathbf{P}$ and the major and minor axes are parallel to the directions of $\widehat{x}_1$ and $\widehat{x}_2$ in the new tranformed world involving variables x_1 and x_2.

Without loss of generality, let $\{\widehat{x}_1, \widehat{x}_2\}$ and $\{\widehat{y}_1, \widehat{y}_2\}$ be orthonormal bases for the space $\mathbb{R}^2$. In such a scenario, $\mathbf{P}$ will be an orthogonal matrix. To begin with, we know that the equation of the ellipse with respect to the new set of coordinates $\widehat{x}_1$ and $\widehat{x}_2$ is $\tilde{x}^T\Sigma_x^{-1}\tilde{x} = 1$ where $\tilde{x}^T = (x_1, x_2)$ and Σ_x is the diagonal matrix consisting of the squared values of the major and minor axes lengths.

Now, this information can easily be generalized to the n-dimensional case. In the n-dimensional space, we have the following information

$$\text{ellipse equation w.r.t } \{\widehat{x}_1, \widehat{x}_1\} : \tilde{x}\Sigma_x^{-1}\tilde{x} = 1 \text{ such that } \mathbf{P}\tilde{x} = \tilde{y}$$

Therefore,

$$\tilde{x}\Sigma_x^{-1}\tilde{x} = \tilde{y}^T\mathbf{P}\Sigma_x^{-1}\mathbf{P}^T\tilde{y} = 1 \tag{1.46}$$

From (1.46), it follows the the original rotated ellipse has the equation $\boxed{\tilde{y}^T\Sigma_y\tilde{y} = 1}$ where $\Sigma_y = \mathbf{P}\Sigma_x\mathbf{P}^T$.

1.8.5 Rotated and translated ellipse

Using the ideas and methods we used so far, it is easy to show that the equation of a rotated and translated ellipse is given by

$$\boxed{(\tilde{y} - \tilde{\alpha})^T\Sigma_y^{-1}(\tilde{y} - \tilde{\alpha}) = 1}$$

Here, $\tilde{\alpha}$ is the centre of the n-dimensional ellipse and $\Sigma_y = \mathbf{P}\Sigma_x^{-1}\mathbf{P}$ where Σ_x corresponds to the equation of an un-rotated ellipse with centre at $\tilde{\alpha}$.

Using the observations made so far, it is easy to conclude that the locus of all points $\tilde{x}$ such that $n(\tilde{x}|\ \tilde{\mu}, \Sigma) = c_0$ for some constant $c_0 \in \mathbb{R}$.

Chapter 2

Estimation of mean and covariance matrix

The references for this chapter is [And03].

We know that the multivariate normal distribution is completely known if the parameters $\tilde{\mu}$ and Σ are given to us. However, it is not always possible to know the mean vector or the covariance matrix. In such a scenario, we can try to estimate the mean and covariance of a random vector using a sample of observations we have. In this chapter, we discuss about the maximum likelihood estimators for the mean and covariance matrix of a random vector $\tilde{X}$, inference concerning the mean when the covariance matrix is known and the James-Stein estimator.

2.1 Maximum Likelihood estimation

Through out the section, we assume that we have N samples of observations on the random vector $\tilde{X} \sim \mathcal{N}(\tilde{\mu}, \Sigma)$. Let these observations be labelled as $\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_N$ where $N > p$. Then the likelihood function is given by:

$$\mathcal{L}(\tilde{\mu}, \Sigma | \tilde{x}_1, \cdots, \tilde{x}_N) = \prod_{\alpha=1}^{N} n(\tilde{x}_\alpha |\ \tilde{\mu}, \Sigma)$$

Note that, in a likelihood equation, the vectors $\tilde{x}_\alpha$ for all $\alpha \in \{1, 2, \cdots, N\}$ are fixed or known. However, $\tilde{\mu}$ and Σ are unknowns in the likelihood equation. In order to highlight the fact that $\tilde{\mu}$ and Σ are variables (NOT parameters), we use the symbols $\tilde{\mu}^*$ and Σ^* instead of $\tilde{\mu}$ and Σ. With this notation in mind, the log likelihood of the multivariate normal random vector is given by:

$$\ln \mathcal{L} = -\frac{1}{2} pN \ln(2\pi) - \frac{1}{2} N \ln(\det \Sigma^*) - \frac{1}{2} \sum_{\alpha=1}^{N} (\tilde{x}_\alpha - \tilde{\mu}^*)^T (\Sigma^*)^{-1} (\tilde{x}_\alpha - \tilde{\mu}^*) \tag{2.1}$$

We wish to find $\tilde{\mu}^*$ and $(\Sigma^*)^{-1}$ such that the log likelihood (and hence the likelihood) function is maximized.

The following notations will be used in this section for the sake of convenience.

1. The sample mean vector, denoted by $\vec{x}$, is defined as follows:

$$\bar{x} = \frac{1}{N}\sum_{\alpha=1}^{N}\tilde{x}_\alpha = \begin{bmatrix} \frac{1}{N}\sum_{\alpha=1}^{N} x_{1\alpha} \\ \frac{1}{N}\sum_{\alpha=1}^{N} x_{2\alpha} \\ \vdots \\ \frac{1}{N}\sum_{\alpha=1}^{N} \; x_{p\alpha} \end{bmatrix} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_p \end{bmatrix} \text{ where } \tilde{x}_\alpha = \begin{bmatrix} x_{\alpha 1} \\ x_{\alpha 2} \\ \vdots \\ x_{\alpha p} \end{bmatrix} \text{ for all } \alpha \in \{1, 2, \cdots, N\} \tag{2.2}$$

2. Define a matrix $\mathbf{A}$ as given below. This will be useful in defining the sample covariance matrix.

$$\mathbf{A} = \sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \bar{x})(\tilde{x}_\alpha - \bar{x})^T$$

It is easy to show that the aboce equation can be simplified to give

$$\mathbf{A} = \sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \bar{x})(\tilde{x}_\alpha - \bar{x})^T = \sum_{\alpha=1}^{N}\tilde{x}_\alpha\tilde{x}_\alpha^T - N\bar{x}\bar{x}^T$$

Our aim is to prove the following theorem:

Theorem 2.1. *If $\tilde{x}_1$, $\tilde{x}_2 \cdots$, $\tilde{x}_N$ are $p \times 1$ vectors sampled from a normal distribution with mean $\tilde{\mu}$ and covariance Σ such that $p < N$. Then the maximum likelihood estimator of $\tilde{\mu}$ and Σ, denoted by $\widehat{\mu}$ and $\widehat{\Sigma}$, are:*

$$\widehat{\mu} = \bar{x} = \frac{1}{N}\sum_{\alpha=1}^{N}\tilde{x}_\alpha \text{ and } \widehat{\Sigma} = \frac{\mathbf{A}}{N} = \frac{1}{N}\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \bar{x})(\tilde{x}_\alpha - \bar{x})^T \tag{2.3}$$

This theorem easily follows from the two lemmas that will be proved now.

Lemma 2.1. *Let $\tilde{x}_1$, $\tilde{x}_2$, $\cdots$, $\tilde{x}_N$ be N $p \times 1$ vectors. $\bar{x}$ denotes the mean vector as defined in* (2.2). *Then, for any vector $\tilde{b}$, we have the following:*

$$\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \tilde{b})(\tilde{x}_\alpha - \tilde{b})^T = \sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \bar{x})(\tilde{x}_\alpha - \bar{x})^T + N(\bar{x} - \tilde{b})(\bar{x} - \tilde{b})^T \tag{2.4}$$

Proof. The Lemma can be proved by re-writing and then later simplifying the left hand side of the equation 2.4 as follows:

$$\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \tilde{b})(\tilde{x}_\alpha - \tilde{b})^T = \sum_{\alpha=1}^{N}[(\tilde{x}_\alpha - \bar{x}) + (\bar{x} - \tilde{b})][(\tilde{x}_\alpha - \bar{x}) + (\bar{x} - \tilde{b})]^T$$

□

From the lemma just proved, the following result follows directly:

$$\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \tilde{\mu})(\tilde{x}_\alpha - \tilde{\mu})^T = \sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \bar{x})(\tilde{x}_\alpha - \bar{x})^T + N(\bar{x} - \tilde{\mu})(\bar{x} - \tilde{\mu})^T$$

The log likelihood function has been mentioned again for the sake of convenience.

$$\ln \mathcal{L} = -\frac{1}{2}pN \ln(2\pi) - \frac{1}{2}N \ln(\det\Sigma^*) - \frac{1}{2}\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \tilde{\mu}^*)^T(\Sigma^*)^{-1}(\tilde{x}_\alpha - \tilde{\mu}^*) \tag{2.5}$$

We now simplify the term $\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \tilde{\mu}^*)^T(\Sigma^*)^{-1}(\tilde{x}_\alpha - \tilde{\mu}^*)$ present in the log likelihood function.

$$\begin{aligned}
\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \tilde{\mu}^*)^T(\Sigma^*)^{-1}(\tilde{x}_\alpha - \tilde{\mu}^*) &= \text{trace}\Big\{\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \tilde{\mu}^*)^T(\Sigma^*)^{-1}(\tilde{x}_\alpha - \tilde{\mu}^*)\Big\} \\
&= \sum_{\alpha=1}^{N}\text{trace}\Big\{(\tilde{x}_\alpha - \tilde{\mu}^*)^T(\Sigma^*)^{-1}(\tilde{x}_\alpha - \tilde{\mu}^*)\Big\} \\
&= \sum_{\alpha=1}^{N}\text{trace}\Big\{(\Sigma^*)^{-1}(\tilde{x}_\alpha - \tilde{\mu}^*)^T(\tilde{x}_\alpha - \tilde{\mu}^*)\Big\} \\
&= \text{trace}\Big\{\sum_{\alpha=1}^{N}(\Sigma^*)^{-1}(\tilde{x}_\alpha - \tilde{\mu}^*)^T(\tilde{x}_\alpha - \tilde{\mu}^*)\Big\} \\
&= \text{trace}\Big\{\Sigma^{-1}[\mathbf{A} + N(\bar{x} - \tilde{\mu})(\bar{x} - \tilde{\mu})^T]\Big\} \\
&= \text{trace}(\Sigma^{-1}\mathbf{A}) + \text{trace}\Big\{N(\bar{x} - \tilde{\mu})^T\Sigma^{-1}(\bar{x} - \tilde{\mu})\Big\} \\
&= \text{trace}(\Sigma^{-1}\mathbf{A}) + N(\bar{x} - \tilde{\mu})^T\Sigma^{-1}(\bar{x} - \tilde{\mu})
\end{aligned} \tag{2.6}$$

Using the simplification in 2.6, we can re-write the log likelihood function as:

$$\ln \mathcal{L} = -\frac{pn}{2}\ln 2\pi - \frac{n}{2}\ln\det(\Sigma) - \frac{1}{2}\text{tr}(\Sigma^{-1}\mathbf{A}) - \frac{N}{2}(\bar{x} - \tilde{\mu})^T\Sigma^{-1}(\bar{x} - \tilde{\mu}) \tag{2.7}$$

Note that maximizing the function $\ln \mathcal{L}$ is same as that of minimizing the function $-\ln \mathcal{L}$. Now, it is clear that minimizing $-\ln \mathcal{L}$ with respect to $\tilde{\mu}$ is the same as minimizing the term $(\bar{x} - \tilde{\mu})^T\Sigma^{-1}(\bar{x} - \tilde{\mu})$. Since, Σ^{-1} is a positive definite matrix, we know that the expression $-\ln \mathcal{L}$ will be minimized at $\tilde{\mu} = \bar{x}$. This proves the claim that the sample mean vector defined earlier is actually the maximum likelihood estimator for mean $\tilde{\mu}$.

Now the next step will be to find a positive definite matrix Σ such that $-\ln \mathcal{L}$ is minimized.

$$\begin{aligned} \ln \mathcal{L} &= -\frac{pn}{2}\ln 2\pi - \frac{n}{2}\ln\det(\Sigma) - \frac{1}{2}\mathrm{tr}(\Sigma^{-1}\mathbf{A}) - \frac{N}{2}(\tilde{x}-\tilde{\mu})^T\Sigma^{-1}(\tilde{x}-\tilde{\mu}) \\ &\leqslant -\frac{pn}{2}\ln 2\pi - \frac{n}{2}\ln\det(\Sigma) - \frac{1}{2}\mathrm{tr}(\Sigma^{-1}\mathbf{A}) \end{aligned} \tag{2.8}$$

From 2.8, it is clear that our job now is to maximize $\frac{n}{2}\ln\det(\Sigma) + \frac{1}{2}\mathrm{tr}(\Sigma^{-1}\mathbf{A})$. This maximization problem is solved through a lemma given below:

Lemma 2.2. *Let* $\mathbf{D}$ *be a positive definite matrix of size* $p \times p$*. Then the maximum of* $f(G) = N\ln\det(G) + \mathrm{trace}(\mathbf{G}^{-1}\mathbf{D})$ *with respect to the positive definite matrices* $\mathbf{G}$ *exists, and occurs at* $\mathbf{G} = \frac{1}{N}\mathbf{D}$ *and has value* $f(\frac{1}{N}\mathbf{D}) = pN\ln N - N\ln\{\det\mathbf{D}\} - pN$*.*

Proof. Since $\mathbf{D}$ is a positive definite matrix, there exists invertible lower traingular matrix $\mathbf{E}$ such that $\mathbf{D} = \mathbf{E}\mathbf{E}^T$. Define the matrix $\mathbf{H}$ as $\mathbf{H} = \mathbf{E}^T\mathbf{G}^{-1}\mathbf{E}$. Then, it is easy to see that $\mathbf{G} = \mathbf{E}\mathbf{H}^{-1}\mathbf{E}^T$.

$$\mathbf{G} = \mathbf{E}\mathbf{H}^{-1}\mathbf{E}^T \implies \det(\mathbf{G}) = \frac{\det(\mathbf{D})}{\det(\mathbf{H})}$$

Now, the following simplification can be made:

$$\mathrm{trace}(\mathbf{G}^{-1}\mathbf{D}) = \mathrm{trace}(\mathbf{G}^{-1}\mathbf{E}\mathbf{E}^T) = \mathrm{trace}(\mathbf{E}^T\mathbf{G}^{-1}\mathbf{E}) = \mathrm{trace}(\mathbf{H}) \tag{2.9}$$

The function we wish to maximize is a function of the positive definite matrix $\mathbf{G}$. However, using (2.9) in $f(\mathbf{G})$, we can rewrite it as a function of $\mathbf{H}$:

$$f(\mathbf{G}) = \tilde{f}(\mathbf{H}) = N\ln(\det\mathbf{D}) - N\ln(\det\mathbf{H}) - \mathrm{trace}(\mathbf{H}) \tag{2.10}$$

So we changed the problem of maximizing w.r.t $\mathbf{G}$ into a problem of maximizing w.r.t. $\mathbf{H}$. $\mathbf{G}$ is a positive definite matrix and therefore $\mathbf{G}^{-1}$ is positive definite as well. Hence, $(\mathbf{E}\tilde{x})^T\mathbf{G}^{-1}(\mathbf{E}\tilde{x}) > 0$ unless $\mathbf{E}\tilde{x} = 0$. Invertibility of $\mathbf{E}$ implies that $\tilde{x}^T\mathbf{H}\tilde{x} = 0$ iff $\tilde{x} = \tilde{0}$. This proves that $\mathbf{H}$ is a positive definite matrix. Hence, (2.10) has to be maximized with respect to the positive definite matrix $\mathbf{H}$.

$\mathbf{H}$ is a positive definite matrix $\implies$ $\mathbf{H} = \mathbf{T}\mathbf{T}^T$ for some lower triangular matrix $\mathbf{T}$

Therefore,

$$\begin{aligned} \tilde{f}(\mathbf{H}) &= -N\ln(\det\mathbf{D}) + N\ln(\det\mathbf{H}) - \mathrm{trace}(\mathbf{H}) \\ &= -N\ln(\det\mathbf{D}) + N\ln\{(\det\mathbf{H})^2\} - \mathrm{trace}(\mathbf{T}\mathbf{T}^T) \end{aligned} \tag{2.11}$$

Let the elements of the lower triangle matrix $\mathbf{T}$ be denoted by t_{ij}. Since, $\mathbf{T}$ is lower triangular $t_{ij} = 0$ for all $j > i$. Using this notation, $\tilde{f}(\mathbf{H})$ can be re-written as follows:

$$\tilde{f}(\mathbf{H}) = -N\ln(\det\mathbf{D}) + N\sum_{i=1}^{p}\ln(t_{ii})^2 - \sum_{i=1}^{p}t_{ii}^2 - \sum_{i>j}t_{ij}^2 \tag{2.12}$$

Then, the maximum of $\tilde{f}$ occurs at $t_{ii}^2 = N$ and $t_{ij} = 0$ for all $i \neq j$. In other words, this happens when $\mathbf{H} = N\mathcal{I}$. This implies that $\mathbf{G} = \frac{1}{N}\mathbf{D}$. □

Therefore, we have proved the Theorem 2.1.

Theorem 2.2. *The theorem will consist of two parts. The first part of the theorem given below implies the second result which is the result of our interest.*

1. $f : \mathcal{S} \longrightarrow \mathbb{R}$ *is a function while* $\phi : \mathcal{S} \longrightarrow \mathcal{S}^*$ *is a bijective function. Define a function* $g : \mathcal{S}^* \longrightarrow \mathbb{R}$ *as follows:*

   $$\text{for } \theta^* \in \mathcal{S}^*, g(\theta^*) = f[\phi^{-1}(\theta^*)]$$

   Then,

   - *If* $f(\theta)$ *attains a maximum at* $\theta = \theta_0$, *then* $g(\theta^*)$ *attains a maximum at* $\theta^* = \phi(\theta_0)$.
   - *If* $f(\theta)$ *attains a* ***unique*** *maximum at* $\theta = \theta_0$, *then* $g(\theta^*)$ *attains a* ***unique*** *maximum at* $\theta^* = \phi(\theta_0)$.

2. *Let* $\theta_1, \theta_2, \cdots, \theta_m$ *be the parameters associated with a distribution. Based on a given sample, the maximum likelihood estimators ()MLEs were calculated. Let these estimators be denoted by* $\hat{\theta}_1, \hat{\theta}_2, \cdots, \hat{\theta}_m$. *Then* $\psi_1(\hat{\theta}_1,\ \hat{\theta}_2\ ,\ \cdots,\ \hat{\theta}_m)$, $\psi_2(\hat{\theta}_1,\ \hat{\theta}_2,\ \cdots,\ \hat{\theta}_m)$, $\cdots$, $\psi_m(\hat{\theta}_1,\ \hat{\theta}_2,\ \cdots,\ \hat{\theta}_m)$ *are the MLEs of* $\psi_1(\theta_1, \theta_2, \cdots, \theta_m)$, $\psi_2(\theta_1, \theta_2, \cdots, \theta_m)$, $\cdots$, $\psi_m(\theta_1, \theta_2, \cdots, \theta_m)$ *if the transformations from* $\theta_i \longrightarrow \psi_i(\theta_i)$ *are all one-one.*

Theorem 2.3. *We just prove the first part of the theorem. The second part of the theorem can be proved easily using the first result.*

Proof. It is given that $f(\theta_0) \geqslant f(\theta)$ for all $\theta \in \mathcal{S}$. Let $\theta^* \in \mathcal{S}^*$, then $g(\theta^*) = f(\theta_1)$ for some $\theta_1 \in \mathcal{S}$. Therefore,

$$g(\theta^*) = f[\phi^{-1}(\theta^*)] = f(\theta_1) \leqslant f(\theta_0) = g[\phi(\theta_0)] \text{ for all } \theta^* \in \mathcal{S}^*$$

Now assume that f attains a unique maximum at θ_0. Then,

$$g(\theta^*) = f[\phi^{-1}(\theta^*)] = f(\theta_1) < f(\theta_0) = g[\phi(\theta_0)]$$

Therefore, its a unique maximum. □

2.2 Distribution of the sample mean vector and covariance matrix

The theorem we proved in the previous section has been restated below for the sake of convinience.

Theorem 2.4. *If $\tilde{x}_1$, $\tilde{x}_2 \cdots$, $\tilde{x}_N$ are $p \times 1$ vectors sampled from a normal distribution with mean $\tilde{\mu}$ and covariance Σ such that $p < N$. Then the maximum likelihood estimator of $\tilde{\mu}$ and Σ, denoted by $\widehat{\mu}$ and $\widehat{\Sigma}$, are:*

$$\widehat{\mu} = \bar{x} = \frac{1}{N}\sum_{\alpha=1}^{N} \tilde{x}_\alpha \text{ and } \widehat{\Sigma} = \frac{\mathbf{A}}{N} = \frac{1}{N}\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \bar{x})(\tilde{x}_\alpha - \bar{x})^T \tag{2.13}$$

This section will consist of two parts. In the first part, we show that the sample mean is normally distributed with mean $\tilde{\mu}$ and covariance $\frac{1}{N}\Sigma$. The covariance matrix is then proved to follow a distribution called the "Wishart distribution".

2.2.1 Distribution of the sample mean

The theorem we wish to prove is given below. This theorem will be proved in parts for the sake of convenience.

Theorem 2.5. *Let $\tilde{X}_1, \tilde{X}_2, \cdots, \tilde{X}_N$ be sampled from $\mathcal{N}(\tilde{\mu}, \Sigma)$. Then, the sample mean $\bar{X}$ is distributed according $\mathcal{N}(\tilde{\mu}, \frac{1}{N}\Sigma)$ and independently of $\widehat{\Sigma}$, the MLE of Σ. In addition to this, $N\widehat{\Sigma}$ is distributed as $\sum_{\alpha=1}^{N-1} \tilde{Z}_\alpha \tilde{Z}_\alpha^T$ where $\tilde{Z}_\alpha \sim \mathcal{N}(\tilde{0}, \Sigma)$ for all $\alpha = 1, 2, \cdots, N-1$*

The theorem 2.5 follows very easily from the two lemmas that will be proved in this subsection.

Lemma 2.3. *$\tilde{X}_1, \tilde{X}_2, \cdots, \tilde{X}_N$ are independent random variables such that $\tilde{X}_i \sim \mathcal{N}(\tilde{\mu}_i, \Sigma)$. Let $\mathbf{C} = (c_{ij})$ be an $N \times N$ orthogonal matrix. Then, $\tilde{Y}_i = \sum_{j=1}^{N} c_{ij}\tilde{X}_j \sim \mathcal{N}(\tilde{\nu}_i, \Sigma)$ where $\tilde{\nu}_i = \sum_{j=1}^{N} c_{ij}\tilde{\mu}_j$ for all $i \in \{1, 2, \cdots, N\}$. In addition to this, $\tilde{Y}_1, \tilde{Y}_2, \cdots, \tilde{Y}_N$ are mutually independent random vectors.*

Proof. First of all, observe that any linear combination of $\tilde{Y}_i$'s is nothing but a linear combination of $\tilde{X}_i$'s. Since each $\tilde{X}_i$ is normally distributed, it is true that every linear combination of $\tilde{Y}_i$'s is also normally distributed. This implies that the vector $\tilde{Y}_1, \tilde{Y}_2, \cdots, \tilde{Y}_N$ is also normally distributed. Then,

$$\mathbb{E}[\tilde{Y}_i] = \sum_{j=1}^{N} c_{ij}\mathbb{E}[\tilde{X}_j] = \sum_{j=1}^{N} c_{ij}\tilde{\mu}_j = \tilde{\nu}_i \text{ for all } i \in \{1, 2, \cdots, N\}$$

Now we are left to show the mutual independence and variance of $\tilde{Y}_i$ for each possible i. This is proved by showing that $cov(\tilde{Y}_i, \tilde{Y}_j) = 0$ for all $i \neq j$.

$$cov(\tilde{Y}_i, \tilde{Y}_j) = \mathbb{E}\Big[(\sum_{k=1}^{N} c_{ik}\tilde{X}_k - \sum_{k=1}^{N} c_{ik}\tilde{\mu}_k)(\sum_{t=1}^{N} c_{jt}\tilde{X}_t - \sum_{t=1}^{N} c_{jt}\tilde{\mu}_t)^T\Big] = \sum_{k=1}^{N}\sum_{t=1}^{N} c_{ik}c_{jt}cov(\tilde{X}_k, \tilde{X}_t) \tag{2.14}$$

Since $cov(\tilde{X}_i, \tilde{X}_j) = 0$ for $i \neq j$ due to the independence condition, we can rewrite (2.14) as follows:

$$cov(\tilde{Y}_i, \tilde{Y}_j) = \sum_{k=1}^{N} \sum_{t=1}^{N} c_{ik} c_{jt} \delta = \begin{cases} 0 \text{ if } k \neq t \\ \Sigma \text{ when } k = t \end{cases}$$

We can show that $cov(\tilde{Y}_i, \tilde{Y}_j) = \delta_{ij}\Sigma$ as $\mathbf{C}$ is an orthogonal matrix. This proves that the $\tilde{Y}_i$'s are mutually independent and that the variance of $\bar{X}$ is Σ. □

Lemma 2.4. *Let* $\mathbf{C}$ *be an orthogonal matrix whose* $(i, j)^{th}$ *entry is denoted by* c_{ij}. *Then the following holds true:*

$$\sum_{\alpha=1}^{N} \tilde{x}_\alpha \tilde{x}_\alpha^T = \sum_{\alpha=1}^{N} \tilde{y}_\alpha \tilde{y}_\alpha^T \textit{ where } \tilde{y}_\alpha = \sum_{\beta=1}^{N} c_{\alpha\beta} \tilde{x}_\beta$$

Proof.

$$\begin{aligned}
\sum_{\alpha=1}^{N} \tilde{y}_\alpha \tilde{y}_\alpha^T &= \sum_{\alpha=1}^{N} \left\{ \Big(\sum_{\beta=1}^{N} c_{\alpha\beta} \tilde{x}_\beta \Big) \Big(\sum_{\beta=1}^{N} c_{\alpha\beta} \tilde{x}_\beta^T \Big) \right\} \\
&= \sum_{\alpha=1}^{N} \left\{ \sum_{k=1}^{N} \sum_{t=1}^{N} c_{\alpha k} c_{\alpha t} \tilde{x}_k \tilde{x}_t^T \right\} \\
&= \sum_{k=1}^{N} \sum_{t=1}^{N} \Big(\sum_{\alpha=1}^{N} c_{\alpha k} c_{\alpha t} \Big) \tilde{x}_k \tilde{x}_t^T \text{ (since summation is finite)} \\
&= \sum_{k=1}^{N} \sum_{t=1}^{N} \delta_{kt} \tilde{x}_k \tilde{x}_t^T \\
&= \sum_{\alpha=1}^{N} \tilde{x}_\alpha \tilde{x}_\alpha^T
\end{aligned}$$

□

Lemma 2.5. *Let* $\tilde{X}_1, \tilde{X}_2, \cdots, \tilde{X}_N$ *be sampled from* $\mathcal{N}(\tilde{\mu}, \Sigma)$. *Then the sample mean* $\widehat{\mu}$ *and* $\widehat{\Sigma}$ *are independently distributed.*

Proof. Consider the $N \times 1$ vector $(\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \cdots, \frac{1}{\sqrt{N}})^T \in \mathbb{R}^N$. Since every linearly independent set can be extended to form a basis set, we can add extra vectors $\tilde{b}_1, \cdots, \tilde{b}_{N-1}$ to $(\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \cdots, \frac{1}{\sqrt{N}})^T$ such that all these N vectors together form an orthonormal basis for $\mathbb{R}^N$. Let these N orthonormal basis vectors, as row vectors, be clubbed together to give the orthogonal matrix $\mathbf{B}$, with its $(i, j)^{th}$ element denoted by d_{ij}. Define a vector $\tilde{Z}_\alpha$ as follows:

$$\tilde{Z}_\alpha = \sum_{\beta=1}^{N} b_{\alpha\beta} \tilde{X}_\beta \ \forall \alpha \in \{1, 2, \cdots, N\}$$

From the construction of the matrix $\mathbf{B}$, $\tilde{Z}_N = \sqrt{N}\bar{X}$. As defined earlier, let $\mathbf{A} = \sum_{\alpha=1}^{N} \tilde{X}_\alpha \tilde{X}_\alpha^T - N\bar{X}\bar{X}^T$. Using Lemma 2.4, we know that:

$$\mathbf{A} = \sum_{\alpha=1}^{N} \tilde{X}_\alpha \tilde{X}_\alpha^T - N\bar{X}\bar{X}^T = \sum_{\alpha=1}^{N} \tilde{Z}_\alpha \tilde{Z}_\alpha^T - \tilde{Z}_N \tilde{Z}_N^T = \sum_{\alpha=1}^{N-1} \tilde{Z}_\alpha \tilde{Z}_\alpha^T$$

Now, $(\tilde{Z}_1, \cdots, \tilde{Z}_{n-1})$ are independent random vectors. This implies:

$$\tilde{Z}_n \perp (\tilde{Z}_1, \tilde{Z}_2, \cdots, \tilde{Z}_{n-1})$$

We will now use the fact that if $\tilde{X}$ and $\tilde{Y}$ are independent random vectors, then the random vectors $f(\tilde{X})$ and $g(\tilde{Y})$ are also independent for borel measurable functions $f, g : \mathbb{R}^{n-1} \longrightarrow \mathbb{R}^{p^2}$. Taking $f(\tilde{Z}) = \frac{1}{\sqrt{N}}\tilde{Z}$ and $g(\tilde{Z}_1, \tilde{Z}_2, \cdots, \tilde{Z}_{n-1}) = \frac{1}{\sqrt{N}} \sum_{\alpha=1}^{N-1} \tilde{Z}_\alpha \tilde{Z}_\alpha^T$, we can prove that $\bar{X} \perp \widehat{\mu}$ □

Lemma 2.6. *Let $\bar{X}$ be the sample mean vector of a sample of N observations drawn from $\mathcal{N}(\tilde{\mu}, \Sigma)$. Then $\bar{X}$ has mean $\tilde{\mu}$ variance $\frac{1}{N}\Sigma$*

Proof. The vectors $\tilde{Z}_i$'s in this proof are as defined in the earlier theorem. The, we know that $\tilde{Z}_n = \sum_{\beta=1}^{N} \frac{1}{\sqrt{N}}\tilde{X}_\beta$. Then, clearly $\mathbb{E}[\tilde{Z}_N] = \sqrt{N}\tilde{\mu}$. Additionally, since $\tilde{Z}_N = \sum_{\beta=1}^{N} \frac{1}{\sqrt{N}}\tilde{X}_\beta$, we can use Lemma 2.4, to show that $\mathcal{V}[\tilde{Z}_N] = \Sigma$. Therefore,

$$\tilde{Z}_N \sim \mathcal{N}(\sqrt{N}\tilde{\mu}, \Sigma) \tag{2.15}$$

We showed in the Lemma 2.4 that $\bar{X} = \frac{1}{\sqrt{N}\tilde{Z}_N}$. From (2.15), it follows that $\mathbb{E}[\bar{X}] = \tilde{\mu}$.

$$\mathcal{V}[\bar{X}] = \mathbb{E}\Big[\Big(\frac{\tilde{Z}_N}{\sqrt{N}} - \tilde{\mu}\Big)\Big(\frac{\tilde{Z}_N}{\sqrt{N}} - \tilde{\mu}\Big)^T\Big] = \frac{1}{N}\mathbb{E}[\tilde{Z}_N \tilde{Z}_N^T] - \tilde{\mu}\tilde{\mu}^T \tag{2.16}$$

Now, using $\mathcal{V}[\tilde{Z}_N] = \mathbb{E}[\tilde{Z}_N \tilde{Z}_N^T] - N\tilde{\mu}\tilde{\mu}^T$, we can show that:

$$\mathbb{E}[\tilde{Z}_N \tilde{Z}_N^T] = \Sigma + N\tilde{\mu}\tilde{\mu}^T \tag{2.17}$$

From (2.16) and (2.17) it can be shown that $\mathcal{V}[\bar{X}] = \frac{1}{N}\Sigma$. Hence $\bar{X} \sim \mathcal{N}(\tilde{\mu}, \frac{1}{N}\Sigma)$ □

Remark 2.1. A few useful observatios/results have been noted below:

1. $\bar{X}$ is an unbiased estimator of $\tilde{\mu}$.

2. $\widehat{\Sigma}$ is not an unbiased estimator for Σ. In fact, it can be shown that $\mathbb{E}[\widehat{\Sigma}] = \frac{N-1}{N}\Sigma$. Therefore, $\frac{N}{N-1}\widehat{\Sigma}$denoted by $\mathcal{S}$ is an unbiased estimator for Σ. From now onwards, $\mathcal{S}$ is called as the sample covariance matrix.

3. Note that $\frac{\mathbf{A}}{N}$ is the MLE for Σ while $\frac{N}{N-1}\mathbf{A}$ is the unbiased estimator of Σ.

2.2.2 Distribution of the covariance matrix

We earlier showed that the sample covariance matrix $\mathcal{S} = \frac{1}{N}\sum_{\alpha=1}^{N}(\tilde{x}_\alpha - \bar{x})(\tilde{x}_\alpha - \bar{x})^T$ is an unbiased estimator for Σ. The distribution of $\mathbf{A} = (\tilde{x}_\alpha - \bar{x})(\tilde{x}_\alpha - \bar{x})^T$ is often called the **Wishart distribution**.

Before moving ahead with the proof, it is important to recall the Gram-Schmidt process.

Gram-Schmidt process

Let $\{\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_p\}$ be a given basis $\mathcal{W}$ of $\mathbb{R}^n$. Define a new set of vectors $\tilde{v}_1, \tilde{v}_2, \cdots, \tilde{v}_p$ as follows:

$$\begin{aligned}
\tilde{v}_1 &= \tilde{x}_1 \\
\tilde{v}_2 &= \tilde{x}_2 - \frac{\langle \tilde{x}_2, \tilde{v}_1\rangle}{\sqrt{\langle \tilde{v}_1, \tilde{v}_1\rangle}}\tilde{v}_1 \\
\tilde{v}_3 &= \tilde{x}_3 - \frac{\langle \tilde{x}_3, \tilde{v}_1\rangle}{\sqrt{\langle \tilde{v}_1, \tilde{v}_1\rangle}}\tilde{v}_1 - \frac{\langle \tilde{x}_3, \tilde{v}_2\rangle}{\sqrt{\langle \tilde{v}_2, \tilde{v}_2\rangle}}\tilde{v}_2 \\
&\vdots \\
\tilde{v}_p &= \tilde{x}_p - \frac{\langle \tilde{x}_p, \tilde{v}_1\rangle}{\sqrt{\langle \tilde{v}_1, \tilde{v}_1\rangle}}\tilde{v}_1 - \cdots \frac{\langle \tilde{x}_p, \tilde{v}_{p-1}\rangle}{\sqrt{\langle \tilde{v}_{p-1}, \tilde{v}_{p-1}\rangle}}\tilde{v}_{p-1}
\end{aligned}$$

Then $\{\tilde{v}_1, \tilde{v}_2, \cdots, \tilde{v}_p\}$ form an orthogonal basis for $\mathcal{W}$.

Theorem 2.6. *Let $\tilde{X}_1, \tilde{X}_2, \cdots, \tilde{X}_N$ be $p \times 1$ independent random vectors following $\mathcal{N}(\tilde{\mu}, \Sigma)$. The, the density of $\mathbf{A}$ for $\mathbf{A}$ positive definite is given by:*

$$\frac{(\det\mathbf{A})^{\frac{1}{2}(n-p-1)}\exp\Big\{-\frac{1}{2}\mathrm{tr}(\Sigma^{-1}\mathbf{A})\Big\}}{2^{\frac{np}{2}}\pi^{\frac{p(p-1)}{4}}(\det\Sigma)^{\frac{n}{2}}\prod_{i=1}^{p}\Gamma[\frac{1}{2}(n+1-i)]} \tag{2.18}$$

Proof. Let us first start with the special case $\Sigma = \mathcal{I}_p$. Consider the matrix $\mathbf{Z}$ to be the matrix formed by clubbing together the vectors $\tilde{Z}_i$'s ($i \in \{1, 2, \cdots, N\}$) that we defined earlier in Lemma (2.5).

$$\mathbf{Z} = (\tilde{Z}_1, \tilde{Z}_2, \cdots, \tilde{Z}_N) = \begin{bmatrix} \tilde{v}_1^T \\ \tilde{v}_2^T \\ \vdots \\ \tilde{v}_p^T \end{bmatrix} \quad \text{i.e. } \tilde{v}_i^T \text{ denotes } i^{th} \text{ row of } \mathbf{Z}. \tag{2.19}$$

Since, $\tilde{X}_1, \cdots, \tilde{X}_N$ are independent vectors, it is clear by the definition of $\tilde{Z}_\alpha$ that $\mathcal{V}[\tilde{Z}_\alpha] = \Sigma$ for all $\alpha \in \{1, 2, \cdots, N\}$. The expected value of $\tilde{Z}_\alpha$ can be found out as follows:

$$\begin{aligned}\mathbb{E}[\tilde{Z}_\alpha] &= \sum_{\beta=1}^{N} b_{\alpha\beta}\mathbb{E}[\tilde{X}_\beta] \\ &= \sum_{\beta=1}^{N} b_{\alpha\beta}\tilde{\mu} \\ &= \sum_{\beta=1}^{N} b_{\alpha\beta}b_{N\beta}\sqrt{N}\tilde{\mu} \text{ where } b_{N\beta} = \frac{1}{\sqrt{N}}\ \forall\beta \in \{1,\cdots,N\} \\ &= 0\end{aligned} \tag{2.20}$$

Therefore, $\tilde{Z}_\alpha \sim \mathcal{N}(\tilde{0},\Sigma)$ for all α. Since we are in the special case 0f $\Sigma = \mathcal{I}_p$, we know that each component of $\tilde{Z}_\alpha$, which we denote by $z_i^{(\alpha)}$ is a $\mathcal{N}(0,1)$ random variable. This means that each component of $\tilde{v}_\alpha^T$, which we will denote by $v_i^{(\alpha)}$, is also normally distributed. $\Sigma = \mathcal{I}_p$ also implies that the components of vectors $\tilde{v}_\alpha^T$ are all independent. Hence each $\tilde{v}_\alpha^T \sim \mathcal{N}(\tilde{0},\mathcal{I}_N)$.

Now, we use the Gram-Schmidt process that we earlier recalled. We apply the Gram-Schmidt process on vectors $\{\tilde{v}_1,\cdots,\tilde{v}_p\}$ to produce an orthogonal basis set $\mathcal{W} = \{\tilde{w}_1,\cdots,\tilde{w}_p\}$ such that:

$$\tilde{w}_i = \tilde{v}_i - \sum_{j=1}^{i-1}\frac{\tilde{w}_j^T\tilde{v}_i}{\sqrt{\tilde{w}_j^T\tilde{w}_j}}\tilde{w}_j \text{ such that } \langle\tilde{w}_i,\tilde{w}_j\rangle = 0\ \forall i \neq j \tag{2.21}$$

Define t_{ii} and t_{ij} as follows:

$$t_{ii} = \langle\tilde{w}_i,\tilde{w}_i\rangle^{\frac{1}{2}}\ \forall i \in \{1,\cdots,p\} \text{ and } t_{ij} = \frac{\tilde{v}_i^T\tilde{w}_j}{\sqrt{\langle\tilde{w}_i,\tilde{w}_i\rangle}} \text{ for } (i,j) \in \{1,\cdots,i-1\}\times\{2,\cdots,p\} \tag{2.22}$$

Using the notation in (2.22) in equation (2.21), we can show that:

$$\tilde{v}_i = \sum_{j=1}^{i}\frac{t_{ij}}{\langle\tilde{w}_j,\tilde{w}_j\rangle^{\frac{1}{2}}}\tilde{w}_j \tag{2.23}$$

Since $\mathbf{A}$ and $\sum_{i=1}^{N-1}\tilde{Z}_i\tilde{Z}_i^T$, where $\tilde{Z}_i \sim \mathcal{N}(\tilde{0},\Sigma)$, have the same distributions it can be easily shown that the $(h,i)^{th}$ element of $\mathbf{A}$, denoted by a_{hi}, is given by

$$a_{hi} = \sum_{\alpha=1}^{N-1} z_h^{(\alpha)}z_i^{(\alpha)} = \sum_{\alpha=1}^{N-1} v_\alpha^{(h)}v_\alpha^{(i)} = \tilde{v}_h^T\tilde{v}_i$$

(2.23), along with the fact that $\langle \tilde{w}_i, \tilde{w}_j \rangle = 0$ for $i \neq j$ can be used to simplify a_{hi} further.

$$a_{hi} = \tilde{v}_h^T \tilde{v}_i = \left[\sum_{j=1}^{h} \frac{t_{hj}}{\langle \tilde{w}_j, \tilde{w}_j \rangle^{\frac{1}{2}}} \tilde{w}_j \right]^T \left[\sum_{j=1}^{i} \frac{t_{ij}}{\langle \tilde{w}_j, \tilde{w}_j \rangle^{\frac{1}{2}}} \tilde{w}_j \right] = \sum_{j=1}^{min\{h,i\}} t_{hj} t_{ij} \tag{2.24}$$

Now, define a new matrix $\mathbf{T}$ whose entries are t_{ij} that we defined earlier. Since, we earlier defined t_{ij} only for $j < i$, we set $t_{ij} = 0$ when $i < j$. Therefore, $\mathbf{T}$ is a lower triangular matrix. From (2.24), it is clear that $\mathbf{A} = \mathbf{T}\mathbf{T}^T$.

$$\text{Let } \tilde{v}_i = \sum_{j=1}^{i-1} \gamma_j \tilde{w}_j \text{ then, } \gamma_j = t_{ij} \text{ for all } j \in \{1, 2, \cdots, i-1\} \tag{2.25}$$

The magnitude of the vector $\tilde{v}_i$ will be the same irrespective of the choice of basis. Then,

$$\langle \tilde{v}_i, \tilde{v}_i \rangle = \sum_{j=1}^{i} t_{ij}^2$$

This implies that the sum of the remaining $n - (i-1)$ co-ordinate coefficients of $\tilde{v}_i$ is $\langle \tilde{v}_i, \tilde{v}_j \rangle - \sum_{j=1}^{i-1} t_{ij}^2 = t_{ii}^2 = \langle \tilde{w}_i, \tilde{w}_j \rangle$.

Claim: $t_{ii}^2 \sim \chi^2_{n-i+1}$ and $t_{ij} \sim \mathcal{N}(0,1)$ for $i > j$
$t_{ii}^2 = \tilde{w}_i^T \tilde{w}_i = [V_i^{(i)}]^2 + \cdots + [V_n^{(i)}]^2$. Since each component of $\tilde{V}_i^T = (V_1^{(i)}, V_2^{(i)}, \cdots, V_n^{(i)})$ is a $\mathcal{N}(0,1)$ random variable. This clearly implies that t_{ii}^2 follows the chi-squared distribution with $n-(i-1)$ degrees of freedom. Using Lemma 2.3, it can be proved that $t_{ij} \sim \mathcal{N}(0,1)$ for $i > j$ and that the random variables $t_{11}, t_{22}, \cdots, t_{pp}$ is independently distributed.

Since $t_{11}, t_{22}, \cdots, t_{pp}$ are all independently distributed, it can be shown that the joint density of all t_{ij} for $(i,j) \in \{1, \cdots, i\} \times \{1, 2, \cdots, p\}$ is:

$$f(t_{11}, t_{12}, \cdots, t_{pp}) = \prod_{i>j} \Big[\prod_{j=1}^{p} n(t_{ij} | \ \exp\{-\frac{t_{ij}^2}{2}\}) \times \prod_{i=1}^{p} \chi^2_{n-i+1}(t_{ii}) \Big] \tag{2.26}$$

The joint density in (2.26), can be simplified to give

$$f(t_{11}, \cdots, t_{pp}) = \exp\Big\{ -\frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{i} t_{ij}^2 \Big\} \times \Big\{ \prod_{i=1}^{p} \frac{t_{ii}^{n-i}}{\Gamma(\frac{n-i+1}{2})} \Big\} \times \frac{1}{\pi^{\frac{p(p-1)}{4}}} \times \frac{1}{2^{\frac{p(n-2)}{2}}} \tag{2.27}$$

Σ is a symmetric positive definite matrix, then by cholesky decomposition of Σ, there exists a lower triangular matrix $\mathbf{C}$ such that $\Sigma = \mathbf{C}\mathbf{C}^T$. Now, consider the linear transformation $\mathbf{T}^* = \mathbf{C}\mathbf{T}$. Then,

$$t_{ij}^* = \begin{cases} \sum_{k=1}^{i} c_{ik} t_{kj} \text{ for } i \geqslant j \\ 0 \text{ for } i < j \end{cases}$$

This, in matrix form, wil be written as:

$$\begin{bmatrix} t^*_{11} \\ t^*_{21} \\ t^*_{22} \\ \vdots \\ t^*_{p1} \\ \vdots \\ t^*_{pp} \end{bmatrix} = \begin{bmatrix} c_{11} & 0 & 0 & \cdots & 0 & \cdots & 0 \\ * & c_{22} & 0 & \cdots & 0 & \cdots & 0 \\ * & * & c_{22} & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ * & * & * & \cdots & c_{pp} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ * & * & * & \cdots & * & \cdots & c_{pp} \end{bmatrix} \begin{bmatrix} t_{11} \\ t_{21} \\ t_{22} \\ \vdots \\ t_{p1} \\ \vdots \\ t_{pp} \end{bmatrix} \tag{2.28}$$

It is easy to check that the Jacobian of the transformation from $\mathbf{T}$ to $\mathbf{T}^*$ mentioned in (2.28) is $\frac{1}{\prod_{i=1}^{p} c_{ii}^{i}}$. Note that the matrix $\mathbf{T}$ and $\mathbf{T}^*$ are sometimes considered as column vectors as shown in (2.28) even though the notation used for the matrix and vector form is the same.

Now, the density of $\mathbf{T}^*$ can be found out by substituting $t_{ii} = \frac{t^*_{ii}}{c_{ii}}$ in (2.26). In addition to that, the term $\sum_{i=1}^{p}\sum_{j=1}^{i} t_{ij}^2$ in (2.27) can be simplified as:

$$\begin{aligned} \sum_{i=1}^{p}\sum_{j=1}^{i} t_{ij}^2 &= \text{trace}(\mathbf{T}\mathbf{T}^T) \\ &= \text{trace}[\mathbf{C}^{-1}\mathbf{T}^*\{\mathbf{T}^*\}^T\{\mathbf{C}^{-1}\}^T] \\ &= \text{trace}[\mathbf{T}^*\{\mathbf{T}^*\}^T\mathbf{C}^{-1}\{\mathbf{C}^{-1}\}^T] \\ &= \textbf{trace}[\{\mathbf{T}^*\}^T\Sigma^{-1}\mathbf{T}^*] \text{ since } \Sigma = \mathbf{C}\mathbf{C}^* \end{aligned}$$

The density of $\mathbf{T}^*$ can be calculated to be

$$\tilde{f}_{\mathbf{T}^*}(t^*_{11}, t^*_{21}, \cdots, t^*_{pp}) = \exp\Big\{-\frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{i} t_{ij}^2\Big\} \times \Big\{\prod_{i=1}^{p}\frac{\{t^*_{ii}\}^{n-1}}{\Gamma(\frac{n-i+1}{2})}\Big\} \times \frac{\det(\Sigma)^{-\frac{n}{2}}}{\pi^{\frac{p(p-1)}{4}}} \times \frac{1}{2^{\frac{p(n-2)}{2}}}$$

Using (2.24), it can be shown that $a_{hi} = \sum_{j=1}^{i} t^*_{hj}t^*_{ij}$ for $h \geqslant i$. Then

$$\frac{\partial a_{hi}}{\partial t^*_{kl}} = \begin{cases} 0 \text{ if } k > h \\ 0 \text{ if } k = h, l > i \end{cases} \qquad \frac{\partial a_{hh}}{\partial t^*_{hh}} = 2t^*_{hh}\frac{a_{hi}}{t^*_{hi}} = t^*_{ii} \text{ if } h > i \tag{2.29}$$

Using (2.29), it can be shown that the Jacobian of the transformation from $\mathbb{T}^*$ to $\mathbf{A}$ is $2^{-p} \times \frac{1}{\prod_{i=1}^{p}(t^*_{ii})^{p+1-i}}$.

USing this, our theorem of interest, i.e. Theorem 2.6, can be proved. In other words, we can show that the density of $\mathbf{A} = \sum_{\alpha=1}^{N-1}\tilde{Z}_\alpha\tilde{Z}_\alpha^T$, where each $\tilde{Z}_\alpha$ follows $\mathcal{N}(\tilde{0}, \Sigma)$, is as follows:

$$\frac{(\det\mathbf{A})^{\frac{1}{2}(n-p-1)}\exp\Big\{-\frac{1}{2}\text{tr}(\Sigma^{-1}\mathbf{A})\Big\}}{2^{\frac{np}{2}}\pi^{\frac{p(p-1)}{4}}(\det\Sigma)^{\frac{n}{2}}\prod_{i=1}^{p}\Gamma[\frac{1}{2}(n+1-i)]} \tag{2.30}$$

□

The following two corollaries follow easily from the theorem we just proved.

Corollary 2.1. *Let $\tilde{X}_1$, $\tilde{X}_2$, $\cdots$, $\tilde{X}_N$ be independently distributed according to $\mathcal{N}(\tilde{\mu}, \Sigma)$. Then the density of $\mathbf{A} = \sum_{\alpha=1}^{N-1}(\tilde{X}_\alpha - \bar{X})(\tilde{X}_\alpha - \bar{X})^T$ is given by* (2.30).

Remark 2.2. The distribution in (2.30) is called the **Wishart distribution**. It is denoted by $\mathcal{W}(\Sigma, N-1)$

Corollary 2.2. *Let $\tilde{X}_1$, $\tilde{X}_2$, $\cdots$, $\tilde{X}_N$ be independently distributed, each according to $\mathcal{N}(\tilde{\mu}, \Sigma)$. Then the sample covariance matrix $\mathcal{S}$ follows the distribution $\mathcal{W}(\frac{1}{N-1}\Sigma, N-1)$.*

Proof. Using the Corollary (2.1), we can show that $\mathcal{S}$ has the same distribution as that of $\sum_{\alpha=1}^{N-1}[\frac{1}{\sqrt{N-1}}\tilde{Z}_\alpha][\frac{1}{\sqrt{N-1}}\tilde{Z}_\alpha]^T$ where $\frac{1}{\sqrt{N-1}}\tilde{Z}_1$, $\cdots$, $\frac{1}{\sqrt{N-1}}\tilde{Z}_N$ are independently distributed according to $\mathcal{N}(\tilde{0}, \frac{1}{N-1}\Sigma)$. Then, (2.30) implies this corollary. □

Chapter 3

Principal component analysis

The following chapter is based on [Ize08], [Koc14], [Shl14], [Gho15] and [Bur10] and [Lee14]. The dataset has been taken from [DG17]

Principal component analysis(PCA) is a standard technique used in the data analysis of complex and confusing data sets in the hope of uncovering hidden patterns and structures in the data. PCA is a“ **dimension reduction techniques**” ,or in other words, a technique that reduces the number of variables involved in the experiment without the loss of “important information”. This has been explained further in the paragraph that follows.

Dimension reduction

Dimension reduction is mapping data to a lower dimensional space such that the “important information” regarding the data is retained or a subspace is identified such that the data points live in this subspace. Note that the word “**dimension**” indicates the number of variables in the data set. The variables involved may or may not be correlated. Additionally, these variables are also sometimes referred to as attributes or features. PCA is one of the many different dimension reduction techniques that have been proposed.

Any dimension reduction technique falls under one of the two following categories:

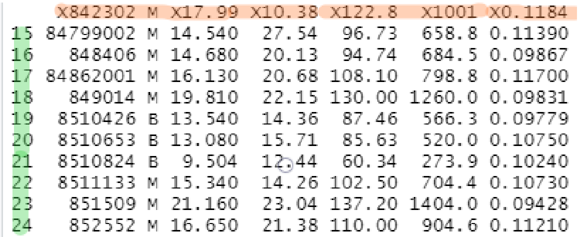
1. Methods that rely on **projections**

2. Methods that attempt to model the manifold on which the data lies.

The focus of this chapter will be Principal component analysis, which is a dimension reduction technique that relies on projections. Since dimension reduction aims to preserve “important information”, it is important to ask the question, “How is the term important defined?”. Important information for PCA is the variation present in the data. Details of the PCA procedure, which will be mentioned later in this chapter, imply that PCA assumes that “useful” or interesting information is only present in the regions of high variance. In conclusion, PCA reduces the dimension of the data while retaining as much as possible about the variation present in the data.

How is the dimension reduced? - Brief Outline

Let us consider the Wisconsin Diagnostic Breast Cancer (WDBC) data to understand the purpose of PCA. WDBC is a data frame of size 569×32. In other words, the data frame consists of 569 observations taken on 32 variables. A few of the rows, which correspond to observations, and columns, corresponding to variables, for the WDBC dataset are given below.



```
    X842302 M X17.99 X10.38 X122.8  X1001 X0.1184
15 84799002 M 14.540  27.54  96.73  658.8 0.11390
16   848406 M 14.680  20.13  94.74  684.5 0.09867
17 84862001 M 16.130  20.68 108.10  798.8 0.11700
18   849014 M 19.810  22.15 130.00 1260.0 0.09831
19  8510426 B 13.540  14.36  87.46  566.3 0.09779
20  8510653 B 13.080  15.71  85.63  520.0 0.10750
21  8510824 B  9.504  12.44  60.34  273.9 0.10240
22  8511133 M 15.340  14.26 102.50  704.4 0.10730
23   851509 M 21.160  23.04 137.20 1404.0 0.09428
24   852552 M 16.650  21.38 110.00  904.6 0.11210
```

Figure 3.1: Wisconsin Diagnostic Breast Cancer (WDBC) data.

In the Figure 3.1, the numbers highlighted in green indicate the serial number while the characters highlighted in red are the variables. First column mentions the ID of a patient from whom each of the 31 variable values are being noted. The second variable is a binary variable. It can only take the values "B" and "M" which stand for benign and malignant respectively.

It is clear from the breast cancer data set that the number of variables involved is large and therefore complicates the calculations while using the existing data to predict a given variable of interest. Therefore, reducing the number of variables involved would be of great use in such instances. PCA, when applied to this data, utilizes the original 30 variables to construct a much smaller set of variables called "**principal component scores**" denoted by $PC_1, PC_2, \cdots, PC_i \cdots$. Often, the term principal component variables is used to describe these newly constructed set of variables. It will be shown later that only, in this example, 7 principal component variables are enough to retain "most of the important information" in the data. In other words, seven principal component variables explain at least 90% of the total variation present in this data. Thus, PCA allows us to convinently work with these 7 new (or principal component) variables instead of the 30 old variables.

Remark 3.1. Note that each of the new variables, denoted by PC_i for $i \in \{1, \cdots d\}$, must be **linear combinations of the original set of variables**. Although this appears to be like a strict restriction on the kind of new variables that we can define, it definitely gives the advantage of exploiting all the nice properties of linear combinations of random variables. In fact, this restriction allows us to use change the problem of finding the new set of variables into the problem of finding a new basis set to re-express the data.

Firstly, all the definitions and notations are mentioned **without motivation** in the section that follows. However, the meaning behind these definitions will be clear by the end of the chapter.

3.1 Notations

Let us consider a $d \times 1$ random vector $\tilde{X}$ which follows a distribution with mean $\tilde{\mu}$ and covariance matrix Σ. From now onwards, this information will be conveyed using the notation

$$\tilde{X} \sim (\tilde{\mu}, \Sigma)$$

Each of the p components of this vector correspond to a variable or feature that we can measure. Assume that the vector $\tilde{X}$ is a continuous real valued random variable with the spectral decomposition of the covariance matrix given by

$$\boxed{\Sigma = \Gamma\Lambda\Gamma^T} \tag{3.1}$$

Here, $\Gamma = [\tilde{\eta}_1, \cdots, \tilde{\eta}_d]$ is an orthogonal matrix whose columns consist of the eigen vectors of the matrix Σ. $\Lambda = \mathbf{diag}(\lambda_1, \lambda_2, \cdots, \lambda_d)$ is the diagonal matrix made up of the eigen values of Σ.

We will assume Σ to be a $d \times d$ full rank matrix with distinct eigen values unless otherwise stated. However, if Σ is a rank deficient matrix of rank $r < d$, then the decomposition of Σ is given by:

$$\boxed{\Sigma = \Gamma_r\Lambda_r\Gamma_r^T} \tag{3.2}$$

Note that Γ_r is the $d \times q$ matrix $[\ \tilde{\eta}_1\ \tilde{\eta}_2\ \cdots\ \tilde{\eta}_r\ ]$ while Λ_r is the square matrix $\mathbf{diag}(\lambda_1, \lambda_2, \cdots, \lambda_r)$ of size $r \times r$. Observe that the matrix Γ in (3.1) is an orthogonal matrix while the matrix Γ_r in (3.2) is a rectangular matrix which is **not** orthogonal. However, the matrix Γ_r is **r-orthogonal** i.e. it follows the following property

$$\Gamma_r^T\Gamma_r = \mathcal{I}_r \text{ and } \Gamma_r\Gamma_r^T \neq \mathcal{I}_r$$

In addition to that, the following properties of Γ_k are also worth mentioning.

$$\Gamma_r^T\Gamma = \mathcal{I}_{k\times d} \text{ and } \Gamma^T\Gamma_k = \mathcal{I}_{d\times k} \tag{3.3}$$

Definitions of certain terms are introduced without mentioning the motivation behind the definitions. However, those motivations will become clear by the time the chapter is completed.

3.2 Population and Sample Principal Components

Population principal components

For all the definitions given below, the rank of Σ is assumed to be $r \leqslant d$. Let $k \in \{1, \cdots, r\}$. The notations used are according to those defined in the previous section.

Definition 3.1. The k-th **principal component score**, denoted by W_k, is given by

$$\boxed{W_k = \tilde{\eta}_k^T(\tilde{X} - \tilde{\mu})}$$

Definition 3.2. The k-dimensional principal component vector, denoted by $\mathbf{W}^{(k)}$, is defined as

$$\boxed{\mathbf{W}^{(k)} = [W_1\ W_2\ \cdots\ W_k]^T = \Gamma_k^T(\tilde{X} - \tilde{\mu})}$$

Definition 3.3. The k-th principal component projection vector, denoted by $\tilde{P}_k$, is defined as

$$\boxed{\tilde{P}_k = W_k\tilde{\eta}_k = \tilde{\eta}\tilde{\eta}^T(\tilde{X} - \tilde{\mu})}$$

Sample principal components

Let $\mathbb{X} = [\tilde{X}_1, \tilde{X}_2, \cdots, \tilde{X}_n]$ be a $d \times n$ matrix which I will refer to as a **data matrix**. Since the true mean and variance of the population are generally unknown, we estimate these parameters using the sample mean, $\bar{X}$, and sample covariance, $\mathcal{S}$, respectively.

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n}\tilde{X}_i \text{ and } \mathcal{S} = \frac{1}{n-1}\sum_{i=1}^{n}(\tilde{X}_i - \bar{X})(\tilde{X}_i - \bar{X})^T$$

The data that we use is always centered unless otherwise stated. Centered data will be denoted by $\mathbb{X}_{cent} = [\tilde{X}_1 - \bar{X},\ \tilde{X}_2 - \bar{X},\ \cdots,\ \tilde{X}_n - \bar{X}]$. Data is centered as it simplifies the calculation of $\mathcal{S}$. Covariance matrix, for centered data is given by $\frac{1}{n-1}\mathbb{X}_{cent}\mathbb{X}_{cent}^T$

This information regarding the sample variance and sample mean for the data matrix $\mathbb{X}$ will be provided using the notation $\mathbb{X} \sim \mathrm{Sam}(\bar{X}, \mathcal{S})$.

Assume that the matrix $\mathcal{S}$ has rank $r \leqslant d$. Then, using (3.2), implies that $\mathcal{S} = \widehat{\Gamma}\widehat{\Lambda}\widehat{\Gamma}^T$. Denote the eigen valuess and vectors of this matrix by $\widehat{\lambda}_j$ and $\widehat{\tilde{\eta}}_j$. Assuming that the rank of the matrix $\mathcal{S}$ is $r < d$, then the definitions of Principal components for a sample are given below:

Definition 3.4. The k-th **principal component score** of the data $\mathbb{X}$, denoted by $\tilde{W}_{\bullet k}$ is given by the row vector

$$\boxed{\tilde{W}_{\bullet k} = \widehat{\eta}_k^T \mathbb{X}_{cent}}$$

Definition 3.5. The **principal component data** is the matrix $\mathbb{W}^{(k)}$ consisting of the first k principal component scores, $\tilde{W}_{\bullet k}$, that have been defined above.

$$\boxed{\mathbb{W}^{(k)} = \widehat{\Gamma}_k^T \; \mathbb{X}_{cent} = \begin{bmatrix} \tilde{W}_{\bullet 1} \\ \tilde{W}_{\bullet 2} \\ \vdots \\ \tilde{W}_{\bullet k} \end{bmatrix} = \begin{bmatrix} \widehat{\eta}_1^T \; \mathbb{X}_{cent} \\ \widehat{\eta}_2^T \; \mathbb{X}_{cent} \\ \vdots \\ \widehat{\eta}_k^T \; \mathbb{X}_{cent} \end{bmatrix}}$$

Definition 3.6. The $d \times n$ matrix of k-th **principal component projections** $\mathbb{P}_{\bullet k}$ is defined as follows

$$\boxed{\mathbb{P}_{\bullet k} = \widehat{\eta}_k \tilde{W}_{\bullet k} = \widehat{\eta}_k \left\{ \widehat{\eta}_k^T \mathbb{X}_{cent} \right\}}$$

3.2.1 Properties of principal components

Before mentioning the properties of the principal components, an outline of how principal components are calculated is quite useful.

Let the data under consideration have dimension "*d*". This implies that each observation is a $d \times 1$ column vector whose i-th row corresponds to the value taken by i-th random variable X_i. In other words, all the observations can be viewed as the realizations of the random vector $\tilde{X} = (X_1, X_2, \cdots, X_d)^T$. Let us assume that our data consists of n observations of the random vector $\tilde{X}$. Let the observed vectors be denoted as $\tilde{X}_1, \tilde{X}_2, \cdots, \tilde{X}_d$.

1. As mentioned earlier, the true covariance matrix of the population is often unknown and hence estimated using the sample covariance matrix $\mathcal{S}$. The first step therefore is to center the data and calculate the sample covariance matrix $\mathcal{S}$.

2. The eigen values and eigen vectors of the symmetric matrix $\mathcal{S}$ must be calculated. Let the eigen values be defined as $\widehat{\lambda}_1 \geqslant \widehat{\lambda}_2 \geqslant \cdots \geqslant \widehat{\lambda}_d$.

3. The sample principal component vectors can now be calculated using the formula $\widehat{\eta}_i^T \mathbb{X}_{cent}$.The i-th column of the matrix just defined corresponds to the principal component score corresponding to the i-th observation.

Remark 3.2. I earlier mentioned that PCA utilizes the original set of variables $X_1, X_2, \cdots, X_d$ to produce a new set of variables called principal component scores

$PC_1, PC_2, \cdots, PC_m$ such that $m << d$. It will be shown later that these new set of m variables are nothing but the m random vectors $\tilde{\eta}_i^T \tilde{X}$ for $i \in \{1, 2, \cdots, m\}$. However, it is important to remember that these new set of variables are "best" in the sense that they are those linear combinations of the original variables which explain the maximum variablity in the data. The new set of variables, as shown in Theorem 3.1, have the special property that they are always **uncorrelated** to one another.

Correlation structure of the PC scores

Theorem 3.1. *$\tilde{X}_{d\times 1} \sim (\tilde{\mu}, \Sigma)$ and let $r \leqslant d$ be the rank of matrix Σ. Let $\tilde{W}^{(k)}$ be defined as in definition* (3.2)*.*

1. *The mean and covariance of $\tilde{W}^{(k)}$ are given by*

$$\mathbb{E}[\tilde{W}^{(k)}] = \tilde{0} \text{ and } \mathcal{V}[\tilde{W}^{(k)}] = \Lambda_k$$

2. *The variance and covariance properties of the individual principal component scores are as follows:*

$$\mathcal{V}[W_k] = \lambda_k \text{ for all } k \in \{1, \cdots, r\} \text{ and } \operatorname{cov}(W_k, W_l) = 0 \text{ for all } k \neq l.$$

Proof. $\mathbb{E}[\tilde{W}^{(k)}] = \tilde{0}$ follows from the linearity of expectation. Clearly, proving that $\mathcal{V}[\tilde{W}^{(k)}] = \Lambda_k$ automatically proves the second part of the above theorem. Only thing left to show is that $\mathcal{V}[\tilde{W}^{(k)}] = \Lambda_k$.

$$\begin{aligned}
\mathcal{V}[\tilde{W}^{(k)}] &= \mathbb{E}[\Gamma_k^T(\tilde{X} - \tilde{\mu})^T(\Gamma_k)] \\
&= \Gamma_k^T \mathbb{E}[(\tilde{X} - \tilde{\mu})(\tilde{X} - \tilde{\mu})]\Gamma_k \\
&= \Gamma_k^T \Sigma \Gamma_k \\
&= \Gamma_k^T \Gamma \Lambda \Gamma^T \Gamma_k \\
&= \Lambda_k
\end{aligned}$$

Note that the last step follows from (3.3) that I mentioned earlier. □

Theorem 3.2. *$\tilde{X}_{d\times 1} \sim (\tilde{\mu}, \Sigma)$ such that the matrix Σ has eigen values $\lambda_1, \cdots, \lambda_d$. Then,*

$$\sum_{j=1}^{d} \mathcal{V}[X_j] = \sum_{j=1}^{d} \mathcal{V}[W_j] = \sum_{j=1}^{d} \lambda_j$$

Proof. The theorem (3.1) implies that $\sum_{j=1}^{d} \mathcal{V}[W_j] = \sum_{j=1}^{d} \lambda_j$. So, it is left to prove that $\sum_{j=1}^{d} \mathcal{V}[X_j] = \sum_{j=1}^{d} \lambda_j$. From the definition of Σ, $\sum_{j=1}^{d} \mathcal{V}[X_j] = \operatorname{trace}(\Sigma)$. Since Σ and Λ are similar matrices (i.e. $\Sigma = \Gamma^T \Lambda \Gamma$), the trace of both the matrices will be equal. □

Remark 3.3. Observe that the theorem 3.2 implies that the total variation explained by all the old variables is equal to that explained by the new variables.

Proposition 3.1. *Let $\tilde{X} \sim (\tilde{\mu}, \Sigma)$. Assume that Σ has rank d. For $k \leqslant d$, consider the k-dimensional principal component vector, $\tilde{W}^{(k)}$, of $\tilde{X}$. If Σ has the spectral decomposition $\Sigma = \Gamma\Lambda\Gamma^T$, then*

$$\text{cov}[\tilde{X}, \tilde{W}^{(k)}] = \Gamma\Lambda\mathcal{I}_{d\times k}$$

In particular,

$$\text{cov}(X_j, W_l) = \lambda_l \eta_{lj}$$

In the above notation, λ_l is the l-th largest eigen value. $\tilde{W}^{(k)} = [W_1\ W_2\ \cdots\ W_k]^T$. Finally, $\tilde{\eta}_l = (\tilde{\eta}_{l1}\ \tilde{\eta}_{l2}\ \cdots \tilde{\eta}_{ld})^T$.

Proof. Without loss of generality, the random vector $\tilde{X}$ can be assumed to have zero mean. Then, using the definition of covariance and principal component score, we can write that:

$$\text{cov}[\tilde{X}, \tilde{W}^{(k)}] = \mathbb{E}[(\tilde{X} - \tilde{0})(\tilde{W}^{(k)} - \tilde{0})^T] = \mathbb{E}(\tilde{X}\tilde{X}^T\Gamma_k) = \Sigma\Gamma_k = \Sigma\Lambda\mathcal{I}_{d\times k}$$

because $\Gamma^T\Gamma_k = \mathcal{I}_{d\times k}$.

However, if $\mathbb{E}[\tilde{X}] = \tilde{\mu} \neq \tilde{0}$, then

$$\text{cov}[\tilde{X}, \tilde{W}^{(k)}] = \mathbb{E}[(\tilde{X} - \tilde{\mu})(\tilde{W}^{(k)})^T] = \mathbb{E}[\tilde{X}(\tilde{W}^{(k)})^T] - \mathbb{E}[\tilde{\mu}(\tilde{W}^{(k)})^T] = \mathbb{E}[\tilde{X}(\tilde{W}^{(k)})^T]$$

Above statement holds since $\mathbb{E}[\tilde{W}^{(k)}] = \tilde{0}$. The result follows from the above calculations. □

Remark 3.4. It was shown in theorem 3.1 that the set of new variables $W_1, W_2, \cdots, W_d$ are uncorrelated variables. However, theorem 3.1 implies that it is possible for an old variable to be correlated with the new variable.

3.3 PCA - A variance maximization technique

I earlier mentioned that the variables $\{\tilde{\eta}_1^T\tilde{X}, \tilde{\eta}_2^T\tilde{X}, \cdots, \tilde{\eta}_d^T\}$ are the new set of variables that are derived using the original set of variables $X_1, X_2, \cdots, X_d$. It was also stated that the new set of variables are always uncorrelated irrespective of correlation relations between the original set of variables.

In this section, we will prove that the eigen vectors of the covariance matrix represent the perpendicular directions in the space (where the data lies) in which out data exhibits "maximum" variance. What "maximum variance" implies will become more clear in theis section.

Let $\tilde{\beta}_1$ be the vector in $\mathbb{R}^d$ such that the variance of $\tilde{\beta}_1^T\tilde{X}$ is maximized.

$$\mathcal{V}[\tilde{\beta}_1^T\tilde{X}] = \tilde{\beta}_1^T\Sigma\tilde{\beta}_1 \tag{3.4}$$

Clearly, the quantity in (3.4) is unbounded as $\tilde{\beta}_1$ can take the value of any vector of arbitrarily large magnitude. Therefore, a unique vector that maximizes the variance in (3.4) can be obtained only after placing a restriction on the magnitude of $\tilde{\beta}$. Therefore, the optimization problem that we want to solve is

$$\text{maximize } \tilde{\beta}_1^T\Sigma\tilde{\beta}_1 \text{ over all } \tilde{\beta}_1 \in \mathbb{R}^d, \text{ under the constraint } \tilde{\beta}^T\tilde{\beta} = 1$$

Since the problem we have is that of constrained optimization, it can be solved using the Lagrange multiplier method.

$$\mathcal{L}(\tilde{\beta}_1) = \tilde{\beta}_1\Sigma\tilde{\beta}_1 - \theta_1(1 - \tilde{\beta}_1^T\tilde{\beta}_1), \quad \text{where } \theta_1 = \text{Lagrange multiplier} \tag{3.5}$$

Differentiating both the sides of (3.5), we get

$$\frac{\partial\mathcal{L}(\tilde{\beta}_1)}{\partial\tilde{\beta}_1} = \Sigma\tilde{\beta}_1 - \theta_1\tilde{\beta}_1 = 0$$

Therefore, the value of θ_1 is such that

$$\Sigma\tilde{\beta}_1 = \theta_1\tilde{\beta}_1 \tag{3.6}$$

This implies that the pair $(\theta_1, \tilde{\beta}_1)$ is one of the eigen value-eigen vector pairs of the matrix Σ.

In order to identify which eigen value-eigen vector pair it corresponds to, we use the information that the variance of $\tilde{\beta}_1^T\tilde{X}$ is maximum. We know that

$$\begin{aligned} \mathcal{V}[\tilde{\beta}_1^T\tilde{X}] &= \tilde{\beta}_1^T\Sigma\tilde{\beta}_1 &(3.7)\\ &= \tilde{\beta}_1^T\theta_1\tilde{\beta}_1 &(3.8)\\ &= \theta_1 \text{ since } \tilde{\beta}_1^T\tilde{\beta}_1 = 1 &(3.9) \end{aligned}$$

From equations (3.6) and (3.7), it is clear that θ_1 is an eigen value of highest possible value. This means that $\theta_1 = \lambda_1$ and $\tilde{\beta}_1 = \tilde{\eta}_1$.

Now, we want the vector $\tilde{\beta}_2$ such that

$$\begin{aligned} \tilde{\beta}_2^T\tilde{\beta}_2 &= 1\\ \text{cov}(\tilde{\beta}_1{}^T\tilde{X}, \tilde{\beta}_2{}^T\tilde{X}) &= 0 &(3.10)\\ \mathcal{V}[\tilde{\beta}_2^T\tilde{X}] &= \tilde{\beta}_2^T\Sigma\tilde{\beta}_2 \text{ is the second largest projection variance} \end{aligned}$$

Simplification of the condition that the covariance between the first two principal components scores is zero is given below:

$$
\begin{aligned}
&\operatorname{cov}(\tilde{\beta}_1^{\,T}\tilde{X}, \tilde{\beta}_2^{\,T}\tilde{X}) = 0 \\
\implies &\mathbb{E}[(\tilde{\beta}_1^T\tilde{X} - \tilde{\beta}_1^T\tilde{\mu})(\tilde{\beta}_1^T\tilde{X} - \tilde{\beta}_1^T\tilde{\mu})^T] = 0 \\
\implies &\tilde{\beta}_1^T\Sigma\tilde{\beta}_2 = \tilde{\beta}_1^T(\lambda_2\tilde{\beta}_2) = 0 \\
\implies &\boxed{\tilde{\beta}_1^T\tilde{\beta}_2 = 0 \text{ and } \tilde{\beta}_2^T\tilde{\beta}_1 = 0}
\end{aligned}
$$

Conversely, if $\tilde{\beta}_1^T\tilde{\beta}_2 = 0$ and $\tilde{\beta}_2^T\tilde{\beta}_1 = 0$, then

$$
\begin{aligned}
\text{Given that } \tilde{\beta}_1^T\tilde{\beta}_2 &= \det(\tilde{\beta}_1^T\tilde{\beta}_2) = \det(\tilde{\beta}_1\tilde{\beta}_1^T) = 0 \\
\text{Thus, } \det(\tilde{\beta}_1^T\Sigma\tilde{\beta}_2) &= \det(\tilde{\beta}_2^T\Sigma\tilde{\beta}_1) = \det(\tilde{\beta}_2\tilde{\beta}_1^T) \times \det\Sigma
\end{aligned}
$$

Therefore, we showed that the conditions that $\tilde{\beta}_1^T\tilde{\beta}_2 = \tilde{\beta}_2^T\tilde{\beta}_1 = 0$ and $\operatorname{cov}(\tilde{\beta}_1^T\tilde{X}, \tilde{\beta}_2^T\tilde{X}) = 0$ Hence, the problem mentioned in 3.10 now converts to finding a vector $\tilde{\beta}_2$ such that the variance of the vector $\tilde{\beta}_2^T\tilde{X}$ is maximized under the constraints

$$
\begin{aligned}
\tilde{\beta}_2^T\tilde{\beta}_2 &= 1 \\
\tilde{\beta}_2^T\tilde{\beta}_1 = \tilde{\beta}_1^T\tilde{\beta}_2 &= 0
\end{aligned}
$$

The Lagrange equation will now be

$$
\mathcal{L}(\tilde{\beta}_2) = \tilde{\beta}_2^T\Sigma\tilde{\beta}_2 + \theta_2(1 - \tilde{\beta}_2^T\tilde{\beta}_2) + \theta_{2\bullet}\tilde{\beta}_1\tilde{\beta}_2 \tag{3.11}
$$

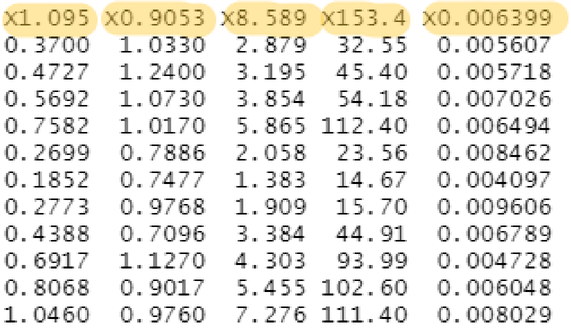
with θ_2 and $\theta_{2\bullet}$ as the legrange multipliers.

3.4 Visualizing PCA and the need to standardize data

This section discusses the different ways in which PCA can be visualized. In addition to that, this section discusses the need to standardize the raw data before applying data analysis.

Wisconsin Diagnostic Breast Cancer (WDBC)

We return to the WDBC data. As mentioned earlier, it consists of data collected from 569 individuals suffering from breast cancer. Values of thirty different quantitative variables were measured on each individual. In addition to these 30 variables, there is a categorical variable indicating the diagnosis of the patient. It is a binary random variable that can take the values “M” (malignant) and “B” (benign). The first few rows and columns of this data set are given in figure 3.1. Note that the first variable



```
x1.095 x0.9053 x8.589 x153.4 x0.006399
0.3700  1.0330  2.879  32.55  0.005607
0.4727  1.2400  3.195  45.40  0.005718
0.5692  1.0730  3.854  54.18  0.007026
0.7582  1.0170  5.865 112.40  0.006494
0.2699  0.7886  2.058  23.56  0.008462
0.1852  0.7477  1.383  14.67  0.004097
0.2773  0.9768  1.909  15.70  0.009606
0.4388  0.7096  3.384  44.91  0.006789
0.6917  1.1270  4.303  93.99  0.004728
0.8068  0.9017  5.455 102.60  0.006048
1.0460  0.9760  7.276 111.40  0.008029
```

Figure 3.2: Some of the variables in WDBC data

refers to the unique ID given to a particular patient while the second variable is the categorical variable defined earlier.

The characters highlighted in yellow in the picture 3.2 are five of the 30 different quantitative variable values noted as part of the study. Clearly, the variable corresponding to the 4th column exhibits large variance due to the large values that the variable takes. On the contrary, the variable corresponding to the 5th column has a variance close to 0. From 3.2, we observe that the magnitude of each variable is different. This is a problem because the amount of variation in the data will depend on units in which each variable is defined.

Consider for instance, the third variable of the WDBC data shown in 3.1. This variable corresponds to the mean radius of the breast cancer cells for a given patient. Let us assume that the radius values are measured in micrometers. The first 15 values written in the third column are printed using R code.

```
> breast.cancer.data = read.csv(choose.files())
> cancer.sub = breast.cancer.data[,-c(1,2)]
> cancer.sub[1:15,1]
[1] 20.57 19.69 11.42 20.29 12.45 18.25 13.71 13.00 12.46 16.02 15.78
[12] 19.17 15.85 13.73 14.54

> max(cancer.sub[,1])
[1] 28.11

> min(cancer.sub[,1])
[1] 6.981
```

From the first fifteen values listed above, we can say that the radius of the cancerous cells is probably lying between $10\mu m$ and $20\mu m$ for most observations. This gives us an approximate range of $10\mu m$. However the maximum and minimum values taken by the variable implies that it has a range of $28.11\mu m - 6.981\mu m = 21.129\mu m$.

However, the range of the radius variable would have been quite close to 0 if the radius was measured in meters instead of micro-meters. The range would then be $21.129 \times 10^{-6}\mu m \approx 0$. Therefore, the spread associated with a given variable is dependent on the units in which it is measured. This implies that the principal component vectors (or the eigen vectors of Σ) are more likely to align in the directions which exhibit large variance and this large variation might simply be due the choice of the units in which the variable is expressed. The problem can be better visualized with the help of **parallel co-ordinate plots**.

Parallel co-ordinate plots

Parallel co-ordinate plots are one of tools used to visualize multivariate data. It allows us to look at multiple quatitative variables at once. Consider the WDBC data again. A parallel co-ordinate plot for a subset of the data is given in Figure 3.3. In this plot, the values taken by the first five variables for three observations is shown graphically.

```
> library(ggplot2)
>library(GGally)   # GGally is required for parallel co-ord plots
> ggparcoord(cancer.sub[200:202, 1:5],groupColumn = 1, scale = "globalminmax" )
```

In the above R code, the 200th, 201st and 202nd observations are plotted. In addition to that, the values taken by the first five variables is plotted for each of the three observations. Note that the code "scale = globalminmax" implies that the raw data is plotted as it is without applying any transformations. The plot is shown in Figure 3.3.

The horizontal axis mentions the names of the first five variables. Each vertical line passing through the variable points on x-axis are independent axes which measure the magnitude of the corresponding variable. All the points joined by a line correspond to values of the same observation. Since each variable has its own co-ordinate and all such co-ordinates are parallel to each other, the plot in figure 3.3 is called a parallel co-ordinate plot.

Parallel co-ordinate plot for raw WDBC data

The parallel co-ordinate plot for raw WDBC data is given in figure 3.4. The variables of the breast cancer data set have been renamed $1, 2 \cdots, 30$ respectively for the sake of convenience in visualizing parallel co-ordinate plots.

It is quite clear from the parallel plot that the fourth, fourteenth and twenty-fourth variables of the WDBC dataset vary to a great extent. In fact, the variation found in other variables is almost negligible in comparison to these three variable. This means that the eigen vectors of Σ (or $\mathcal{S}$) will be dragged towards the axes along which these three variables are present. This, in fact, can be seen by observing the first three eigen vectors of Σ in 3.5. As expected, the weights corresponding to the fourth and twenty-fourth variables are the largest in magnitude for the first two eigen

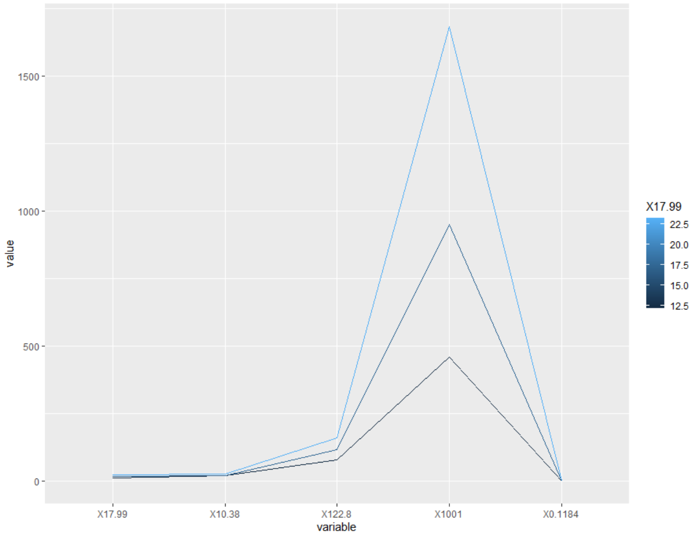


Figure 3.3: Parallel co-ordinate plot for a subset of the WDBC data.

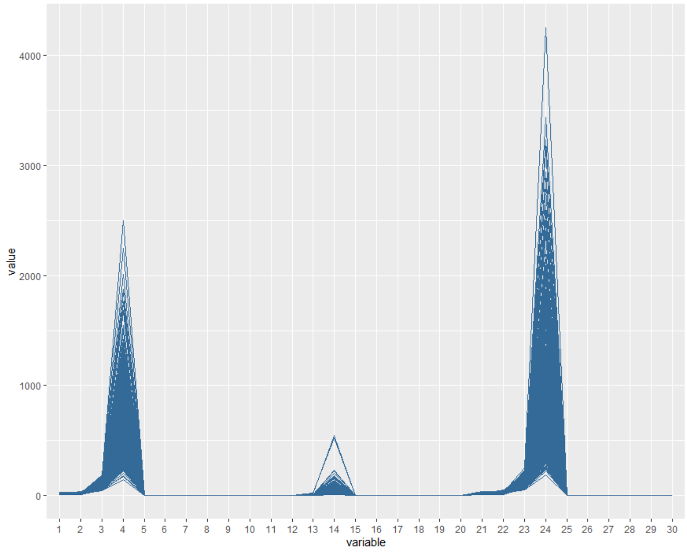
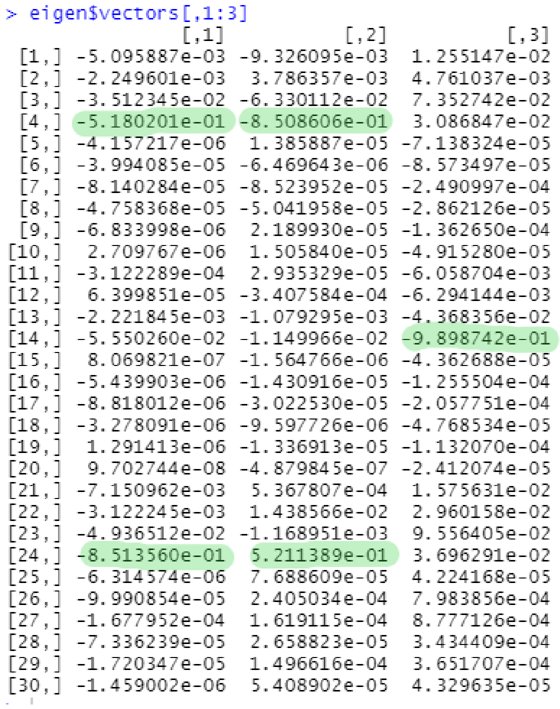


Figure 3.4: Parallel co-ordinate plot for untransformed WDBC data.

vectors. However, the third eigen vector has maximum weight for the fourteenth variable which is the third largest varying variable in the data set.

```
> var = cov(cancer.sub)
> dim(var)
[1] 30 30
eigen = eigen(var, symmetric = T, only.values = F)
```



```
> eigen$vectors[,1:3]
               [,1]          [,2]          [,3]
 [1,] -5.095887e-03 -9.326095e-03  1.255147e-02
 [2,] -2.249601e-03  3.786357e-03  4.761037e-03
 [3,] -3.512345e-02 -6.330112e-02  7.352742e-02
 [4,] -5.180201e-01 -8.508606e-01  3.086847e-02
 [5,] -4.157217e-06  1.385887e-05 -7.138324e-05
 [6,] -3.994085e-05 -6.469643e-06 -8.573497e-05
 [7,] -8.140284e-05 -8.523952e-05 -2.490997e-04
 [8,] -4.758368e-05 -5.041958e-05 -2.862126e-05
 [9,] -6.833998e-06  2.189930e-05 -1.362650e-04
[10,]  2.709767e-06  1.505840e-05 -4.915280e-05
[11,] -3.122289e-04  2.935329e-05 -6.058704e-03
[12,]  6.399851e-05 -3.407584e-04 -6.294144e-03
[13,] -2.221845e-03 -1.079295e-03 -4.368356e-02
[14,] -5.550260e-02 -1.149966e-02 -9.898742e-01
[15,]  8.069821e-07 -1.564766e-06 -4.362688e-05
[16,] -5.439903e-06 -1.430916e-05 -1.255504e-04
[17,] -8.818012e-06 -3.022530e-05 -2.057751e-04
[18,] -3.278091e-06 -9.597726e-06 -4.768534e-05
[19,]  1.291413e-06 -1.336913e-05 -1.132070e-04
[20,]  9.702744e-08 -4.879845e-07 -2.412074e-05
[21,] -7.150962e-03  5.367807e-04  1.575631e-02
[22,] -3.122245e-03  1.438566e-02  2.960158e-02
[23,] -4.936512e-02 -1.168951e-03  9.556405e-02
[24,] -8.513560e-01  5.211389e-01  3.696291e-02
[25,] -6.314574e-06  7.688609e-05  4.224168e-05
[26,] -9.990854e-05  2.405034e-04  7.983856e-04
[27,] -1.677952e-04  1.619115e-04  8.777126e-04
[28,] -7.336239e-05  2.658823e-05  3.434409e-04
[29,] -1.720347e-05  1.496616e-04  3.651707e-04
[30,] -1.459002e-06  5.408902e-05  4.329635e-05
```

Figure 3.5: First three eigen vectors of the raw data's covariance matrix.

If the variables 4, 14 and 24 are the most informative variables for the breast cancer data, then the PCA on raw data would give reliable results. However, if at least one of these three variables happens to have large variance because of the scale it was measured on, then we are likely to make wrong predictions and estimates. If the latter scenario occurs, **scaling the raw data** is a possible solution to our problem.

Definition 3.7. Let $\tilde{X} \sim (\tilde{\mu}, \Sigma)$. Define a diagonal matrix Σ_{diag} (different from the sample covariance matrix Σ) with variance of X_i as the (i, i)-th element of the matrix Σ_D.

$$\Sigma_{diag} = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_d^2 \end{bmatrix}$$

Then, the **scaled** or **standardized** random vector $\tilde{X}_{scale}$ is given by

$$\tilde{X}_{scale} = \Sigma_{diag}^{-\frac{1}{2}}(\tilde{X} - \tilde{\mu}) \text{ where } \mathbb{E}[\tilde{X}] = \tilde{\mu}$$

Definition 3.8. Similarly, for $\mathbb{X} \sim \text{Sam}(\bar{X}, \mathcal{S})$, the **scaled** or **standardized** data is analogously defined as given below. Note that $\mathcal{S}_{diag}$ is defined just like the way Σ_{diag} was defined earlier.

$$\mathbb{X}_{scale} = \mathcal{S}_{diag}^{-\frac{1}{2}}(\mathbb{X}_{cent})$$

Theorem 3.3. *Let $\tilde{X} \sim (\tilde{\mu}, \Sigma)$ and assume that the matrix Σ has rank d. Σ_{diag} is defined as in definition 3.7. Also, $\tilde{X}_{scale} = \Sigma_{diag}^{-\frac{1}{2}}(\tilde{X} - \tilde{\mu})$. Then the following two results hold.*

1. *The covariance matrix of $\tilde{X}_{scale}$ is the matrix of correlation coefficients $\mathcal{R}$.*

$$\mathcal{V}[\tilde{X}_{scale}] = \Sigma_{diag}^{-\frac{1}{2}}\Sigma\Sigma_{diag}^{-\frac{1}{2}} = \mathcal{R}$$

2. *The covariance between the i-th and j-th elements of $\tilde{X}_{scale}$ is equal to the correlation between the same two elements.*

$$\textit{If } \tilde{X}_{scale} = (X_1^s, X_2^s, \cdots, X_d^s), \textit{ then } \text{cov}(X_i^s, X_j^s) = \frac{\text{cov}(X_i, X_j)}{\sqrt{\mathcal{V}[X_i]}\sqrt{\mathcal{V}[X_j]}}$$

Proof. The theorem can be proved by stright-forward calculations. Note that the matrix Σ needs to be full rank to ensure the existance of $\Sigma_{diag}^{-\frac{1}{2}}$. □

Corollary 3.1. *Let $\tilde{X} \sim (\tilde{\mu}, \Sigma)$. Let the rank of Σ be r and let $\mathcal{R}$ be the corresponding correlation matrix. Then,*

$$\text{trace}(\mathcal{R}) = \sum_{j=1}^{r} \lambda_j^{scale} = r$$

Proof. If Σ is a matrix of rank $r \leqslant d$, then Σ will be similar to a diagonal matrix $\mathcal{D}$ of rank r. This is because Σ is a symmetric matrix. $\mathcal{D}$ is a diagonal matrix of rank r implies that there are r non-zero diagonal entries in the matrix $\mathcal{D}$. —— □

Example 3.1. Let us go back to the WDBC dataset. R code to calculate the trace of the correlation matrix is given below.

```
> var = cov(cancer.sub)
> corr.matrix = cov2cor(var)
> trace = 0
> a = corr.matrix[1,1]
> for (i in 1:30) {
+  trace = trace + corr.matrix[i,i]
+  i = i+1
+}
> trace
[1] 30
```

The sum of the r largest eigen values of the correlation matrix will be equal to the sum of all eigen values of $\mathcal{R}$ because $\mathcal{R}$ must have $d - r$ zero eigen values. This is because $\mathcal{R}$ is similar to a diagonal matrix Λ of rank r.

```
> eigen2 = eigen(corr.matrix, symmetric = T, only.values = F)
> sum(eigen2$values)
[1] 30
```

The fact that the trace of $\mathcal{R}$ is 30 implies that none of the variables involved exhibit linear dependencies on each other. △

3.5 Principal Component analysis as a change of basis problem

3.5.1 Toy example

Consider the following hypothetical experiment to understand the purpose of Principal component analysis. Let us assume that we are interested in studying the motion of a spring placed in a room. The system we are interested in consists of a spring and a mass "M" attached to it. If the spring is released a small distance away from the equilibrium, then the spring begins to oscillate about its equilibrium position. In this case, we know that the spring's motion can be viewed as a one dimensional motion along the axis of the spring. However, consider the case where experimenters have no clue of how the spring moves. Their aim instead is to identify the dynamics of the spring.

In such a scenario, the experimenters would probably record the snapshots of the moving spring at different points of time using three movie cameras (since we live in a 3-dimensional world) placed at three different locations as depicted in the Picture 3.6. The position of the spring is plotted in a 2-dimensional world spanned by each movie camera. The experimenters might have arbitrarily chosen their camera angles to measure the motion because they may or may not know the perfect way to view the motion. The experimenters must therefore utilize this 2D information to somehow identify that there exists a one dimensional axis or direction along which the dynamics of the spring can be well observed. This is what PCA does, it identifies the direction along the axis of the spring using the 2D data we procure from each camera.

This example is important because this also helps us to understand what happens in the real world. We often do not know which kind of measurements best reflect the dynamics of interest. Just like in the spring experiment, we often do not know the number of variables we need to sufficiently understand the property of interest. We might even end up using more variables than required. In addition to all this, real world data is imperfect due to noise. In the spring example, noisy data could result due to imperfect cameras or aerial friction. This is where principal component analysis comes into picture. PCA allows us to **re-express the data we have** in a way that uncovers hidden structures which were not initially apparent in the raw data.

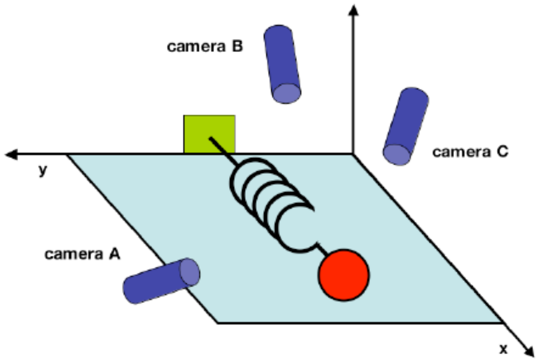


Figure 3.6: Toy example to understand PCA. Picture taken from [Shl14]

3.5.2 Change of Basis

As mentioned earlier, PCA aims to "re-express the data" we have in a way that uncovers important information about the system under consideration. This re-expression of data happens because PCA *changes the basis vectors* with respect to which we view the data points in space.

An important assumption to remember is that PCA assumes that the data points lie in **low dimensional linear subspace** of the original space. The data points are NOT assumed to lie on a geometric surface that has a curvature. For example, PCA can not extract useful information from a data set which has all its points generated from the uniform density on unit sphere. I would like to mention a assumption that is often assumed without question. It is that the orthonormal basis with respect to which we represent the raw data is the standard basis $\{\tilde{e}_1, \tilde{e}_2, \cdots, \tilde{e}_n\}$ where $e_i = (\delta_{ij})_{j=1}^{m}$. Additionally, all the given data points, without loss of generality, are assumed to be centered and scaled.

Notation: We will assume that each data point or vector we take note of lies in $\mathbb{R}^m$. The number of samples collected is given by n. Each data vector is denoted by $\tilde{x}_i$ for $i \in \{1, 2, \cdots, n\}$. The $m \times n$ matrix $\mathbf{X}$ consists of the sample vectors $\tilde{x}_i$'s as its n column vectors.

Now, let $\mathbf{X}$ be the original $m \times n$ data matrix. Let $\mathbf{Y}$ be a $m \times n$ matrix related to $\mathbf{X}$ through the invertible linear transformation $\mathbf{P}$ of size $m \times m$. Then,

$$\mathbf{PX} = \mathbf{Y} \text{ or } \mathbf{P}\tilde{x}_i = \tilde{y}_i \text{ for all } i \in \{1, 2, \cdots, n\}$$

Further simplification shows that

$$\mathbf{PX} = \begin{bmatrix} \tilde{p}_1 \\ \tilde{p}_2 \\ \vdots \\ \tilde{p}_m \end{bmatrix} \begin{bmatrix} \tilde{x}_1 & \tilde{x}_2 & \cdots & \tilde{x}_n \end{bmatrix} = \begin{bmatrix} \tilde{p}_1 \cdot \tilde{x}_1 & \cdots & \tilde{p}_1 \cdot \tilde{x}_n \\ \vdots & \ddots & \vdots \\ \tilde{p}_m \cdot \tilde{x}_1 & \cdots & \tilde{p}_m \cdot \tilde{x}_n \end{bmatrix} \tag{3.12}$$

From (3.12), it is clear that the i^{th} column of $\mathbf{Y}$ is given by:

$$\tilde{y}_i = \begin{bmatrix} \tilde{p}_1 \cdot \tilde{x}_i \\ \vdots \\ \tilde{p}_m \cdot \tilde{x}_i \end{bmatrix} \tag{3.13}$$

The following important observation can be made using (3.13). Assume that the vectors $\tilde{p}_i$ are unit vectors.

Each coefficient of $\tilde{y}_i$ is given by the dot product of $\tilde{x}_i$ with the corresponding row in $\tilde{p}_i$. Geometrically, the j^{th} component of $\tilde{y}_i$ is the length of the projection of $\tilde{x}_i$ onto the vectors $\tilde{p}_j$ for $j \in \{1, 2, \cdots, m\}$. In other words, the rows of $\mathbf{P}$ form a basis for the column space of $\mathbf{X}$ i.e. we can use the rows of $\mathbf{P}$ to re-express the original data set $\mathbf{X}$.

Finally, what we wish to show is that PCA uses certain vectors called principal components to re-express the original data frame. In terms of the notations we just used, $\tilde{p}_i$'s are the principal component vectors with respect to which we re-express the data $\mathbf{X}$ and get $\mathbf{Y}$ in terms of the basis $\{\tilde{p}_1, \cdots, \tilde{p}_m\}$.

Questions Remaining

By restricting to look for new variables which are **linear combinations** of the original variables, the problem reduces to the problem of change of basis. Now, the natural question that will follow are "What is a "good" choice of the basis?" or "What is the best way to re-express the data matrix $\mathbf{X}$?"

The decision of what is the "best" basis or procedure to re-express the data will depend on the kind of features of properties we would like to retain in our information. In the case of PCA, the property of the data that we are most interested in is **variance**. Therefore, we would like to find the directions (or principal component vectors) in $\mathbb{R}^m$ where the data exhibits maximum variability and minimum redundancy.

Consider the equation (3.12) once again. In (3.12), the original m variables of vector $\tilde{x}$ is converted to a different set of variables via the transformation $\mathbf{P}$. Our aim is to find a transformation such that the new set of variables in $\tilde{y}$ explain maximum variance and eliminate redundancy as much as possible. Since covariance measures redundancy, it implies that the transformation P must diagonalize the matrix Σ_x (covariance matrix of the vector $\tilde{X}$) or in other words we wish to find a basis $\{\tilde{p}_1, \cdots, \tilde{p}_m\}$ such that the basis diagonalizes the matrix Σ_x.These new set of basis vectors are called "**principal component vectors**". Since Σ_x is often unknown, it is estimated as $\frac{1}{n-1}\mathbf{X}\mathbf{X}^T$.

The process of PCA as a change of basis problem can be summarized pictorially in the following manner in 3.7:

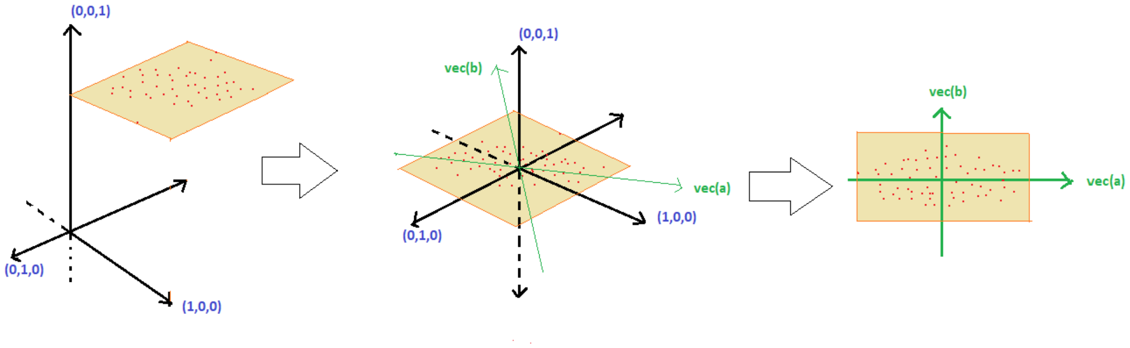


Figure 3.7: Illustration of what PCA is

3.5.3 PCA and spectral decomposition

Since, Σ_x is a symmetric matrix, it is clear that $\Sigma_x = \mathbf{E}^T\mathbf{D}\mathbf{E}$ holds where $\mathbf{E}$ is the matrx of eigen vectors of Σ_x while $\mathbf{D} is a diagonal matrix.$.

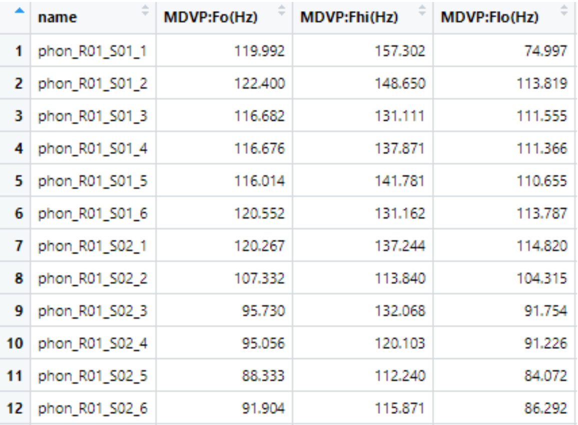
$$\begin{aligned}\widehat{\Sigma}_y &= \mathbf{P}\Sigma_x\mathbf{P}^T\\ &= \mathbf{P}\mathbf{E}\mathbf{D}\mathbf{E}^T\mathbf{P}^T\end{aligned}$$

Clearly by taking $\mathbf{P} = \mathbf{E}^T$, we can show that a matrix with eigen vectors as rows diagonalizes the matrix Σ_x.

3.6 Data Analysis

In this section, we apply PCA technique on real life data set. The data set I considered is the "Parkinsons dataset". This dataset consists of 195 voice recordings. 22

attributes that are a measure of voice (Eg. frequency, amplitude, pitch etc) were calculated. In addition to this, there is an additional variable that records the status of each patient as either 1 or 2. Having a status of 1 implies that the patient has the Parkinson's disease while a status of 0 implies that the patient does not have Parkinson's disease. The data has been shown below for better understanding.



	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)
1	phon_R01_S01_1	119.992	157.302	74.997
2	phon_R01_S01_2	122.400	148.650	113.819
3	phon_R01_S01_3	116.682	131.111	111.555
4	phon_R01_S01_4	116.676	137.871	111.366
5	phon_R01_S01_5	116.014	141.781	110.655
6	phon_R01_S01_6	120.552	131.162	113.787
7	phon_R01_S02_1	120.267	137.244	114.820
8	phon_R01_S02_2	107.332	113.840	104.315
9	phon_R01_S02_3	95.730	132.068	91.754
10	phon_R01_S02_4	95.056	120.103	91.226
11	phon_R01_S02_5	88.333	112.240	84.072
12	phon_R01_S02_6	91.904	115.871	86.292

Figure 3.8: Parkinsons dataset.

```
> library(psy) # install the psy package for scree.plot() function \\
> library(readxl) \\
> # use ‘‘Import Data set’’ in R to read the data into R \\
> dim(parkinsons_data_set)
[1] 195  24
> names(parkinsons_data_set)
[1]  "name"              "MDVP:Fo(Hz)"
[3]  "MDVP:Fhi(Hz)"      "MDVP:Flo(Hz)"
[5]  "MDVP:Jitter(%)"    "MDVP:Jitter(Abs)"
[7]  "MDVP:RAP"          "MDVP:PPQ"
[9]  "Jitter:DDP"        "MDVP:Shimmer"
[11]  "MDVP:Shimmer(dB)" "Shimmer:APQ3"
[13]  "Shimmer:APQ5"      "MDVP:APQ"
[15]  "Shimmer:DDA"       "NHR"
[17]  "HNR"               "status"
[19]  "RPDE"              "DFA"
[21]  "spread1"           "spread2"
[23]  "D2"                "PPE"
> parkinson.sub = parkinsons_data_set[,2:24]
# ‘‘name’’ column removed
> parkinson.sub.2 = parkinson.sub[-c(17)]
# excludes the status attribute
```

Calculating variance and eigen vectors of the covariance matrix:

```
> library(ggfortify)
> pca = prcomp(parkinson.sub.2, scale = T) \\
# gives std  dev values and eigen vectors
> v = c(pca$sdev)^2
> v
 [1] 1.295811e+01 2.485875e+00 1.542030e+00
 [4] 1.464986e+00 9.739161e-01 7.291084e-01
 [7] 5.522449e-01 3.624033e-01 2.898381e-01
[10] 2.241263e-01 1.405651e-01 1.048413e-01
[13] 6.973693e-02 3.816628e-02 2.201169e-02
[16] 1.778754e-02 1.245640e-02 7.214136e-03
[19] 3.496567e-03 1.084955e-03 3.618346e-07
[22] 3.312384e-08
```

Since a long list eigen values are hard to interpret or order in terms of increasing variance, we plot a scree plot to obtain a visual representation of what is going on. Note that a scree plot is nothing but a plot between the magnitude of the variance of a principal component versus the order of that vector in terms of its variance. The scree plot for the Parkinson data is given below:

```
> scree.plot(v, title = "Scree Plot", type = "E")
```

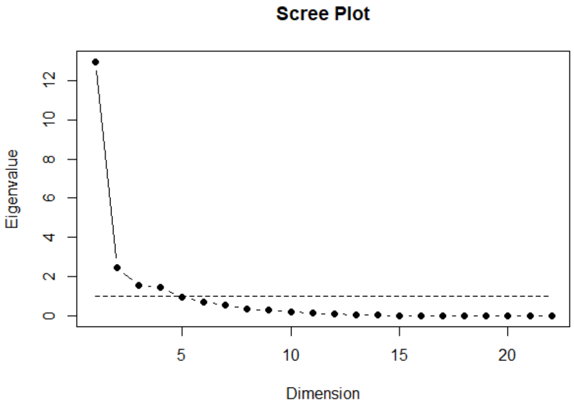


Figure 3.9: Scree plot for the Parkinsons dataset.

From the 3.9, it is clear that the first four or five PC scores dominate over all the others. Therefore, reducing the dimensions to 4 or 5 variables seems reasonable pictorially.

```
> sum(v[1:2])/sum(v)
```

```
[1] 0.7019993
> sum(v[1:3])/sum(v)
[1] 0.7720916
> sum(v[1:6])/sum(v)
[1] 0.9160921
```

Since the first 6 PCs contribute to 90 percent of the variance, we can reduce the dimension of our data to 6 from 22.

In addition to this, visualizing PC score plot in 2 and 3 dimensional world also help us identify hidden patterns in the data set. 2-D and 3-D PC score plots for the parkinson data have been plotted.

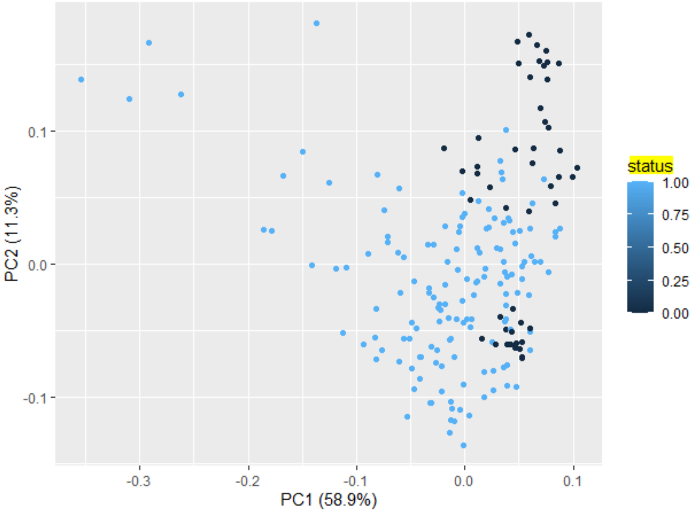


Figure 3.10: 2D PC score plot for Parkinsons dataset.

It is clear from both the 2D and 3D PC score plots that healthy individuals seem to have a positive PC1 score with low variance while the PC1 score of Parkinson's patients is quite spread out. Therefore, there seems to be some association between the voice measurements and a person's status of health.

Although the 2D and 3D PC scores have provided us with the same set of information, it is not the case this way all the time. Sometimes, 3D plots might exhibit information that is not clearly seen in the 2D score plot.

3.7 Limitations and drawbacks of PCA

The limitionas and drawbacks of PCA can be identified by going through the assumptions of PCA.

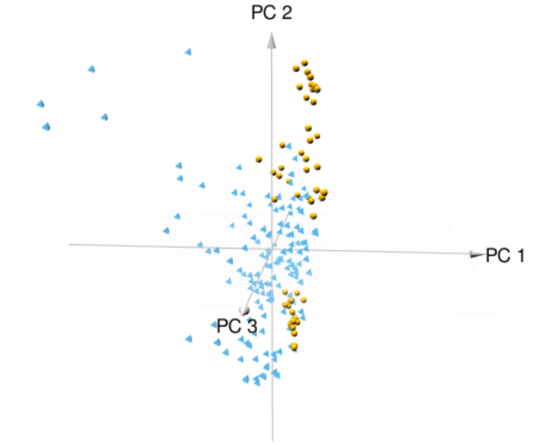


Figure 3.11: 3D PC score plot for Parkinsons dataset.

- The restriction to search for directions of maximum variance that are perpendicular to each other might lead to loss of information as not all data exhibits maximum variance in perpendicular directions.

- PCA is sensitive to scaling, in other words its values depend on the units of the variables under consideration.

- The meaning of the new set of variables that we obtain after using PCA techniqe have no meaning.

- The restriction that the data points must lie approximately on the subspace of the original space definitely implies that the data lying on some surface with curvature cannot be interpreted well with PCA.

- In PCA, we assume that the directions in which variance is low has no interestingproperties of interest. Therefore, if a data set has important information stored indirectons of low variance, then that information will be lost.

Chapter 4

Linear, Quadratic and Fisher Discriminant Analysis

The content of this chapter has been taken from [Koc14], [Gho15], [WB], [BW20], [CBRO], [Fid16], [Ste] and [Ora08].

Definition 4.1. Classification is the prediction of a discrete random variable Y using the observations made on random vector $\tilde{X}$. Here, $\tilde{X}$ is a d-dimensional vector with values in $\mathcal{X} \subset \mathbb{R}^d$ while Y is a uni-variate random variable that takes values from a **finite** set $\mathcal{Y}$.

A **classification rule** or **classifier** is a function $h : \mathcal{X} \longrightarrow \mathcal{Y}$ so that the result of feeding an **unseen** observation $\tilde{X} = \tilde{x}$ is $Y = y$.

A classifier is also called as a **discriminant function**.

The definition above is better understood with the help of an example.

Example 4.1. A subset of the iris dataset (available in R) is given below.

```
   > data("iris")
   > sub.iris = iris[c(1,2,3,51,52,53),]
   > sub.iris
   Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1           5.1         3.5          1.4         0.2     setosa
2           4.9         3.0          1.4         0.2     setosa
3           4.7         3.2          1.3         0.2     setosa
51          7.0         3.2          4.7         1.4 versicolor
52          6.4         3.2          4.5         1.5 versicolor
53          6.9         3.1          4.9         1.5 versicolor
```

Clearly, the above dataset has observations that can be grouped into two categories - data vectors corresponding to the flowers belonging to the species “setosa” and those corresponding to the species of the flower "versicolor". In this example, the variable Y, introduced in definition 4.1, is a binary variable with support $\mathcal{Y} = \{\text{setosa}, \text{versicolor}\}$.The random vector $\tilde{X}$ introduced in 4.1 is a 4-dimensional random vector. The 6 observations made of this four dimensinal random vector are the six rows of the table above.

Our aim, then is to find a function $h : \mathcal{X} \longrightarrow \mathcal{Y}$ such that $h(\tilde{x}_0)$, for some unseen observation (corresponding to a flower) $\tilde{x}_0$, predicts the class or label or species of the flower correctly. △

Definition 4.2. The definition 4.1 can be extended to the scenario where k different classes exist. Let $\tilde{X}$ denote a d-dimensional random vector which takes the values $\mathcal{X} \subset \mathbb{R}^d$ and Y is a discrete random variable that takes k different values or labels. In other words, Y takes one of the values from $\mathcal{Y} = \{1, 2, \cdots, k\}$. Then, classification is the problem of producing a function $h : \mathcal{X} \longrightarrow \mathcal{Y}$ such that $h(\tilde{x})$, for some unseen data vector $\tilde{x}$, predicts the class to which $\tilde{x}$ belongs to.

Example 4.2. The following is an example of a multi-class classification.

```
    > iris[c(1,2,51,52, 102,103),]
    >   Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
51           7.0         3.2          4.7         1.4 versicolor
52           6.4         3.2          4.5         1.5 versicolor
102          5.8         2.7          5.1         1.9  virginica
103          7.1         3.0          5.9         2.1  virginica
```

In the above example, there are three unordered levels of the discrete random variable Y. That is, $\mathcal{Y} = \{ \text{ setosa }, \text{ versicolor }, \text{ virginica } \}$. Here, $\tilde{X}$ is a 4-dimensional vector as in the previous example. △

Definition 4.3. The **true error rate** of a classifier $h : \mathcal{X} \longrightarrow \mathcal{Y}$, as defined earlier, is defined as

$$L(h) = \mathcal{P}(h(\tilde{X}) \neq Y)$$

The true error rate calculates the chance that what h predicts is not equal to the true y value.

However, the distribution of $\tilde{X}$ and Y are unknown. Therefore, the true error rate of a classifier must be estimated. This estimate is called as the empirical/training error rate.

Definition 4.4. The **empirical/training error rate** of a classifier is defined as

$$\widehat{L}_n = \frac{1}{n} \sum_{i=1}^{n} \mathcal{I}(h(\tilde{x}_i) \neq y_i)$$

In the above definition, $\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_n$ are the n realizations of the random variable $\tilde{X}$.

Definition 4.5. One of the possible options for the classifier h is the **Bayes classifier** h^*. $\tilde{X}$ is a d-dimensional random vector while Y is a random variable which takes one of the t possible values $\{1, 2, \cdots, t\}$. For an unseen observation $\tilde{x}$, it is defined as

$$h^*(\tilde{x}) = \max_{\forall k} \mathcal{P}(Y = k|\tilde{X} = \tilde{x}) \tag{4.1}$$

Using Bayes rule, the right hand side of the equation 4.1 can be simplified as

$$\mathcal{P}(Y = k|\tilde{X} = \tilde{x}) = \frac{\mathcal{P}(\tilde{X} = \tilde{x}|Y = k)\mathcal{P}(Y = k)}{\sum_{i=1}^{n} \mathcal{P}(\tilde{X} = \tilde{x}|Y = i)\mathcal{P}(Y = i)} \tag{4.2}$$

As observed above, $\mathcal{P}(Y = k|\tilde{X} = \tilde{x})$ is given more importance than $\mathcal{P}(\tilde{X} = \tilde{x}|Y = k)$ although one can be obtained from the other using equation 4.2. The reason behind this importance can be understood using the **Prosecutor's Fallacy**.

Prosecutor's Fallacy

Prosecutor's fallacy can be better understood with the help of an example. The fallacy is called "Prosecutor's fallacy" because this "logic" is very often used by prosecutors as evidence in favour of the crime committed by the accused.

Let us consider the following hypothetical scenario.
Let Mr. X, who is innocent, be accused of murdering a man "M". Assume that this accusation is solely based on the fact that the fingerprint of the murderer and that of Mr. X match. Mr. A, the prosecutor accusing Mr. X of the murder, claims that *the chance of two fingerprints matching, based on the fingerprint data set they have, is one in ten million. Therefore, it is highly unlikely for Mr. X to be innocent.*

The logic, given in italics, used as evidence to accuse Mr. X of the crime is the Prosecutor's fallacy. However, the reason why the "logic" is a fallacy is explained by Mr. X's defence prosecutor Mr. D. Mr. D explains it using the table in 4

	Match	No match
Guilty	1	0
Innocent	5	50 million

Firstly, for Mr. X to be accused of the crime, it must be assumed that the data base of 50 million finger prints actually includes the finger print of the true murderer. Even if it is assumed that the true murderer's finger print is in the records being used, the probability that two fingerprints match is one in ten million. This implies that there are approximately five individuals whose finger print matches with that of the murderer simply by chance. Hence, the probability of Mr. X being innocent, given that his finger print matches with that of the murderer, is $\frac{5}{6} \approx 83\%$. Thus, according to Mr. D, is not yet proved to be guilty. Therefore, he is innocent until proven guilty.

Mr. A uses the conditional probability $\mathcal{P}($ Match $|$ Innocent $) = 1/10\text{m}$ while Mr. D uses the conditional probability $\mathcal{P}(\text{innocent}|$ match $)$ to prove their respective points. However, the conditional probability used by Mr. A is not important in the current context for two reasons:

1. The conditional probability $\mathcal{P}($ Match $|$ Innocent $) = 1/10\text{m}$ quantifies the efficacy of the method that uses finger prints to identify people. It does not (directly) explain anything about the guilt of Mr. X.
2. We already know that there is a match. So, calculating the probability of obtaining a match completely by chance is of no use.

In conclusion, although it is quite rare for the finger prints of two people to match, Mr. X has a very high chance of being innocent given the evidence.

Prosecutor's fallacy and Bayes classifier

The definition of Bayes classifier in 4.5 involved a d-dimensional random vector $\tilde{X}$ and a discrete random variable Y with a finite set as its support.

4.1 Optimality of Bayes Classifier

Theorem 4.1. *Let $h : \mathcal{X} \longrightarrow \mathcal{Y}$ be a classifier as defined earlier. Assume a two class problem i.e. $\mathcal{Y} = \{0, 1\}$. Then, the Bayes classifier $h^* : \mathcal{X} \longrightarrow \mathcal{Y}$ is optimal in the sense that the true error rate of the Bayes classifier is less than the true error rate of any arbitrary classifier h.*

$$L(h^*) \leqslant L(h) \Longleftrightarrow \mathcal{P}(h^*(\tilde{X}) \neq Y) \leqslant \mathcal{P}(h(\tilde{X}) \neq Y)$$

Additionally, Bayes classifier has the property that it has the least Bayes risk when compared to any other classifier.

Proof. Clearly, the following holds for any arbitrarily chosen classifier h.

$$\mathcal{P}(h(\tilde{X}) \neq Y|\tilde{X} = \tilde{x}) = 1 - \mathcal{P}(h(\tilde{X} - Y = 0)|\tilde{X} = \tilde{x}) \tag{4.3}$$

An observation of the random variable $h(\tilde{X}) - Y$ will equal to 0 only when $(h(\tilde{X}), Y)$ is $(1, 1)$ or $(0, 0)$. Therefore, 4.3 can be simplified as

$$1 - \mathcal{P}(h(\tilde{X}) - Y = 0|\tilde{X} = \tilde{x}) = 1 - \mathcal{P}(h(\tilde{X}) = 0, Y = 0|\tilde{X} = \tilde{x}) - \mathcal{P}(h(\tilde{X}) = 1, Y = 1|\tilde{X} = \tilde{x})$$

Claim: Conditional on $\tilde{X} = \tilde{x}$, the events $\{h(\tilde{X}) = k\}$ and $\{Y = k\}$ are independent. Conditional on $\{\tilde{X} = \tilde{x}\}$, we have that $h(\tilde{X}) = h(\tilde{x})$ i.e. $h(\tilde{X})$ is the value of h evaluated at $\tilde{X} = \tilde{x}$. We also know that $h(\tilde{x}) \in \{0, 1\}$.

Assume without loss of generality that $h(\tilde{x}) = 1$. Then, $\mathcal{P}(h(\tilde{x}) = 0, Y = 1|\tilde{X} = \tilde{x}) = 0$ because $h(\tilde{x}) = 1$. Also, $\mathcal{P}(h(\tilde{x}) = 0|\tilde{X} = \tilde{x}) = 0$ implies the following:

$$\mathcal{P}(h(\tilde{X}) = 0, Y = 0|\tilde{X} = \tilde{x}) = \mathcal{P}(h(\tilde{X}) = 0|\tilde{X} = \tilde{x}) \times \mathcal{P}(Y = 0|\tilde{X} = \tilde{x}) \tag{4.4}$$

Since it is already known that $h(\tilde{x}) = 1$, the following holds true.

$$\mathcal{P}(h(\tilde{x}) = 1, Y = 1|\tilde{X} = \tilde{x}) = \mathcal{P}(Y = 1|\tilde{X} = \tilde{x}) \tag{4.5}$$

Using equation 4.5 and the fact that $\mathcal{P}(h(\tilde{X}) = 1|\tilde{X} = \tilde{x}) = 1$, we can write that:

$$\mathcal{P}(h(\tilde{X}) = 1, Y = 1|\tilde{X} = \tilde{x}) = \mathcal{P}(h(\tilde{X}) = 1|\tilde{X} = \tilde{x}) \times \mathcal{P}(Y = 1|\tilde{X} = \tilde{x}) \tag{4.6}$$

The equations 4.4 and 4.6 prove the following independence condition

$$\mathcal{P}(h(\tilde{X}) = k, Y = k|\tilde{X} = \tilde{x}) = \mathcal{P}(h(\tilde{X}) = k|\tilde{X} = \tilde{x}) \times \mathcal{P}(Y = k|\tilde{X} = \tilde{x}) \text{ for } k \in \{1, 2\}. \tag{4.7}$$

Similar arguments can be used to prove the independence condition 4.7 if we start with the assumption that $h(\tilde{x}) = 0$ given that $\tilde{X} = \tilde{x}$. This proves the claim we stated earlier.

Clearly, $\mathcal{P}(h(\tilde{X}) = k|\tilde{X} = \tilde{x}) = 1$ if $h(\tilde{x}) = k$ and 0 if $h(\tilde{x}) \neq k$. I will use the notation $\mathbb{1}_A$ to represent the indicator function on set A. Then,

$$1 - \mathcal{P}(h(\tilde{X} = Y)|\tilde{X} = \tilde{x}) = 1 - \Big\{\mathbb{1}_{h(\tilde{x})=1}\mathcal{P}(Y = 1|\tilde{X} = \tilde{x}) + \mathbb{1}_{h(\tilde{x})=0}\mathcal{P}(Y = 0|\tilde{X} = \tilde{x})\Big\} \tag{4.8}$$

Since equation 4.8 holds for the Bayes classifier h^* as well, the following difference can be computed.

$$\mathcal{P}(h^*(\tilde{X} = Y)|\tilde{X} = \tilde{x}) - \mathcal{P}(h(\tilde{X} = Y)|\tilde{X} = \tilde{x}) \tag{4.9}$$

$$= \mathcal{P}(Y = 1|\tilde{X} = \tilde{x})(\mathbb{1}_{h^*(\tilde{x})=1} - \mathbb{1}_{h(\tilde{x})=1}) + \mathcal{P}(Y = 0|\tilde{X} = \tilde{x})(\mathbb{1}_{h^*(\tilde{x})=0} - \mathbb{1}_{h(\tilde{x})=0}) \tag{4.10}$$

Using the facts that $\mathcal{P}(Y = 0|\tilde{X} = \tilde{x}) = 1 - \mathcal{P}(Y = 1|\tilde{X} = \tilde{x})$ and $\mathbb{1}_{h^*(\tilde{x})=0} = \mathbb{1}_{\mathcal{X}} - \mathbb{1}_{h^*(\tilde{x})=1}$ in equation 4.9, we can simplify 4.9 as follows:

$$\mathcal{P}(h^*(\tilde{X} = Y)|\tilde{X} = \tilde{x}) - \mathcal{P}(h(\tilde{X} = Y)|\tilde{X} = \tilde{x}) = \Big\{2\mathcal{P}(Y = 1|\tilde{X} = \tilde{x}) - 1\Big\}(\mathbb{1}_{h^*(\tilde{x})=1} - \mathbb{1}_{h(\tilde{x})=1}) \tag{4.11}$$

As a reminder, the Bayes classifier for two classes is given below:

$$h^*(\tilde{x}) = \begin{cases} 0, & \text{if } \mathcal{P}(Y = 0|\tilde{X} = \tilde{x}) > \frac{1}{2} \\ 1, & \text{otherwise} \end{cases}$$

Now, if $\mathcal{P}(Y = 0|\tilde{X} = \tilde{x}) > \frac{1}{2}$ then $h^*(\tilde{x}) = 0$. Clearly, this would make equation 4.11 non-negative. Similarly, we can show that the equation 4.11 is non-negative when $\mathcal{P}(Y = 0|\tilde{X} = \tilde{x}) \leqslant \frac{1}{2}$. This proves that the Bayes classifier is optimal. □

Remark 4.1. Clearly, it is not possible to find a better classifier than the Bayes classifier. This is because the classifier is defined using distributions that are not known to us. Therefore, several approaches to estimate the Bayes classifier are utilized to obtain classifiers.

4.2 Linear and Quadratic Discriminant Analysis

$$\mathcal{P}(Y = k|\tilde{X} = \tilde{x}) = \frac{\mathcal{P}(\tilde{X} = \tilde{x}|Y = k) \times \mathcal{P}(Y = 1)}{\sum_{i=1}^{m} \mathcal{P}(\tilde{X} = \tilde{x}|Y = i)\mathcal{P}(Y = i)}$$

Bayes classifier utilizes the probability $\mathcal{P}(Y = k|\tilde{X} = \tilde{x})$ which, as pointed out earlier, is often not known. One possible way to estimate this unknown probability is by estimating the probabilities $\mathcal{P}(\tilde{X} = \tilde{x}|Y = k)$ and $\mathcal{P}(Y = k)$ for all possible values of k. Once we obtain an estimate $\widehat{p}(k|\tilde{x})$ for the probability $\mathcal{P}(Y = k|\tilde{X} = \tilde{x})$, then Bayes classifier can be estimated as follows:

$$\widehat{h}^*(\tilde{x}) = k \text{ when } \max_{\forall i} \widehat{P}(i|\tilde{x}) = \widehat{P}(k|\tilde{x})$$

Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis(QDA) utilize the density estimation approach to obtain classifiers.

Assume the 2-class scenario once again as it helps understand the ideas better.

Definition 4.6. The **Decision Boundary**, D_h, of a classifier h is defined as the the set of points for which the probability of being in class 0 is the same as that of class 1.

$$D_h = \Big\{\tilde{x} \mid \mathcal{P}(Y = 0|\tilde{X} = \tilde{x}) = \mathcal{P}(Y = 1|\tilde{X} = \tilde{x})\Big\}$$

Notations and terminology:

$$\mathcal{P}(Y = k|\tilde{X} = \tilde{x}) = \frac{\mathcal{P}(\tilde{X} = \tilde{x}|Y = k) \times \mathcal{P}(Y = 1)}{\sum_{i=1}^{m} \mathcal{P}(\tilde{X} = \tilde{x}|Y = i)\mathcal{P}(Y = i)} = \frac{f_k(\tilde{x})\pi_k}{\sum_{i=1}^{m} f_m(\tilde{x})\pi_m} \tag{4.12}$$

The equation 4.12 has been written using the following notations:

- $f_k(\tilde{x}) = \mathcal{P}(\tilde{X} = \tilde{x}|Y = k)$ is called the "**class conditional**".
- $\pi_k = \mathcal{P}(Y = k)$ is called as the "**prior probability**" i.e. it is our belief regarding the probability of an arbitrary point lying in class k before we learn the information that $\tilde{X} = \tilde{x}$.
- The data points belonging to class i, for each i, are assumed to be coming from a distribution with mean $\tilde{\mu}_i$ and covariance matrix Σ_i.

- The number of classes is assumed to be m while the number of points from each class is denoted by n_k such that $n = \sum_{i=1}^{m} n_i$.
- Observations or realizations of the vector $\tilde{X}$ are given given by $\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_n$.

As mentioned earlier, the quantity $\mathcal{P}(Y = k|\tilde{X} = \tilde{x})$ must be estimated since its true value is usually unknown. The estimation of this quantity is usually done with the help of one of the two approaches.

1. Use density estimation methods to estimate prior and class conditional probabilities using the data given.
2. Assume that the class conditional follows a parametric density. Using data, the parameters associated with this density can be estimated.

LDA is based on the second method mentioned above. LDA assumes the class conditional to be **multivariate Gaussian**. The class conditional need not always be Gaussian but this assumptions makes calculations easier since the Gaussian is a well understood distribution. An additional assumption used in LDA is that the covariance matrices Σ_0 **and** Σ_1 **are equal**. This assumption is made to simplify calculations. Since the world around is complicated and hard to understand, we use assumptions in order to create models that we can work with.

Decision boundary in LDA

As mentioned earlier, decision boundary is defined as

$$D = \left\{ \tilde{x} \mid \mathcal{P}(Y = 0|\tilde{X} = \tilde{x}) = \mathcal{P}(Y = 1|\tilde{X} = \tilde{x}) \right\}$$

The decision bounder under the assumptions of LDA can be simplified as follows:

$$\mathcal{P}(Y = 1|\tilde{X} = \tilde{x}) = \mathcal{P}(Y = 0|\tilde{X} = \tilde{x})$$

$$\implies \frac{f_1(\tilde{x})\pi_1}{f_0(\tilde{x})\pi_0 + f_1(\tilde{x})\pi_1} = \frac{f_0(\tilde{x})\pi_0}{f_0(\tilde{x})\pi_0 + f_1(\tilde{x})\pi_1} \implies f_1(\tilde{x})\pi_1 = f_0(\tilde{x})\pi_0 \tag{4.13}$$

Under the assumptions of normality and equality of covariance matrix, the following further simplification can be made. Let $\Sigma_0 = \Sigma_1 = \Sigma$

Then the equation 4.13 can be simplified as follows:

$$\pi_1 \times \exp\left\{ -\frac{1}{2}(\tilde{x} - \tilde{\mu_1})^T \Sigma^{-1} (\tilde{x} - \tilde{\mu_1}) \right\} = \pi_0 \times \exp\left\{ -\frac{1}{2}(\tilde{x} - \tilde{\mu_0})^T \Sigma^{-1} (\tilde{x} - \tilde{\mu_0}) \right\} \tag{4.14}$$

Applying *log* on both sides and then simplifying the equation 4.14 gives us the equation below.

$$-\frac{1}{2}(\tilde{x}-\tilde{\mu_1})^T\Sigma^{-1}(\tilde{x}-\tilde{\mu_1})+\frac{1}{2}(\tilde{x}-\tilde{\mu_0})^T\Sigma^{-1}(\tilde{x}-\tilde{\mu_0})+\log\frac{\pi_1}{\pi_0}=0 \tag{4.15}$$

Simplification after opening the brackets in the above equation gives

$$\tilde{x}^T\Sigma^{-1}\tilde{\mu}_1+\tilde{x}^T\Sigma^{-1}\tilde{\mu}_0+\frac{1}{2}(\tilde{\mu}_0^T\Sigma\tilde{\mu}_0-\tilde{\mu}_1^T\Sigma\tilde{\mu}_1)+\log\frac{\pi_1}{\pi_0}=0 \tag{4.16}$$

Observe that the first two terms of the equation 4.16 are **linear** in $\tilde{x}$ while the remaining terms are constants. Together, 4.16 represents a **linear equation**. The equation 4.16 can be re-written as follows:

$$\tilde{x}^T(\Sigma^{-1}\tilde{\mu}_1+\Sigma^{-1}\tilde{\mu}_0)+\left[\frac{1}{2}(\tilde{\mu}_0^T\Sigma\tilde{\mu}_0-\tilde{\mu}_1^T\Sigma\tilde{\mu}_1)+\log\frac{\pi_1}{\pi_0}\right]=0$$

Let $\Sigma^{-1}\tilde{\mu}_1+\Sigma^{-1}\tilde{\mu}_0$ denote a $d\times 1$ vector $\tilde{\beta}$ while the remaining part of the equation is represented by the scalar a. Therefore, from the equation below it becomes quite clear that 4.16 represents a linear equation.

$$\tilde{x}^T\tilde{\beta}+a=0 \tag{4.17}$$

Therefore, the decision boundary of the 2 class problem is a $d-1$ dimensional **hyperplane** in $\mathbb{R}^d$. Since the decision boundary is linear in $\tilde{x}$, the procedure is called Linear Discriminant Analysis.

Now, assume that $\mathcal{P}(Y=0|\tilde{X}=\tilde{x})>\mathcal{P}(Y=1|\tilde{X}=\tilde{x})$. Using the assumptions of LDA and then using the simplification methods that we just used leads to the following results.

$$\{\tilde{x}\mid\mathcal{P}(Y=0|\tilde{X}=\tilde{x})>\mathcal{P}(Y=1|\tilde{X}=\tilde{x})\}=\{\tilde{x}\mid\tilde{x}^T\tilde{\beta}+a<0\} \tag{4.18}$$

CHECK —- general locus of above equation

Consider for the sake of simplicity the following example where $\tilde{x}$ is assumed to belong to $\mathbb{R}^2$ and $\mathbb{R}^3$ respectively.

Example 4.3. Let $\tilde{x}\in\mathbb{R}^2$ i.e. $\tilde{x}=(x_1\ x_2)^T$. Then, the (4.18) is equivalent to

$$\{\tilde{x}\ :\ x_1\beta_1+x_2\beta_2+a<0\}=\left\{\tilde{x}\ :\ \frac{x_1}{\left(-\frac{a}{\beta_1}\right)}+\frac{x_2}{\left(-\frac{a}{\beta_2}\right)}-1<0\right\} \tag{4.19}$$

The locus of the points $\tilde{x}$ which satifies the expression in 4.19 is the one of the two halves of the $\mathbb{R}^2$ plane that is formed by the line $x_1\beta_1+x_2\beta_2+a=0$

Similarly, if $\tilde{x}\in\mathbb{R}^3$ then the locus of the points, $\tilde{x}$, satisfying $\tilde{x}^T\tilde{\beta}+a<0$ is one of the two halves of $\mathbb{R}^3$ that is created by the plane $x_1\beta_1+x_2\beta_2+x_3\beta_3+a=0$ △

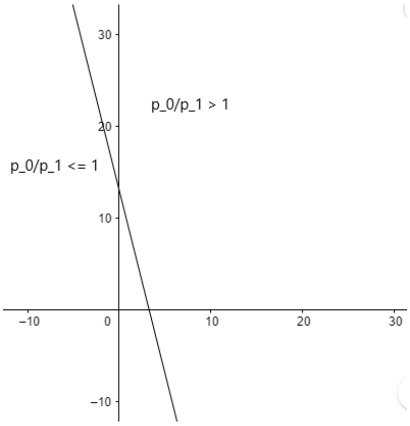


Figure 4.1: Hyperlane in 2-d space

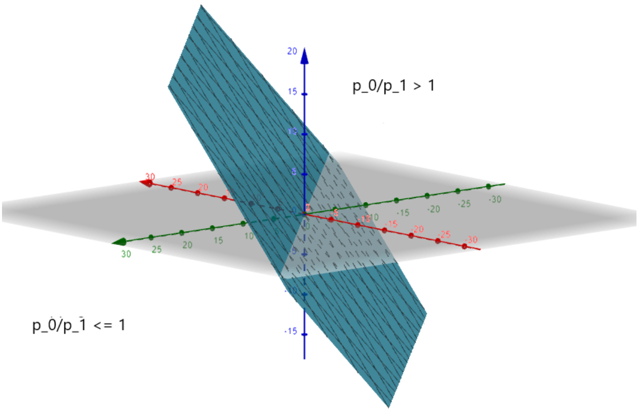


Figure 4.2: Hyperlane in 3-d space

Bayes classifier:

Let $\mathcal{P}(Y=0|\tilde{X}=\tilde{x})$ be denoted by p_0 while $\mathcal{P}(Y=1|\tilde{X}=\tilde{x})$ be denoted by p_1

$$h^*(\tilde{x}) = \begin{cases} 0, & \text{if } \frac{p_0}{p_1} > 1 \\ 1, & \text{if } \frac{p_0}{p_1} \leqslant 1 \end{cases}$$

The equation 4.17 implies that the decision boundary between classes 0 and 1 can be estimated only after estimating the values of $\tilde{\beta}$ and a. This requires us to estimate the mean and covariance matrix of the normal distributions which the m class conditionals are assumed to follow. The estimators for the general m-class problem is mentioned below athough the general m-class problem hasn't yet been discussed in the chapter.

The mean, $\tilde{\mu}_k$, of the points sampled from class k is estimated as

$$\widehat{\mu}_k = \frac{1}{n_k} \sum_{\{i:y_i=k\}} \tilde{x}_i \tag{4.20}$$

Due to the assumption $\Sigma_0 = \Sigma_1 = \cdots = \Sigma_{k-1} = \Sigma$, the common Σ is estimated as follows:

$$\widehat{\Sigma}_k = \frac{1}{n_k - 1} \sum_{\{i:y_i=k\}} (\tilde{x}_i - \widehat{\mu}_k)(\tilde{x}_i - \widehat{\mu}_k)^T$$

$$\widehat{\Sigma} = \frac{\sum_{r=1}^{n} n_r \Sigma_r}{\sum_{i=1}^{n} n_r} \tag{4.21}$$

Observe that the equation 4.21 is a weighted average. Therefore, more weight is given to the covariance of classes with greater n_k value.

Quadratic Discriminant Analysis

Relaxing the condition that $\Sigma_1 = \Sigma_0$ gives us Quadratic Discriminant Analysis. In QDA, class conditional is still assumed to be multivariate normal. Using simplifications similar to that done earlier, it can be shown that the decision boundery in this scenario is the set of all $\tilde{x}$ such that:

$$-\frac{1}{2}\tilde{x}^T\left[\Sigma_1 - \Sigma_0\right]\tilde{x} + \tilde{x}^T\left[\Sigma_1\tilde{\mu}_1 + \Sigma_0\tilde{\mu}_0\right] - \frac{1}{2}\tilde{\mu}_1^T\left(\Sigma_1^{-1} + \Sigma_0^{-1}\right)\tilde{\mu}_0 + \log\frac{\pi_1}{\pi_0} = 0 \quad (4.22)$$

The equation in 4.22 is clearly quadratic in $\tilde{x}$ as it can be re-written in a simplified form as follows:

$$\tilde{x}^T\mathbf{A}\tilde{x} + \tilde{b}^T\tilde{x} + c = 0 \quad (4.23)$$

In the equation above, $\mathbf{A}$ represents a $d \times d$ matrix, $\tilde{b}$ represents a $d \times 1$ vector while c is a scalar from $\mathbb{R}$.

If $\tilde{x} \in \mathbb{R}^2$, then equation 4.23 can be simplified as follows:

$$\tilde{x}^T\mathbf{A}\tilde{x} + \tilde{b}^T\tilde{x} + c = \begin{bmatrix} x_1 & x_2 \end{bmatrix}\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + c = 0 \quad (4.24)$$

$$\implies \left\{a_{11}x_1^2 + a_{22}x_2^2 + (a_{12} + a_{21})x_1x_2\right\} + \left\{b_1x_1 + b_2x_2\right\} + c = 0 \quad (4.25)$$

From the simplifications in 4.24, it is clear that the decision boundary for a 2 class problem in $\mathbb{R}^2$ is parabolic (which is a quadratic equation). Now if we assume $\tilde{x}$ to be a vector in $\mathbb{R}^3$, then the equation 4.23 can be simplified to give a quadratic equation in three variables as follows.

$$\tilde{x}^T\mathbf{A}\tilde{x} + \tilde{b}^T\tilde{x} + c = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + c = 0 \quad (4.26)$$

$$\implies \sum_{i=1}^{3} a_{ii}x_i^2 + \sum_{\forall(i,j):i\neq j} a_{ij}x_ix_j + \sum_{k=1}^{3} b_kx_k + c = 0 \quad (4.27)$$

Clearly, the equation in 4.26 corresponds to that of a **quadratic surface**, hence the name quadratic discriminant analysis.

Remark 4.2. In LDA, we saw that decision boundaries in $\mathcal{R}^3$ are always 2-dimensional subspaces or planes. However, in the case of QDA,the decision boundary in $\mathbb{R}^3$ is a quadratic surface which is not one unique geometric object.
The general equation of a quadratic surface in two and three variables is given below:

$$Ax^2 + By^2 + Dxy + Gx + Hy + J = 0 \text{ where } A, B, D, G, H \in \mathbb{R} \quad (4.28)$$

$$Ax^2 + By^2 + Cz^2 + Dxy + Exz + Fyz + Gx + Hy + Iz + J = 0 \text{ where } A, B, \cdots, J \in \mathbb{R} \tag{4.29}$$

Examples of quadratic surfaces in $\mathbb{R}^3$ are spheres, ellipsoids, paraboloids, hyperboloids etc.

4.3 Multiclass classifier decision boundaries

Since we have already seen the 2-class case in some detail, it is time to go through the multi-class scenario in some detail as well. If the number of classes is $k > 2$, then multiple decision boundaries which we are expected to predict using the data that we have. Certain ideas related to defining decision boundaries have been discussed below. These are the ideas one mostly gets for the first time while trying to figure out how to predict these boundaries.

One vs all classifier

Let us assume that there exist k classes denoted by the numbers $\{1, 2, \cdots, k\}$. One option would be to define k different boundaries, denoted by d_i for $i \in \{1, 2 \cdots, k\}$, as follows:

- d_i if the boundary separating points in class i from the points not in class C_i. Therefore, each d_i is obtained by performing LDA to the two class problem involving classes "i" and "not i".

However, the fact that this idea does not always work can be seen with a simple 3 class scenario.

Example 4.4. Consider a hypothetical case where three classes, C_1, C_2 and C_3 exist. Let the boundaries that we obtain using 2-class LDA be as shown in figure 4.3

Clearly, the class to which points P_1 and P_2 belong to cannot be predicted using this procedure. The point P_1, according to the boundaries given can be concluded to belong to both class C_2 and C_3. On the other hand, the point P_2 belongs to none of the three classes. Both these conclusions are not useful in classification. △

One vs one classifier

Let us again assume the existence of k classifiers. In this procedure, we solve $\binom{k}{2}$ 2-class LDA procedures to obtain a decision boundary between the classes i and j for all possible pairs (i, j) with $i \neq j$. However, this procedure, just like the earlier procedure does not always give useful results.

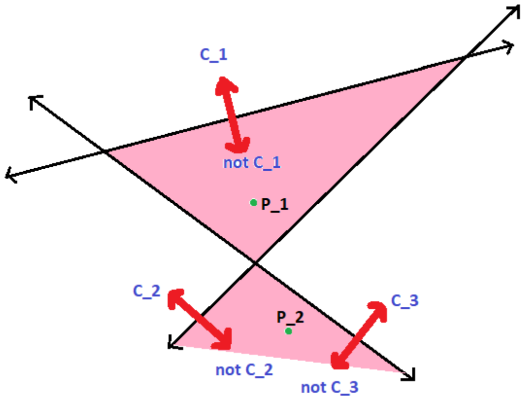


Figure 4.3: 1 vs all classifier example.

Example 4.5. Consider three classes C_1, C_2 and C_3. If the decision boundaries for this scenario turn out to be as shown in the figure 4.4, then it is clear that if an unseen observation belongs to the green triangle, then its class cannot be predicted uniquely. △

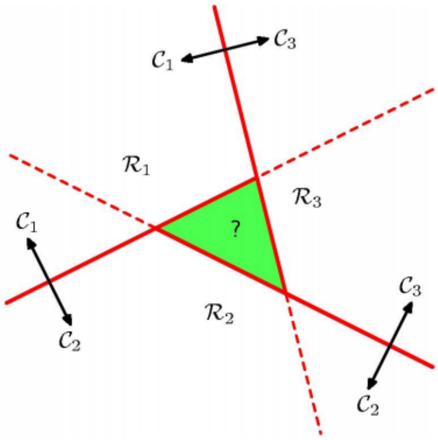


Figure 4.4: 1 vs 1 classifier example. Picture taken from [Fid16]

4.4 Multi-class LDA and QDA

The definitions and proofs seen so far were for the 2-class problem. However, the definition of Bayes classifier along with the methods LDA and QDA can be extended to solve multi-class problems. This section focuses on the scenario where m different classes exist. The multi-class Bayes classifier is defined as follows:

$$h^*(\tilde{x}) = \underset{k}{\text{argmax}} \left[\mathcal{P}(Y = k|\tilde{X} = \tilde{x}) \right] \tag{4.30}$$

Since $\ln : (0, \infty) \longrightarrow \mathbb{R}$ is an increasing function, we know that the value of k that maximizes $\mathcal{P}(Y = k|\tilde{X} = \tilde{x})$ is also the value of k that maximizes $\ln \left[\mathcal{P}(Y = k|\tilde{X} = \tilde{x}) \right]$.

$$\begin{aligned}
\ln \left[\mathcal{P}(Y = k|\tilde{X} = \tilde{x}) \right] &= \ln \left\{ \frac{f_k(\tilde{x})\pi_k}{\sum_{i=1}^{m} f_m(\tilde{x})\pi_m} \right\} && (4.31)\\
&\propto \ln[n(\tilde{x}|\tilde{\mu}_k, \Sigma_k)] + \ln \pi_k && (4.32)\\
&\propto -\frac{1}{2} \ln(\det \Sigma_k) - \frac{1}{2}(\tilde{x} - \tilde{\mu}_k)^T \Sigma^{-1} (\tilde{x} - \tilde{\mu}_k) + \ln \pi_k && (4.33)
\end{aligned}$$

Based on the assumptions involved, the equation (4.31) can be further simplified depending on the choice of method (LDA or QDA).

1. In case of LDA, we have $\Sigma_i = \Sigma$ for all i.

$$\ln \left[\mathcal{P}(Y = k|\tilde{X} = \tilde{x}) \right] \propto -\frac{1}{2}(\tilde{x} - \tilde{\mu}_k)^T \Sigma^{-1} (\tilde{x} - \tilde{\mu}_k) + \ln \pi_k \tag{4.34}$$

2. In case of QDA, common covariance matrix Σ cannot be assumed. Therefore, (4.31) is simplified as:

$$\ln \left[\mathcal{P}(Y = k|\tilde{X} = \tilde{x}) \right] \propto -\frac{1}{2} \ln(\det \Sigma_k) - \frac{1}{2}(\tilde{x} - \tilde{\mu}_k)^T \Sigma_k^{-1} (\tilde{x} - \tilde{\mu}_k) + \ln \pi_k \tag{4.35}$$

Hence, based on whether the procedure chosen is LDA or QDA, the object to be maximized over all possible classes is simplified. Multi-class Bayes classifier is defined as:

$$h^*(\tilde{x}) = \begin{cases} -\frac{1}{2} \ln(\det \Sigma) - \frac{1}{2}(\tilde{x} - \tilde{\mu}_k)^T \Sigma^{-1} (\tilde{x} - \tilde{\mu}_k) + \ln \pi_k, & \text{for LDA} \\ -\frac{1}{2}(\tilde{x} - \tilde{\mu}_k)^T \Sigma_k^{-1} (\tilde{x} - \tilde{\mu}_k) + \ln \pi_k, & \text{for QDA} \end{cases} \tag{4.36}$$

LDA, QDA and the metric function

In this section, we see how LDA and QDA procedures can be simplified to distance comparison problems.

LDA

To begin with, consider the LDA procedure.

Case-1: In addition to the assumption that all covariance matrices have to be equal, assume that all the prior probabilities are equal. In otherwords, it can be interpreted as the assumption that an equal number of points have been sampled for each of the m classes.

Under these two assumptions, the equation (4.36) converts to

$$h^*(\tilde{x}) = \operatorname*{argmax}_k \delta_k(\tilde{x}); \text{ where } \quad \delta_k(\tilde{x}) = -\frac{1}{2}||\tilde{x} - \tilde{\mu}_k||_2 \tag{4.37}$$

The Bayes classifier is now converted into a problem of distance comparison. An unseen observation $\tilde{x}$ is predicted to belong to class k if $\tilde{x}$ is closest to the the mean vector corresponding to class k. This has been explained graphically using the image 4.5.

Remark 4.3. Note that the class of a data point $\tilde{x}$ cannot be determined using a Bayes classifier if the data point is equi-distant to two or more of mean vectors of the data.

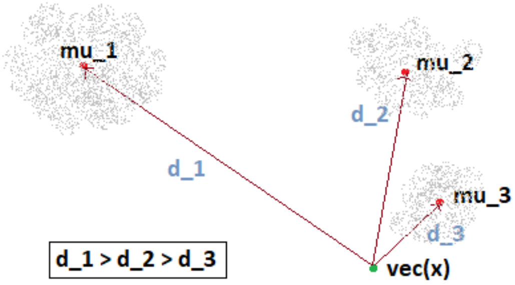


Figure 4.5: Unseen vector $\tilde{x} \in$ class 3

Case-2: Now, consider the case where the covariance matrices are all identity but the priors are unequal i.e. $\Sigma_1 = \Sigma_2 = \cdots = \Sigma_m = \Sigma$ but all π_i's need not necessarily be equal. Under these circumstances, the Bayes classifier can be modified as follows.

$$h^*(\tilde{x}) = \operatorname*{argmax}_k \delta_k(\tilde{x}); \quad \text{where } \delta_k(\tilde{x}) = -\frac{1}{2}(\tilde{x} - \tilde{\mu}_k)^T \Sigma_k^{-1}(\tilde{x} - \tilde{\mu}_k) + \ln \pi_k \tag{4.38}$$

$$= \operatorname*{argmax}_k \exp\{-\frac{1}{2}(\tilde{x} - \tilde{\mu}_k)^T \Sigma_k^{-1}(\tilde{x} - \tilde{\mu}_k)\} \times \pi_k \tag{4.39}$$

From the equation in (4.38), we can see that if the points from a class occur more frequently, i.e. the class has higher prior probability, then it must have a larger posterior probability if all the distance terms were equal in magnitude. However, since this is not always the case. The class with maximum posterior probability must be the term with least distance between the point $\tilde{x}$ and corresponding mean vector.

The value of prior determines the position of the decision boundary as shown in figure 4.6

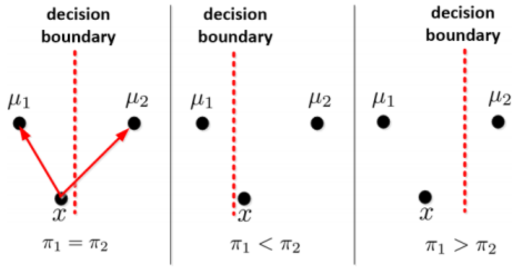


Figure 4.6: Position of decision boundaries

QDA

Case-3: Consider the more general case now. It is that the covariance matrices as well as the priors need not be equal. This is exactly the way it is in QDA.

$$\text{Let } \Sigma_k = \mathbf{U}_k \Lambda_k \mathbf{U}_k^T \text{ for } k \in \{1, 2, \cdots, m\} \text{ be the SVD for matrix } \Sigma_k$$

Since the matrix $\mathbf{U}$ is orthogonal, he inverse of Σ_k can be calculated to be:

$$\sigma_k^{-1} = \mathbf{U}_k \Lambda_k^{-1} \mathbf{U}_k^T$$

Therefore the quadratic term in 4.36 can be simplified as follows:

$$(\tilde{x} - \tilde{\mu}_k)^T \Sigma_k^{-1} (\tilde{x} - \tilde{\mu}_k) = (\tilde{x} - \tilde{\mu}_k)^T \mathbf{U}_k \Lambda_k^{-1} \mathbf{U}_k^T (\tilde{x} - \tilde{\mu}_k) \tag{4.40}$$

$$= (\mathbf{U}_k^T \tilde{x} - \mathbf{U}_k^T \tilde{\mu}_k)^T \Lambda_k^{-1} (\mathbf{U}_k^T \tilde{x} - \mathbf{U}_k^T \tilde{\mu}_k) \tag{4.41}$$

The diagonal matrix Λ_k with non-negative entries can be decomposed as $\Lambda_k^{-1} = \Lambda_k^{-\frac{1}{2}} \Lambda_k^{-\frac{1}{2}}$. Using this piece of information, 4.40 can be simplified as follows:

$$(\tilde{x} - \tilde{\mu}_k)^T \Sigma_k^{-1} (\tilde{x} - \tilde{\mu}_k) = (\mathbf{U}_k^T \tilde{x} - \mathbf{U}_k^T \tilde{\mu}_k)^T \Lambda_k^{-\frac{1}{2}} \Lambda_k^{-\frac{1}{2}} (\mathbf{U}_k^T \tilde{x} - \mathbf{U}_k^T \tilde{\mu}_k)$$

$$= (\Lambda_k^{-\frac{1}{2}} \mathbf{U}_k^T \tilde{x} - \Lambda_k^{-\frac{1}{2}} \mathbf{U}_k^T \tilde{\mu}_k)^T (\Lambda_k^{-\frac{1}{2}} \mathbf{U}_k^T \tilde{x} - \Lambda_k^{-\frac{1}{2}} \mathbf{U}_k^T \tilde{\mu}_k)$$

Consider the following transformation ϕ_k for all $k \in \{1, \cdots, m\}$ where $\mathcal{X}$ is the support of the random vector $\tilde{X}$.

$$\phi_k : \mathcal{X} \longrightarrow \mathbb{R} \text{ such that } \tilde{x} \mapsto \Lambda_k^{-\frac{1}{2}} \mathbf{U}_k^T \tilde{x} \tag{4.42}$$

If we apply the transformation ϕ_k to data coming from class k, then $\tilde{X}$ is transformed to $\Lambda_k^{-\frac{1}{2}}\mathbf{U}_k^T\tilde{X}$ which is a vector with identity covariance matrix. Therefore, this has now transformed into a problem of case one or case two i.e. it is now a comparison of distances or scaled distances.

Conclusion

In conclusion, QDA and LDA deal with maximizing the posterior probability of classes but work with class conditionals and priors.

4.5 Fisher's Discriminant Analysis

Fisher's Discriminant Analysis or FDA is a **supervised** way of **reducing the dimension** of the data.

Definition 4.7. Supervised learning is defined by the use of labeled datasets to train algorithms to classify or predict outcomes accurately. The Bayes classifier that we first observed along with the FDA procedure that we will now see are examples of supervised learning. However, PCA is the example of un-supervised learning as the procedure does not take into account the label information.

LDA, that we just discussed is a **classification technique** while FDA is a **feature extraction technique**. Variables are also referred to as features or attributes.

Definition 4.8. Feature extraction is an attribute reduction process where the original set of features are transformed to produce a smaller set of more meaningful features.

PCA and FDA are examples of feature extraction procedures. In this section, we assume that only 2 classes exist while the data comes from $\mathbb{R}^d$.

Aim:

Our aim is to find a direction/vector, $\tilde{v}$, such that the following two properties are followed.

1. The distance between the mean of the two classes is maximized.
2. Variance of the projected data, for each individual class, has to be minimized.

The figure 4.7 gives a simple example to understand FDA.

FDA is also a **dimension reduction technique** where we wish to learn a vector $\tilde{w}$ such that $\tilde{w}^T\tilde{x}$ is a good representation of the data for classification.

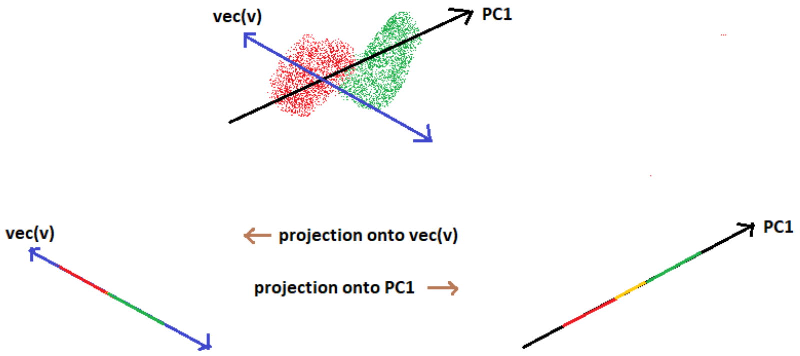


Figure 4.7: Illustration to understand FDA

Notation:

We assume n observations given by vectors $\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_n$ coming from $\mathbb{R}^d$. Each vector $\tilde{x}_i$ belongs to one of the two classes denoted by 0 and 1.

Sample mean of the data points coming from class 0 are denoted by $\widehat{\mu}_0$ while the sample mean of the other class is denoted by $\widehat{\mu}_1$.

$$\widehat{\mu}_t = \frac{1}{n_t} \sum_{\{i|y_i=t\}} \tilde{x}_i \text{ for } i \in \{0,1\}$$

Let $\tilde{w}$ be an arbitrary direction in $\mathbb{R}^d$. We want to find the $\tilde{w}$ which is good for s

Data	Projected data
$\tilde{x}$	$\tilde{w}^T\tilde{x}$
$\tilde{\mu}_0$	$\tilde{w}^T\tilde{\mu}_0$
$\tilde{\mu}_1$	$\tilde{w}^T\tilde{\mu}_1$
Σ_0	$\tilde{w}^T\Sigma_0\tilde{w}$
Σ_1	$\tilde{w}^T\Sigma_1\tilde{w}$

FDA, as mentioned earlier, involves two aims.

1. We wish to maximize the distance between the means of the projected data points.

$$\begin{aligned}
\max_{\tilde{w}} \left[(\tilde{w}^T\tilde{\mu}_1 - \tilde{w}^T\tilde{\mu}_0)^T(\tilde{w}^T\tilde{\mu}_1 - \tilde{w}^T\tilde{\mu}_0)\right] &= \max_{\tilde{w}} \left[(\tilde{\mu}_1 - \tilde{\mu}_0)^T\tilde{w}\tilde{w}^T(\tilde{\mu}_1 - \tilde{\mu}_0)\right] \\
&= \max_{\tilde{w}} \left\{ \operatorname{tr}\left[(\tilde{\mu}_1 - \tilde{\mu}_0)^T\tilde{w}\tilde{w}^T(\tilde{\mu}_1 - \tilde{\mu}_0)\right]\right\} \\
&= \max_{\tilde{w}} \left[\tilde{w}^T(\tilde{\mu}_1 - \tilde{\mu}_0)(\tilde{\mu}_1 - \tilde{\mu}_0)^T\tilde{w}\right] \\
&= \max_{\tilde{w}} \left[\tilde{w}^T\mathcal{S}_B\tilde{w}\right]
\end{aligned}$$

$\mathcal{S}_B$ in the above simplification is called as the **between class covariance matrix**. Therefore, the problem of maximizing the distance between the projected means is equivalent to that of maximizing $\tilde{w}^T\mathcal{S}_B\tilde{w}$ over all possible $\tilde{w} \in \mathbb{R}^d$.

2. The next step is to minimize the variance of projected points within each class.

$$\min_{\tilde{w}} \left\{ \tilde{w}^T \Sigma_0 \tilde{w} \right\} \text{ and } \min_{\tilde{w}} \left\{ \tilde{w}^T \Sigma_1 \tilde{w} \right\} \tag{4.43}$$

Since $\tilde{w}^T\Sigma_0\tilde{w}$ and $\tilde{w}^T\Sigma_1\tilde{w}$ are non-negative functions with respect to variable $\tilde{w}$, problem in (4.43) is equivalent to minimizing the sum of the two functions. That is, (4.43) is equivalent to

$$\min_{\tilde{w}} \tilde{w}^T(\Sigma_0 + \Sigma_1)\tilde{w} = \min_{\tilde{w}} \tilde{w}^T \mathcal{S}_W \tilde{w} \tag{4.44}$$

The matrix $\mathcal{S}_W$ is called **within class covariance matrix**.

To summarize, the problem of FDA is to find a direction $\tilde{w}$ such that

$$\max_{\tilde{w}} \left[\tilde{w}^T \mathcal{S}_B \tilde{w} \right] \text{ and } \min_{\tilde{w}} \left[\tilde{w}^T \mathcal{S}_W \tilde{w} \right] \tag{4.45}$$

The problem in (4.45) is equivalent to the following problem

$$\max_{A} \frac{\tilde{w}^T \mathcal{S}_B \tilde{w}}{\tilde{w}^T \mathcal{S}_W \tilde{w}} \quad \text{where} \quad A = \{\tilde{w} | \tilde{w}^T \mathcal{S}_W \tilde{w} \neq 0\} \tag{4.46}$$

We know that $\mathcal{S}_B = (\tilde{\mu}_1 - \tilde{\mu}_0)^T(\tilde{\mu}_1 - \tilde{\mu}_0)$ while $\mathcal{S}_W = \Sigma_1 + \Sigma_0$. Clearly, both $\mathcal{S}_B$ and $\mathcal{S}_W$ are symmetric matrices. Therefore, (4.45) is the **Rayleigh quotient** of the symmetric matrices $\mathcal{S}_B$ and $\mathcal{S}_W$.

$$\text{Let } R(\tilde{w}) = \frac{\tilde{w}^T \mathcal{S}_B \tilde{w}}{\tilde{w}^T \mathcal{S}_W \tilde{w}}$$

The vector $\tilde{w}$ maximizing the function $R(\tilde{w})$ can be calculated by differentiating $R(\tilde{w})$ with respect to $\tilde{w}$ and then setting it to 0.

$$\frac{\mathrm{d}}{\mathrm{d}\tilde{w}} R(\tilde{w}) = \frac{2\mathcal{S}_B\tilde{w}(\tilde{w}^T\mathcal{S}_W\tilde{w}) - 2\mathcal{S}_W\tilde{w}(\tilde{w}^T\mathcal{S}_B\tilde{w})}{(\tilde{w}^T\mathcal{S}_W\tilde{w})^2} = 0 \tag{4.47}$$

Further simplification of (4.47) is mentioned below:

$$\mathcal{S}_B\tilde{w}(\tilde{w}^T\mathcal{S}_W\tilde{w}) = \mathcal{S}_W\tilde{w}(\tilde{w}^T\mathcal{S}_B\tilde{w}) \implies \mathcal{S}_B\tilde{w} = \lambda\mathcal{S}_W\tilde{w} \text{ where } \lambda = R(\tilde{w}) \tag{4.48}$$

If $d < n$, then $\mathcal{S}_W$ is an invertible matrix. (???) Therefore, when $d < n$

$$\mathcal{S}_W^{-1}\mathcal{S}_B\tilde{w} = \lambda\tilde{w}$$

Therefore, the direction $\tilde{w}$ of our interest is an eigen vector of the matrix $\mathcal{S}_W^{-1}\mathcal{S}_B$.

Claim: $\mathcal{S}_W^{-1}\mathcal{S}_B$ has exactly one non-zero eigen value. $\mathcal{S}_B = (\tilde{\mu}_0 - \tilde{\mu}_1)(\tilde{\mu}_0 - \tilde{\mu}_1)^T$ is product of $d \times 1$ vector and a $1 \times d$ vector. Therefore, $\mathcal{S}_B$ has rank one. Here we use the fact that the rank of a $p \times q$ at most $\min\{p, q\}$.

If $\mathbf{P}$ and $\mathbf{Q}$ are matrices of size $p \times q$ and $q \times r$ respectively then rank($\mathbf{PQ} \leqslant rank(\mathbf{P}))$. This result along with the fact that rank($\mathcal{S}_B$) = 1 implies that rank($\mathcal{S}_W^{-1}\mathcal{S}_B$) = 1.

The matrix $\mathcal{S}_W^{-1}\mathcal{S}_B$ therefore has only one eigen vector (unique upto scaling). Denote this eigen vector by $\tilde{v}$.

$$\mathcal{S}_W^{-1}(\tilde{\mu}_0 - \tilde{\mu}_1)(\tilde{\mu}_0 - \tilde{\mu}_1)^T \tilde{v} = \lambda \tilde{v}$$

The quantity $\tilde{\mu}_0 - \tilde{\mu}_1)^T \tilde{v}$ is a scalar. Call it "c".

$$\mathcal{S}_W^{-1}(\tilde{\mu}_0 - \tilde{\mu}_1)c = \lambda \tilde{v} \tag{4.49}$$

Since we are only interested in the direction of $\tilde{v}$ and not its magnitude, the following follows from (4.49)

$$\boxed{\mathcal{S}_W^{-1}(\tilde{\mu}_0 - \tilde{\mu}_1) \propto \tilde{v}} \tag{4.50}$$

Remark 4.4.

1. Observe that there is only one direction $\tilde{v}$ onto which our data has to be projected in order to minimize the variance of each class while separating the mean of the two class to the maximum extent.

2. It will be shown later that when the number of classes involved is k, then we would project data on $k - 1$ different vectors.

3. $\mathcal{S}_W^{-1}(\tilde{\mu}_0 - \tilde{\mu}_1)$ is estimated using the data we have in order to predict the direction $\tilde{v}$.

Chapter 5

Classification and Regression Trees

The following chapter is based on [Ize08], [BFOS48], [TG21], [Sha48], [Chi19], and [LM13]. The dataset has been taken from [DG17]

This chapter focuses on two specific **tree based methods** called **classification trees** and **regression trees**. Classification tree construction is first discussed in detail, after which regression trees are explained.

Classification tree, as the name suggests, is a supervised learning algorithm that helps classify unlabelled objects. Regression trees, on the other hand, predict the value of a continuous random value (i.e. response variable) based on a set of predictor variable values.

5.1 Classification trees

I will first explain what a classification tree is with the help of an example before mentioning the actual precise definition of a classification tree.

The word "tree" in classification tree implies that it has the structure of a mathematical tree that we know of from graph theory. Figure 5.1 gives the example of a classification tree which is trying to classify individual patients into one of two classes - "High risk" or "Low risk". Each node is associated with a predictor variable whose support is partitioned into two non-empty disjoint subsets. This partition corresponds to a binary split of the predictor variable corresponding to each node. For instance, the root node of the classification tree in 5.1 corresponds to the variable "Minimum systolic blood pressure over the initial 24 hours". Assuming that this variable is real valued, the root node is split based on the partition of its support as $(0, 91] \cup (91, 200]$. Finally, each terminal node is associated with a class label. Therefore, a unseen vector of data values corresponding to a patient can be "dropped down" the tree to see which terminal node it ends up in.

Example 5.1. Let "Sys" denote the minimum systolic blood pressure variable, "A" denote the age and "Sinus" be a binary random variable that takes values 0 or 1.

1 implies the presence of sinus tachycardia while 0 implies the absence the same condition. Let the data vectors observed be the realizations of the random vector $\tilde{V} = (Sys, A, Sinus)^T$.

- If the realization of the random vector $\tilde{V}$, for a given patient, is $(106, 45, 0)^T$, then he/she will be put under the "Low risk" category.
- However, the patient corresponding to the realization $(101, 70, 1)$ will be considered a "High risk" individual.

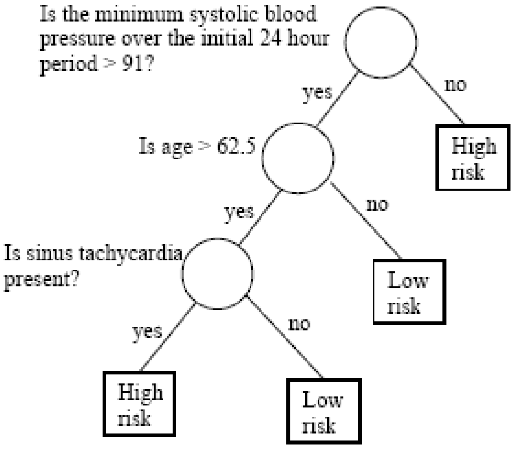

Is the minimum systolic blood pressure over the initial 24 hour period > 91?
yes
no
High risk
Is age > 62.5
yes
no
Low risk
Is sinus tachycardia present?
yes
no
High risk
Low risk

Figure 5.1: Example of a classification tree taken from [OR]

△

Remark 5.1. 1. There can be more than one terminal node with the same class label.

2. More than one non-terminal node can be split based on the same predictor variable.

5.1.1 Classification trees partition the feature space

Let Y denote the random variable corresponding to the class labels while $\tilde{X}$ is a $m \times 1$ random vector consisting of m predictor variables of interest. Then the feature space is defined as the support of the random vector $\tilde{X}$. In other words, it is all possible values that the vector $\tilde{X}$ can take.

Consider the following example tree before going back to the tree in 5.1 that we earlier used.

Example 5.2. Consider the classification tree mentioned in 5.2. The vector of predictor variables is a two dimensional vector $\tilde{X} = (X_1, X_2)^T$. Terminal nodes are shown in red while the non-terminal nodes are shown in blue. $C_1, C_2, \cdots C_5$ are assumed to be the five possible classes. Then the terminal nodes are called a **partition of data** because of the following reason.
Let $\mathcal{C}_i$ denote the set of all $\tilde{X}$ values for which $\tilde{X}$ belongs to $\mathcal{C}_i$. We assume that $\theta_1 < \theta_3$.

- $\mathcal{C}_1 = \{(X_1, X_2) \mid X_2 \leqslant \theta_1 \ \& \ X_1 \leqslant \theta_2\}$
- $\mathcal{C}_2 = \{(X_1, X_2) \mid X_2 \leqslant \theta_1 \ \& \ X_1 > \theta_2\}$
- $\mathcal{C}_3 = \{(X_1, X_2) \mid \theta_1 < X_2 \leqslant \theta_3 \ \& \ X_1 \leqslant \theta_4\}$
- $\mathcal{C}_4 = \{(X_1, X_2) \mid \theta_1 < X_2 \leqslant \theta_3 \ \& \ X_1 > \theta_4\}$
- $\mathcal{C}_5 = \{(X_1, X_2) \mid X_2 > \theta_3\}$

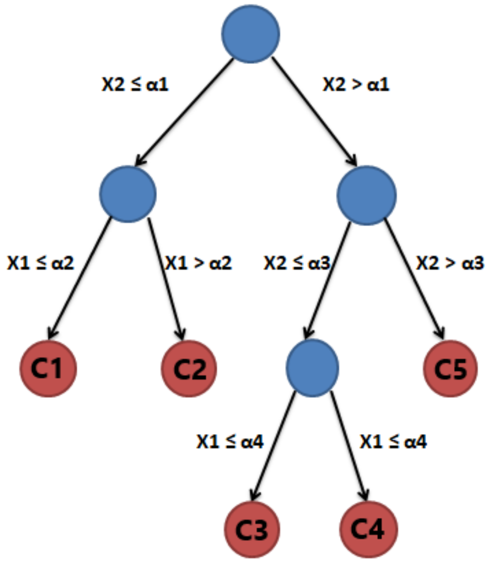


Figure 5.2: Example of a classification tree

Clearly, if the feature space is denoted by $Supp(\tilde{X})$, then by the way $\mathcal{C}_i$ has been defined for each $i = 1, \cdots, 5$, we have

$$Supp(\tilde{X}) = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3 \cup \mathcal{C}_4 \cup \mathcal{C}_5 \tag{5.1}$$

Pictorially, the partition of the the feature space $supp(\tilde{X})$ is as shown in Figure 5.3. Clearly, the decision tree shown in 5.2 partitions the feature space as shown in

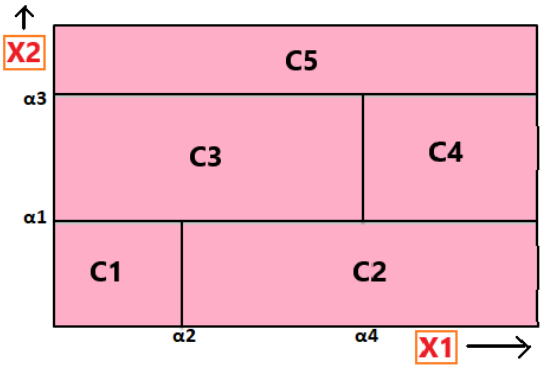


Figure 5.3: Partition of the feature space

5.3. Similarly, the decision tree 5.1, which has a 3-dimensional feature space, also partitions its own feature space into non-empty disjoint sets. △

Example 5.3. Coming back to the classification tree 5.1, the feature space in this scenario is 3-dimensional. The decision tree 5.1 leads to a partition of this 3-dimensional feature space as well. △

Remark 5.2. From the above examples, it clear that every classification tree induces a partition of the feature space $Supp(\tilde{X})$. For obvious reasons, the set of all terminal nodes are said to form a **partition of the data**.

Points to note:

- We only consider the case of binary splits at each non-terminal node. If the predictor variable X_i is continuous, then the node is split by partitioning the the support of X_i into two parts. This is pictorially shown in Figure 5.4.

- If X is a real valued random variable, then all $\tilde{X}$ for which the value of X is less than or equal to a real number v is sent to the left node while the rest of the observations are sent to the right node.

- Finally, if X is a discrete random variable that can take one of the L possible values in $\{1, 2, \cdots, L\}$. Then a splitting of the node is induced by the partition of $Supp(X)$ into any two non-empty subsets J and J'. All observations $\tilde{X}$ whose corresponding X value is one of those present in J will be sent to the left node while the remaining observations go to the right node.

The way the classification tree has been defined so far, it is clear that the classification tree is a **non-linear classifier** with **linear decision boundaries**. The remaining chapter will elaborate on how to construct a classification tree.

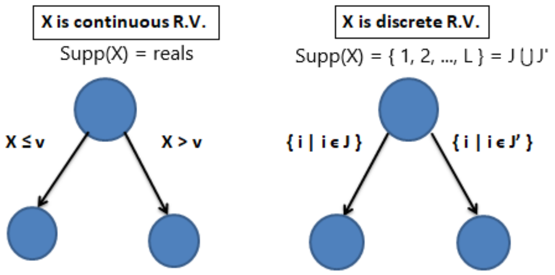
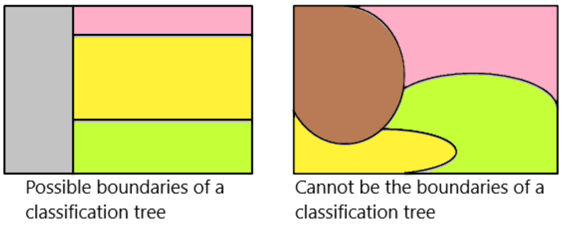


Figure 5.4: Partition of the feature space


Possible boundaries of a classification tree

Cannot be the boundaries of a classification tree

Figure 5.5

5.2 Definitions and notations

- $\mathcal{X}$ denotes the feature space i.e. the set of feature vectors. The vectors lying in $\mathcal{X}$ are denoted by the symbol $\tilde{x}$.

- Each observation vector $\tilde{x}$ is a $m \times 1$ vector i.e. we assume that there are m predictor variables of interest.

- $\mathcal{C}$ denotes the set of all possible class labels. Let the cardinality of this set be k i.e. there exist k distinct classes.

- Let $\mathbf{c} : \mathcal{X} \longrightarrow \mathcal{C}$ denote the ideal classifier.

- We assume that the number of observations noted is n. The set of all observed data vectors is denoted by $\mathcal{D}$.

- Each observation that we have is a **labelled** data point represented by the ordered pair $(\tilde{x}, \mathbf{c}(\tilde{x}))$. Here, the first co-ordinate represents the observed value while the second co-ordinate represents its true class label.

$$\mathcal{D} = \{(\tilde{x}_1, \mathbf{c}(\tilde{x}_1)), (\tilde{x}_2, \mathbf{c}(\tilde{x}_2)), \cdots, (\tilde{x}_n, \mathbf{c}(\tilde{x}_n))\}$$

- A classification tree is generally denoted by T while any node of this tree is represented by τ.

- We will use τ_r to represent the root node of the tree T.
- $\mathcal{X}(\tau)$ is the set of all elements from $\mathcal{X}$ that fall into the node τ.
  Consider for instance, the case of the continuous random variable X mentioned in 5.4. Denote the right and left daughter nodes by τ_R and τ_L respectively. Then,

$$\mathcal{X}(\tau_L) = \{\tilde{X} \mid X \leqslant v\} \; and \; \mathcal{X}(\tau_R) = \{\tilde{X} \mid X > v\}$$

- For a given node τ, $n(\tau)$ denotes the number of observations that fall into node τ.
- $n_i(\tau)$ represents the number of observations from node τ that fall into class i.

Our aim:

Our aim is to find a function $\mathbf{y} : \mathcal{X} \longrightarrow \mathcal{C}$, using the data set $\mathcal{D}$, such that $\mathbf{y}$ is not only a decision tree but also a “good” approximation for $\mathbf{c}$.

Definition 5.1 (**Splitting of feature space**)**.** A splitting of $\mathcal{X}$ is a partitioning of $\mathcal{X}$ into mutually exclusive non-empty subsets $\mathcal{X}_1, \mathcal{X}_2, \cdots, \mathcal{X}_s$ such that

$$\mathcal{X} = \bigcup_{i=1}^{s} \mathcal{X}_i \text{ and } \mathcal{X}_j \cap \mathcal{X}_t = \phi \text{ for all } j \neq t$$

Splitting of the feature space induces a splitting of the example set $\mathcal{D}$ into the sets $\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_s$ defined below.

$$\mathcal{D} = \bigcup_{j=1}^{s} \mathcal{D}_j \text{ where } \mathcal{D}_j = \{(\tilde{x}, \mathbf{c}(\tilde{x})) \mid \tilde{x} \in \mathcal{X}_j\}$$

Definition 5.2 (**Decision Tree**)**.** Let $\mathcal{X}$ be the feature space whose splitting is as given in Definition 5.1. If each $\mathcal{X}_i$ is assigned a class label, then the decision tree is called a **classification tree**. However, if each $\mathcal{X}_i$ is assigned a real number, then it is called a **regression tree**.
A tree T is called a classification tree for $\mathcal{X}$ and $\mathcal{C}$ if the following conditions hold.

1. T is a finite tree i.e. it has a finite number of nodes.
2. T is a rooted tree i.e. it has a single unique root node.
3. Every node τ is associated with a set of features $\mathcal{X}(\tau) \subset \mathcal{X}$.
4. A splitting criteria (w.r.t to a predictor variable) exists at each non-terminal node of the tree while each of its terminal node is assigned a class label or a real number depending on whether T is a classification or a regression tree.

Definition 5.3. A partition of the feature space $\mathcal{X}$ induces a partition of the set of observations $\mathcal{D}$. We denote by $\mathcal{D}(\tau)$ the set of all observations in $\mathcal{D}$ that fall into the node τ.

$$\mathcal{D}(\tau) = \{(\tilde{x}, \mathbf{c}(\tilde{x})) \in \mathcal{D} \mid \tilde{x} \in \mathcal{X}\}$$

Remark 5.3.

- Note that we only consider binary splits at each node. At each non-terminal node, only one feature is measured.
- Every feature vector in $\mathcal{X}$ is associated with a unique path from the root of the tree to the terminal nodes.

Since the definition of a decision tree is clear, the next step is to understand the construction of classification trees and regression trees. In the case of a classification tree, we are trying to predict a categorical variable Y using a vector of predictor variables $\tilde{X}$. However, in the case of a regression tree, the response variable that we wish to predict using $\tilde{X}$ is a continuous random variable. Note that the construction of a classification tree is explained before that of a regression tree.

5.3 Node Splitting

The questions that come to our mind when we go through the definition of a decision tree are: "How do we split a node?", "Which split is the best split for a given node? How do we define best?" etc. In this section, we first discuss about the number of all possible splits a given node can have. After that, the procedure to select the "best" split is elaborated in detail.

5.3.1 Number of possible splits

Let us first consider the root node τ_r. What are all the possible splits at this node τ_r? This is the first question that we are going to deal with in this section. Predictor variables involved are $X_1, X_2, \cdots, X_m$. Each of these variables can be any one of continuous, discrete or categorical. Let the variable X be the variable being used to split a node τ. We use the data set iris to make it easier to understand.

```
> data("iris")
> iris
> dim(iris)
 [1] 150   5
> head(iris)
      Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4          0.2   setosa
2          4.9         3.0          1.4          0.2   setosa
```

```
3          4.7          3.2          1.3          0.2  setosa
4          4.6          3.1          1.5          0.2  setosa
5          5.0          3.6          1.4          0.2  setosa
6          5.4          3.9          1.7          0.4  setosa
```

Case 1 - X is continuous:

I will use the iris dataset to explain how this case is dealt with. Consider the variable "Petal width" for instance. Petal width is clearly a contiuous random variable. We have, in the iris data set, 150 observations on petal width.

```
  > iris$Petal.Width
  [1] 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 0.2 0.2 0.1 0.1 0.2
 [16] 0.4 0.4 0.3 0.3 0.3 0.2 0.4 0.2 0.5 0.2 0.2 0.4 0.2 0.2 0.2
 [31] 0.2 0.4 0.1 0.2 0.2 0.2 0.2 0.1 0.2 0.2 0.3 0.3 0.2 0.6 0.4
 [46] 0.3 0.2 0.2 0.2 0.2 1.4 1.5 1.5 1.3 1.5 1.3 1.6 1.0 1.3 1.4
 [61] 1.0 1.5 1.0 1.4 1.3 1.4 1.5 1.0 1.5 1.1 1.8 1.3 1.5 1.2 1.3
 [76] 1.4 1.4 1.7 1.5 1.0 1.1 1.0 1.2 1.6 1.5 1.6 1.5 1.3 1.3 1.3
 [91] 1.2 1.4 1.2 1.0 1.3 1.2 1.3 1.3 1.1 1.3 2.5 1.9 2.1 1.8 2.2
[106] 2.1 1.7 1.8 1.8 2.5 2.0 1.9 2.1 2.0 2.4 2.3 1.8 2.2 2.3 1.5
[121] 2.3 2.0 2.0 1.8 2.1 1.8 1.8 1.8 2.1 1.6 1.9 2.0 2.2 1.5 1.4
[136] 2.3 2.4 1.8 1.8 2.1 2.4 2.3 1.9 2.3 2.5 2.3 1.9 2.0 2.3 1.8
```

For a given splitting condition $X \leqslant v$, the above values of petal widths will be divided into two parts. Since v can be any real value, it appears as if there are infinitely many possible splitting criterion while using the variable X. However, the number of observations, n, and the number of 2 set partitions of the petal width values is finite. Therefore, it implies that the number of possible splits at the node with respect to the variable X is finite as well.

Instead of looking at all the infinite possible splitting criteria, it is enough to look at splitting criteria of the form $X \leqslant v$ where v is an observed petal width value. The number of distinct values that v takes is found out as shown below.

```
 > petal.fac = as.factor(Petal.Width)
 > levels(petal.fac)
 [1] "0.1" "0.2" "0.3" "0.4" "0.5" "0.6" "1"   "1.1" "1.2" "1.3"
[11] "1.4" "1.5" "1.6" "1.7" "1.8" "1.9" "2"   "2.1" "2.2" "2.3"
[21] "2.4" "2.5"
```

This implies that although there are 150 independent observations noted for the petal width, the observation values are all one of 22 distinct values. This implies that at the root node, there are 22 possible splits with respect to the variable "Petal.Width".

If the node we are working with is a non-terminal node τ, then instead of working with $\mathcal{D}(\tau_r) = \mathcal{D}$, we work with $\mathcal{D}(\tau)$ and follow the procedure mentioned above.

Case 2 - X is categorical:

Let us again consider the dataset "iris". The variable "Species" is categorical. The three levels of this variable are "setosa", "versicolor" and "virginica".

```
 > levels(as.factor(Species))
[1] "setosa"   "versicolor"   "virginica"
```

Then the number of possible splits at a given node, with respect to this variable "Species" is equal to the number of 2-set partitions of the set of all possible labels. If $L = \{$ setosa , versicolor , virginica $\}$. If L is partitioned into sets L_1 and L_2, then all the observations whose "Species" variable is one of the elements of L_1 will go to one daughter node while the rest go to the other daughter node. All possible partitions of L is given in the table below (unique upto swapping roles of L_1 and L_2).

L_1	L_2
{ setosa }	{ versicolor , virginica }
{ versicolor }	{ setosa , virginica }
{ virginica }	{ setosa , versicolor }

Consider the general case where X can take one of L possible labels. Note that, L stands for number of class labels. However, in the example earlier L represents the set of all possible class labels.

The number of possible splits in the general scenario is equivalent to finding the number of possible 2-set partitions of the set of all possible class labels. It is easy to derive that the number of such 2-set partitions is

$$\#\ \text{2-set partitions } = 2^{L-1} - 1$$

Clearly, in this scenario, the same procedure is used to determine the number of possible splits for both a terminal and a non-terminal node.

5.3.2 Determining the best split using impurity measures

In the previous subsection, we discussed about the number of possible splits at each node with respect to a given variable. Now we answer the question of how to identify the best split among all possible splittings at a given node. This problem is solved with the help of a function called **impurity function**. Impurity functions are quantitative measures of how "homogeneous" or "uniform" given node/nodes are.

Definition 5.4. Let $p_0, p_1, \cdots, p_k$ be points from a Euclidean space $E = \mathbb{R}^d$ for some $d \in \mathbb{N}$. Then the **k-simplex** in E with vertices $\{p_0, p_1, \cdots, p_k\}$ is the convex set spanned by $p_0, p_1, \cdots, p_k$.

Notation: $\Delta^k = \left\{(\alpha_0, \alpha_1, \cdots, \alpha_k) \in \mathbb{R}^{k+1} \mid \sum_{i=0}^k \alpha_i = 1 \text{ where } \alpha_i \geqslant 0 \forall i\right\}$

Consider a k-class problem. Then the impurity function, i, is a real-valued function defined on Δ^{k-1}. The variables involved are $p_1, p_2, \cdots, p_k$. The impurity of a node τ is given by $i(p_1, p_2, \cdots, p_k)$ where p_i, for each given i, denotes the proportion of observations in τ that belong to the i^{th} class.

$$i(\tau) = i(p_1, p_2, \cdots, p_k) \text{ where } p_i = \frac{n_i(\tau)}{n(\tau)}$$

Definition 5.5. For a k-class problem, the impurity function $i : \Delta^{k-1} \longrightarrow \mathbb{R}$ is a function that follows the following criteria.

1. The function i is minimized at unit vectors $e_1, e_2, \cdots, e_k$ where e_i is the unit vector with 1 at the i^{th} position and 0 at the remaining positions.

2. i is symmetric with respect to its arguments p_i i.e.

$$i(p_1, p_2, \cdots, p_k) = i(p_{\sigma(1)}, p_{\sigma(2)}, \cdots, p_{\sigma(k)}) \text{ for all } \sigma \in \mathcal{S}_k$$

3. i is maximized at $(\frac{1}{k}, \frac{1}{k}, \cdots, \frac{1}{k})$

It is important to understand why the impurity function i has to satisfy all the three conditions mentioned in the definition above.

Evaluating impurity at e_i implies that we are quantifying how "impure" the associated node is. The vector e_i as input implies that all the observations that have fallen into this node are from the i^{th} class. Since such node is "pure" or homogeneous, we would want our impurity function to be minimized at all such $e_i's$.

The reason why i must be symmetric can be understood with the help of an example. Assume that we have two object A and B.

Object A - 30% gold + 70% copper

Object B - 30% copper + 70% gold

Clearly, if the input vector for object A is $(p_1, p_2) = (0.3, 0.7)$ then the input vector for object B is $(0.7, 0.3)$ where p_1 is the proportion of metal in object while p_2 is proportion of copper. Clearly, irrespective of composition, the "purity" of both objects is same. Therefore, the impurity function is assumed to be symmetric.

Finally, i is supposed to be maximum at $(\frac{1}{k}, \frac{1}{k}, \cdots, \frac{1}{k})$ since the equally likely scenario is the most impure or uncertain scenarios that we can think of.

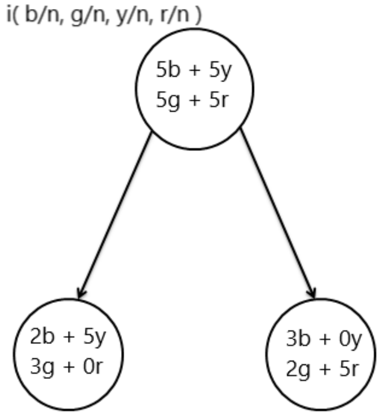


Figure 5.6: Example to understand impurity function

Example 5.4. Consider the following simple tree to understand how impurity function parameters are evaluated. Let the root node of Figure 5.6 be denoted by τ while the left and right daughter nodes are denoted by τ_L and τ_R respectively. Let b, y, g and r represent blue, yellow, green and red balls. Since the node consists of 20 balls with 5 balls of each colour. If the vector (p_1, p_2, p_3, p_4) are the papramEters required for impurity function, then the parameters are the proportion of blue, green, yellow and red balls respectively. Therefore,

- $i(\tau) = i(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$
- $i(\tau_R) = i(\frac{3}{10}, \frac{2}{10}, \frac{5}{10}, 0)$
- $i(\tau_L) = i(\frac{2}{10}, \frac{3}{10}, 0, \frac{5}{10})$

△

Now that we already know what an impurity function is, it is time to know about the measure (based on impurity) that is used to determine the best split. This measure is called **impurity reduction**.

Definition 5.6. Let τ denote the root node of a tree T while the terminal nodes of this tree given by $\{\tau_1, \tau_2, \cdots, \tau_L\}$. Then the impurity reduction, $\Delta_i(\tau, \{\tau_1, \tau_2, \cdots, \tau_L\})$, is given by

$$\Delta_i(\tau, \{\tau_1, \tau_2, \cdots, \tau_L\}) = i(\tau) - \sum_{j=1}^{k} \frac{n(\tau_j)}{n(\tau)} \times i(\tau_j) \tag{5.2}$$

The Definition 5.2 is the difference between the impurity of the root node and the weighted average of the impurities of terminal nodes. In other words, it is the reduction in impurity observed after splitting the root node to give the terminal nodes $\tau_1, \cdots, \tau_L$.

Possible impurity functions

The two types of impurity functions that we will focus on are

1. Impurity based on miss-classification error. This impurity is defined as given in 5.3. Each p_i denotes the proportion of observations in a given node that belong to class i. Here, we assume that the class label that is highest in proportion is the true/correct label while all the other observations (in the node) are miss-classified.

$$i_{miscl}(p_1, p_2, \cdots, p_k) = 1 - \max\{p_1, p_2, \cdots, p_k\} \tag{5.3}$$

2. Impurity based on entropy is given by the following formula. Note that p_i have the usual expected meaning defined earlier.

$$i_H(p_1, p_2, \cdots, p_k) = -\sum_{i=1}^{n} p_i \log(p_i) \tag{5.4}$$

   The motivation behind this function is discussed in the following section.

Before delving further into the intuition behind these impurity functions, let us understand how these impurity functions are used to determine the best split at a given node. Let us start with how the root node can be split.

Splitting the root node

Let $\tilde{X}_1, \tilde{X}_2, \cdots, \tilde{X}_n$ be n observations with each observation belonging to one of the k classes. If each $\tilde{X}_i$ is d-dimensional, it implies that there exist d variables, denoted by $V_1, V_2, \cdots, V_d$. Consider the root node given in Figure 5.7. All the input elements can be assumed to lie in the root node. Our aim is to find the best possible V_i and v to split the node. The root node in Figure 5.7 is denoted by τ while the left and right daughter nodes are denoted by τ_L and τ_R respectively.

We already know that the number of possible splits at a given node is finite with respect to any continuous/discrete predictor variable. The following steps must be followed to choose the best split.

1. Start with variable V_1. Let there be s_1 possible splits with respect to this variable. For each of these s_1 splits, calculate the corresponding impurity reduction. The split with maximum impurity reduction is considered the best split with respect to variable V_1.

2. Repeat the above procedure for each of the d variables and note down the best split, along with its impurity reduction, for each variable V_i. Let the best split at τ with respect to V_i be denoted by $\mathcal{S}_i$.

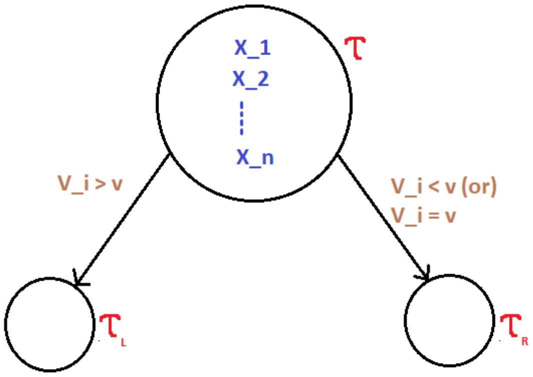


Figure 5.7: Splitting the root node

3. Among all the splits $\mathcal{S}_i$, pick the slit which corresponds to maximum impurity reduction. This split is considered the best split at the root node.

Remark 5.4. The same procedure can be followed for other non-terminal root nodes as well. The only difference will be the number of observations $\tilde{X}_i$ associated with the node under consideration.

5.3.3 Entropy as an impurity function

The motivation behind the impurity function i_{miscl} is clear. However, using the functional form of entropy as an impurity function is not intuitive. This subsection briefly explains how entropy function was introduced and what role it plays in node splitting.

The concept of entropy was first introduced by the mathematician Claude Shannon in a paper he wrote on communication systems. In this paper, he was attempting to quantify the amount of information produced by a discrete information source. (A discrete information source is a source that produces a discrete sequence of symbols.) The following is a brief and simplified summary of how Shannon arrived at the entropy function in this paper.

Let X be a discrete random variable with n possible outcomes given by $x_1, x_2, \cdots, x_n$. Denote the probability $\mathcal{P}(X = x_i) = p_i$ for all $i \in \{1, 2, \cdots, n\}$. If $H : \Delta^{k-1} \longrightarrow \mathbb{R}$ is a function on $(p_1, p_2, \cdots, p_n)$ such that it measures the uncertainty associated with a random variable X. Such a H, according to Shannon, must follow the properties mentioned below.

1. H is a continuous function on the entire domain.

2. If $p_1 = p_2 = \cdots = p_n = \frac{1}{n}$, then H must be a monotonically increasing in n.

3. H must satisfy a certain law called the "composition law" which has been elaborated later.

It is intuitively clear as to why the first two points have to be satisfied. The composition law mentioned has been given in detail below.

Let the possible outcomes be written as a disjoint union of M different sets $\mathcal{C}_i$ i.e. $\{x_1, x_2, \cdots, x_n\} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \cdots \cup \mathcal{C}_M$.

$$\text{Let } \mathcal{C}_i = \{c_1^{(i)}, c_2^{(i)}, \cdots, c_{r_i}^{(i)}\} \text{ i.e } |\mathcal{C}_i| = r_i \forall i$$

Let $p_j^{(i)}$ for all $j \in \{1, 2, \cdots, r_i\}$ denote the probability $\mathcal{P}(X = c_j^{(i)})$ for all j. Therefore $(p_1^{(i)}, p_2^{(i)}, \cdots, p_{r_i}^{(i)})$ is the probability vector associated with events in $\mathcal{C}_i$.

Let $d_l^{(i)}$ denote the probability of the event $C_l^{(i)}$ occurring given that one of the events in $\mathcal{C}_i$ has already occurred.

$$\text{Let } \tilde{d}_i = (d_1^{(i)}, d_2^{(i)}, \cdots, d_{r_i}^{(i)}) \text{ where } d_l^{(i)} = \mathcal{P}(c_l^{(i)} \text{ occurred} \mid \text{one of events in } \mathcal{C}_i \text{ occurred})$$

Using the notations given so far, the following can be derived

1. $0 \leqslant d_l^{(i)} \leqslant 1$ for all possible i, l values.
2. $\sum_{l=1}^{n} d_l^{(i)} = 1$ for all possible i.
3. $p_l^{(i)} = d_l^{(i)} \times z_i$ where $z_i = p_1^{(i)} + p_2^{(i)} + \cdots + p_{r_i}^{(i)}$

The following rule, which Shannon assumed H must follow, is called the composition rule.

$$H(p_1, p_2, \cdots, p_n) = H(z_1, z_2, \cdots, z_n) + \sum_{j=1}^{M} z_j H(d_1^{(j)}, d_2^{(j)}, \cdots, d_{r_j}^{(j)}) \tag{5.5}$$

In the (5.5) rule given above is the mathematical formalization of: "Uncertainty associated with the outcome of X is equal to the sum of uncertainty associated in specifying which set $\mathcal{C}_i$ the observation belongs to and uncertainty in specifying the specific outcome (within $\mathcal{C}_i$) the observation is."

- $H(z_1, z_2, \cdots, z_n)$ is the term that quantifies the uncertainty in specifying which set $\mathcal{C}_i$ the outcome belongs to.
- The second term in (5.5), that is written under summation, quantifies the uncertainty in specifying which $c_j^{(i)}$ we are considering in $\mathcal{C}_i$.

Shannon proved that the only function H that satisfies the conditions given in 5.3.3 is

$$H(p_1, p_2, \cdots, p_n) = -\kappa \sum_{i=1}^{n} p_i \log p_i \quad \text{for some } \kappa > 0 \tag{5.6}$$

The functional form in (5.6) is therefore a function that can quantify the amount of uncertainty associated with the outcome of a discrete random variable X.

For a continuous random variable X, the entropy is given by

$$H(x) = \int_{-\infty}^{\infty} p(x) \log p(x) \text{ where } p(x) \text{ is the pdf of} X \tag{5.7}$$

What does reduction in impurity imply when calculated with respect to entropy function?

It is important to understand the entropy function further in order to answer this question.

Remark 5.5. Observe that the entropy of X, irrespective of whether it is discrete or continuous, can be re-written as follows:

$$H(p) \text{ or } H(X) = -\mathbb{E}_X[\log(p(X))]$$

Definition 5.7. Let $(X, Y)^T$ be a bi-variate random vector with pmf $p(x, y)$. Then its **joint entropy**, H(X,Y), is given by

$$H(X, Y) = -\mathbb{E}_{X,Y}[\log(p(X, Y))] = -\int_{\mathcal{Y}} \int_{\mathcal{X}} p(x, y) \log p(x, y) \mathrm{d}x \mathrm{d}y \tag{5.8}$$

Note that the integrals in (5.8) can be interchanged due to the Fubini-Tonelli's theorem.

Definition 5.8. The **conditional entropy** of X given Y is

$$H(Y|X) = -\mathbb{E}_{X,Y}[log(p(Y|X))] \tag{5.9}$$

Using the definition of conditional density, (5.9) can be simplified as follows:

$$H(Y|X) = -\int_{\mathcal{X}} p(x) H(Y|X = x) \mathrm{d}x \;\; \text{where} \;\; H(Y|X = x) = -\int_{\mathcal{Y}} p(y|x) \log(y|x) \mathrm{d}y \tag{5.10}$$

Entropy $H(Y|X)$ measures the uncertainty associated with experiment Y given tat X has already been conducted. The result of experiment X is unknown. However, in the case of $H(Y|X = x)$, the result of experiment X is known to be x.

Definition 5.9. Let X and Y be continuous variables with joint pdf denoted by $p(x, y)$ while marginal densities denoted by $p(x)$ and $p(y)$ respectively. Then the **mutual information** between X and Y is given by

$$I(X; Y) = \mathbb{E}_{X,Y}\left[\log\left(\frac{p(X, Y)}{p(X)p(Y)}\right)\right] \tag{5.11}$$

Mutual information is a measure of the amount of information one variable has about the other. In other words, it is a measure of the reduction in uncertainty regarding one variable due to the knowledge of another. Using the definition of Kullback–Leibler divergence, denoted by $\mathcal{D}(\cdot||\cdot)$, it can be shown that

$$I(X,Y) = \mathcal{D}\left(p(x,y)||p(x)p(y)\right) \tag{5.12}$$

Since $\mathcal{D}(p_1||p_2)$ of two densities p_1 and p_2 is a measure of how different the two densities are. In other words, it quantifies information lost when p_2 is used to approximate p_1. Therefore, $I(X,Y)$ is a measure of how "far away" X and Y are from being independent.

Few properties of the function I(X,Y)

The proofs of these results can be found in the text book by Thomas and Cover.

1. $I(X,Y) \geqslant 0$ follows from the theorem that the KL-divergence function is always non-negative.
2. $I(X,Y) = 0$ iff $X \perp Y$
3. $I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$

Now we consider the important question that we posed earlier i.e. what does the impurity reduction with respect to entropy quantify? With the help of an example, it will be shown that impurity reduction with respect to entropy function is equivalent to calculating $H(X) - H(X|Y)$ (or $H(Y) - H(Y|X)$) which is nothing but the calculation of $I(X,Y)$ according to the properties given above. Observe that the calculation $H(X) - H(X|Y)$ (or $H(Y) - H(Y|X)$) is nothing but the reduction in uncertainty associated with X after the experiment Y has occurred.

Example 5.5. Consider the example shown in Figure 5.8. Assume the existence of two classes C_1 and C_2. The number of observations from each class is given in the figure. Then the reduction in impurity for this triple is given by:

$$i(\tau) - p_L i(\tau_L) - p_R i(\tau_R)$$

Then, $i(\tau)$ is given by:

$$i_H(\tau) = -\frac{40}{100}\log\left(\frac{40}{100}\right) - \frac{60}{100}\log\left(\frac{60}{100}\right) \tag{5.13}$$

Define the following random variable:

$$X_\tau = \begin{cases} 1, & \text{with prob. } 0.4 \\ 2, & \text{with prob. } 0.6 \end{cases} \tag{5.14}$$

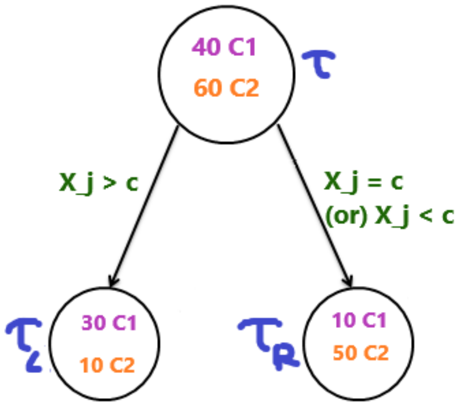


Figure 5.8: Use of entropy as impurity function

Using (5.13) and (5.14), it is clear that

$$\boxed{i_H(\tau) = H(X_\tau)} \tag{5.15}$$

Now define a random variable Y as follows:

$$Y = \begin{cases} 1, & \text{with prob. } p_L = \frac{4}{10} \text{ when } X_j > c \\ 0, & \text{with prob. } p_R = \frac{6}{10} \text{ when } X_j \leqslant c \end{cases} \tag{5.16}$$

Now the impurity associated with the two daughter nodes is given by:

$$i_H(\tau_L) = -\frac{30}{40}\log\left(\frac{30}{40}\right) - \frac{10}{40}\log\left(\frac{10}{40}\right) \tag{5.17}$$

Clearly, the function in (5.17) is equivalent to $P(Y = 0)H(X_\tau|Y = 0) + P(Y = 1)H(X_\tau|Y = 1)$. Therefore the weighted average of the daughter node impurities is equivalent to

$$p_L i_H(\tau_L) + p_R i_H(\tau_R) = H(X_\tau|Y) \tag{5.18}$$

Using (5.15) and (5.18), it is clear that the calculation of impurity reduction for a given triad of nodes is equivalent to calculating $I(X, Y) = H(X) - H(X|Y)$. △

Simple and careful manipulations of the impurity function in general can be used to prove the general result.

5.3.4 Entropy based impurity vs Mis-classification based impurity

We have seen two impurity functions so far - the impurity based on mis-classification error and the impurity based on entropy. While the former impurity function is more intuitive, the latter impurity function has some nice mathematical properties that are desirable for impurity functions.

Definition 5.10. A function $f : D \longrightarrow \mathbb{R}$ is a **concave function** iff

$$f(\sum_{i=1}^{n} \lambda_i x_i) \geqslant \sum_{i=1}^{n} \lambda_i f(x_i) \text{ when } \sum_{i=1}^{n} \lambda_i \tag{5.19}$$

If the inequality in (5.19) is strict for all $x_1, x_2, \cdots, x_n \in D$, then the function f is called **strictly concave**.

It is desirable to have a strictly concave impurity function as it strict concavity is equivalent to non-zero impurity reduction at each step.

Lemma 5.1. *Strictly concave impurity function gives a strictly positive impurity reduction.*

Proof. Let τ denote the parent node while $\tau_1, \tau_2, \cdots, \tau_s$ denote the terminal nodes of the decision tree T. D, as mentioned earlier, denotes the set of all observations. Let D_j denote the number of observations of D that fall into the j^{th} terminal node of T. Let C_j be the set of observations of D that belong to the j^{th} class while $C_i^{(j)}$ denotes the observations in D_j that belong to class i.

$$i(\tau) = i\left(\frac{|C_1|}{D}, \frac{|C_2|}{D}, \cdots, \frac{|C_k|}{D}\right)$$

Each individual term $\frac{|C_j|}{|D|}$ can be simplified further as follows:

$$\frac{|C_i|}{|D|} = \frac{|D_j|}{|D|} \times \frac{|C_i|}{|D_j|} = \frac{|D_j|}{|D|} \times \left(\frac{|C_1^{(i)}| + |C_2^{(i)}| + \cdots + |C_k^{(i)}|}{|D_j|}\right) \tag{5.20}$$

Using (5.20), we can prove that:

$$\frac{|C_i|}{|D|} = \sum_{l=1}^{s} \frac{|D_l|}{|D|} \times \frac{|C_l^{(i)}|}{|D_l|}$$

Now, the weighted impurity term associated with the j^{th} terminal node is given by:

$$\frac{|D_j|}{|D|} \times i(D_j) = \frac{|D_j|}{|D|} \times i\left(\frac{|C_1^{(j)}|}{|D_j|}, \frac{|C_2^{(j)}|}{|D_j|}, \cdots, \frac{|C_k^{(j)}|}{|D_j|}\right)$$

Using the manipulations done so far, it is easy to deduce the following:

$$i(\tau) = i\left(\sum_{l=1}^{s} \frac{|D_l|}{|D|} \times \frac{|C_l^{(1)}|}{|D_l|}, \cdots, \sum_{l=1}^{s} \frac{|D_l|}{|D|} \times \frac{|C_l^{(k)}|}{|D_l|}\right) \tag{5.21}$$

$$= i\left(\sum_{l=1}^{s}\frac{|D_l|}{|D|}\left(\frac{|C_1^{(l)|}}{|D_l|},\cdots,\frac{|C_k^{(l)|}}{|D_l|}\right)\right) \tag{5.22}$$

$$> \sum_{l=1}^{s}\frac{|D_l|}{|D|}i\left(\frac{|C_1^{(l)|}}{|D_l|},\cdots,\frac{|C_k^{(l)|}}{|D_l|}\right) \text{ if } i \text{ is strictly concave} \tag{5.23}$$

It is easy to show that (5.21) is equivalent to a strictly positive reduction in impurity. Hence a strictly concave impurity is desirable. □

Lemma 5.2. *The impurity function based on miss-classification is concave but not strictly concave. However, the impurity based on entropy is strictly concave.*

Proof. Basic undergraduate mathematics can be used to prove this lemma. □

5.3.5 Assigning a class label to the terminal node

Although we haven't yet discussed the complete construction of the decision tree, it is useful to know how a class label is assigned to terminal nodes. Let the terminal node τ have $n(\tau)$ observations in it. Let $n_j(\tau)$ be the number of observations in τ that belong to class j where $j \in \{1, 2, \cdots, k\}$. Let τ_{root} denote the root node. Then the class label (y_i) given to τ_i is:

$$\boxed{y_i = \operatorname*{argmax}_{j=1}^{k}\ [n_j(\tau_j)]} \tag{5.24}$$

The above rule (5.24) can be seen as an estimate of the Baye's rule. If $\mathcal{C}_i$ denotes the set of class i elements, then

$$\mathcal{P}(\tilde{x} \in \mathcal{C}_i|\tilde{x} \in \tau) = \frac{\mathcal{P}(\tilde{x} \in \tau|\tilde{x} \in \mathcal{C}_i) \times \Pi_i}{\mathcal{P}(\tilde{x} \in \tau)} \text{ where } \Pi_i = \mathcal{P}(\tilde{x} \in \mathcal{C}_i) \tag{5.25}$$

The prior probability Π_i and $\mathcal{P}(\tilde{x} \in \tau|\tilde{x} \in \mathcal{C}_i)$ in (5.25) are estimated as follows:

$$\widehat{\Pi}_i = \frac{n_i(\tau)}{n(\tau_{root})} \text{ and } \widehat{\mathcal{P}}(\tilde{x} \in \tau|\tilde{x} \in \mathcal{C}_i) = \frac{n_i(\tau_{root})}{n(\tau_{root})} \tag{5.26}$$

Using (5.25), (5.26) along with the fact that $\mathcal{P}(\tilde{x} \in \tau)$ is constant gives the following estimate for $\mathcal{P}(\tilde{x} \in \mathcal{C}_i|\tilde{x} \in \tau)$:

$$y_i = \operatorname*{argmax}_{i=1}^{k}\mathcal{P}(\tilde{x} \in \mathcal{C}_i|\tilde{x} \in \tau) = \operatorname*{argmax}_{i=1}^{k}\widehat{\mathcal{P}}(\tilde{x} \in \tau|\tilde{x} \in \mathcal{C}_i) \times \widehat{\Pi}_i = \operatorname*{argmax}_{j=1}^{k}\ [n_j(\tau_j)]$$

Therefore, the rule described to assign class label to a terminal node can be derived using the Bayes classifier.

5.4 Finding the tree that best describes the data

So far we have discussed how nodes can be split in order to "grow" a tree. The question that this section will deal with is about how to find the best possible tree that describes our data. This means that we must answer the question - How do we decide when to stop splitting?

Building a tree by sequentially splitting the nodes of a tree is called **recursive partitioning**. If a binary tree is grown till none of the nodes can be split, then the resulting tree is said to be **saturated**. The overall idea is to grow the tree to saturation and then **pruning the tree** or chopping off "unecessary" branches until the "best" sub-tree is obtained.

5.4.1 Notations

Let T be a tree with L terminal nodes denoted by $\tau_1, \cdots, \tau_L$. Let the true proportion of objects that get miss-classified in terminal node τ_i be denoted by $\mathcal{R}(\tau_i)$. Since this true value is mostly never known to us, it's estimator $r(\tau_i)$ is given by:

$$r(\tau_i) = 1 - \max_{j=1}^{k} \widehat{p}(j|\tau_i) \text{ where } \widehat{p}(j|\tau_i) = \frac{n_j(\tau_i)}{n(\tau_j)} \tag{5.27}$$

Remark 5.6. The $r(\tau_i)$ has the same functional form as the impurity based on miss-classification. Although the functional forms are the same, the impurity function has no role to play in estimating $\mathcal{R}(\tau_i)$.

Let all the terminal nodes of the tree T lie in the set $\tilde{T}$ i.e.

$$\tilde{T} = \{\tau_1, \tau_2, \cdots, \tau_L\}$$

Then the miss-classification rate of the tree T, denoted by $R(T)$, is given by the weighted average of all the $R(\tau_i)$ values.

$$\mathcal{R}(T) = \sum_{l=1}^{L} \mathcal{R}(\tau_L)\mathcal{P}(\tau_L) \text{ where } \mathcal{P}(\tau_L) = \text{ prob. that obs. falls into } \tau_l \tag{5.28}$$

$\mathcal{R}^{re}(T)$, called as the **re-substitution error**, is a possible estimator for $\mathcal{R}(T)$ defined above. Using the estimates $r(\tau_l)$ and $\widehat{p}(\tau_l) = \frac{n(\tau_l)}{n(\tau_{root})}$ for $\mathcal{R}(T)$ and $\mathcal{P}(\tau_l)$ respectively gives us:

$$\mathcal{R}^{re}(T) = \sum_{l=1}^{L} r(\tau_l) \times \widehat{p}(\tau_l) = \sum_{l=1}^{L} r(\tau_l) \times \frac{n(\tau_l)}{n(\tau_{root})} = \sum_{l=1}^{L} \mathcal{R}^{re}(\tau_l) \text{ where } \mathcal{R}^{re}(\tau_l) = r(\tau_l) \cdot \frac{n(\tau_l)}{n(\tau_{root})}$$

Lemma 5.3. *Let T be a tree with set of terminal nodes $\tilde{T} = \{\tau_1, \tau_2, \cdots, \tau_{L-1}, \tau_L\}$. T' is a different tree with the terminals nodes $\tilde{T}' = \{\tau_1, \tau_2, \cdots, \tau_{L-1}, \tau_{L_1}, \tau_{L_2}\}$. In other* *words, the tree T' is obtained from tree T by splitting (without loss of generality) the node τ_L into τ_{L_1} and τ_{L_2}. Then,*

$$\boxed{\mathcal{R}^{re}(T') \leqslant \mathcal{R}^{re}(T)}$$

Proof.

$$\mathcal{R}^{re}(T) = \left\{ \sum_{l=1}^{L-1} r(\tau_l)\widehat{p}(\tau_l) \right\} + r(\tau_L)\widehat{p}(\tau_L) \tag{5.29}$$

$$\mathcal{R}^{re}(T') = \left\{ \sum_{l=1}^{L-1} r(\tau_l)\widehat{p}(\tau_l) \right\} + r(\tau_{L_1})\widehat{p}(\tau_{L_1}) + r(\tau_{L_2})\widehat{p}(\tau_{L_2}) \tag{5.30}$$

Comparing $\mathcal{R}^{re}(T)$ and $\mathcal{R}^{re}(T')$ is therefore equivalent to comparing the terms $r(\tau_L)\widehat{p}(\tau_L)$ and $r(\tau_{L_1})\widehat{p}(\tau_{L_1}) + r(\tau_{L_2})\widehat{p}(\tau_{L_2})$ respectively. Let θ be the number of observations in τ_{L_2} while n denotes the total number of observations.‘

$$\begin{aligned}
\sum_{i=1}^{2} \mathcal{R}^{re}(\tau_{L_i}) &= r(\tau_{L_1})\widehat{p}(\tau_{L_1}) + r(\tau_{L_2})\widehat{p}(\tau_{L_2}) \\
&= r(\tau_{L_1}) \cdot \left[\frac{n(\tau_L) - \theta}{n}\right] + r(\tau_{L_2}) \cdot \left[\frac{\theta}{n}\right] \\
&= r\left(\frac{|C_1^{(1)}|}{n(\tau_L) - \theta}, \cdots, \frac{|C_k^{(1)}|}{n(\tau_L) - \theta}\right) \cdot \left[\frac{n(\tau) - \theta}{n}\right] + r\left(\frac{|C_1^{(2)}|}{\theta}, \cdots, \frac{|C_k^{(2)}|}{\theta}\right) \cdot \left[\frac{\theta}{n}\right]
\end{aligned}$$

Using the concavity of r, the above equation can be simplified as

$$\frac{n}{n(\tau_L)}\left[\sum_{i=1}^{2} \mathcal{R}^{re}(\tau_{L_i})\right] \leqslant r\left(\left\{\frac{n(\tau_L) - \theta}{n(\tau_L)}\right\} \times \left(\frac{|C_1^{(1)}|}{n(\tau_L) - \theta}, \cdots, \frac{|C_k^{(1)}|}{n(\tau_L) - \theta}\right) + \left\{\frac{\theta}{n(\tau_L)}\right\} \times \left(\frac{|C_1^{(1)}|}{\theta}, \cdots, \right.\right. \tag{5.31}$$

$$= r\left(\frac{|C_1^{\tau_L}|}{n(\tau_L)}, \cdots, \frac{|C_k^{\tau_L}|}{n(\tau_L)}\right) \tag{5.32}$$

Now the term $r(\tau_L)\widehat{p}(\tau_L)$ can be simplified as:

$$r(\tau_L)\widehat{p}(\tau_L) = r(\tau_L) \times \frac{n(\tau_L)}{n} = r\left(\frac{|C_1^{\tau_L}|}{n(\tau_L)}, \cdots, \frac{|C_k^{\tau_L}|}{n(\tau_L)}\right) \times \frac{n(\tau_L)}{n} \tag{5.33}$$

Using (5.31) and (5.33), it is clear that

$$\mathcal{R}^{re}(\tau_{L_1}) + \mathcal{R}^{re}(\tau_{L_2}) \leqslant \mathcal{R}^{re}(\tau_L)$$

This proves the lemma. □

One of the well-known properties of a concave function is given below without proof.

Proposition 5.1. *Let $\phi : \mathbb{D} \longrightarrow \mathbb{R}$ be a concave function on the domain $\mathbb{D} \subset \mathbb{R}^d$. Let α_i for $i \in \{1, 2, \cdots, n\}$ be such that $\sum_{i=1}^{n} \alpha_i = 1$ and suppose that $\tilde{x}_i$ for $i \in \{1, 2, \cdots, n\}$ be points in D, then ϕ is concave iff*

$$\phi(\sum_{i=1}^{n} \alpha_i \tilde{x}_i) \geqslant \sum_{i=1}^{n} \alpha_i \phi(\tilde{x}_i) \tag{5.34}$$

Equality in above inequality (5.34) *holds iff $\tilde{x}_1 = \tilde{x}_2 = \cdots = \tilde{x}_n$ (OR) ϕ is linear on D.*

From the Proposition 5.1, it is easy to derive that $\mathcal{R}^{re}(\tau_{L_1}) + \mathcal{R}^{re}(\tau_{L_2}) = \mathcal{R}^{re}(\tau_L)$ holds iff

$$\frac{|C_i^{(1)}|}{n(\tau_L - \theta)} = \frac{|C_i^{(2)}|}{\theta} \text{ for all } i \in \{1, 2, \cdots, k\}$$

From the definition of $\mathcal{R}(T)$, it seems that $\mathcal{R}(T)$ must be as low as possible. However, Lemma (5.3) implies that $\mathcal{R}(T)$ value is the least for $T = T_{max}$ where T_{max} is the tree T grown upto saturation.

A **saturated tree** is a tree whose nodes cannot be split further. Generally, a number n_{min} is decided before hand. Once the number of observations falling into a node is less that n_{min} for the first time, then we stop splitting that node. Once every node has less than n_{min} observations, the tree is said to be saturated.

The tree that best describes the data or the tree which we wish to find is a subtree of T_{max} which minimizes a suitable estimate for $\mathcal{R}^{re}(T)$.

Minimizing $\mathcal{R}^{re}(T)$ in order to identify the best subtree is flawed as $\mathcal{R}^{re}(T)$ is always minimized at $T = T_{max}$. The tree T_{max} is very likely to overfit the date at hand. Similarly, a tree that is not "split enough number of times" can underfit the data. Therefore,it is unwise to use $\mathcal{R}^{re}(T)$ to identify the "best" tree that describes the data. Since the intuitive $\mathcal{R}^{re}(T)$ doesn't work, we use a different estimate of the error $\mathcal{R}(T)$ in order to find the best possible subtree.

$$\mathcal{R}_\alpha(T) = \mathcal{R}^{re}(T) + \alpha|\tilde{T}| \tag{5.35}$$

$\alpha 0$ in the above estimate for $\mathcal{R}(T)$ is called as the **complexity parameter**. The **penalty** term ensures that the tree $T \subset T_{max}$ which minimizes the error estimate $\mathcal{R}_\alpha(T)$ is neither "too big" nor "too small".

For each α, let $T(\alpha)$ denote the subtree of T_{max} that minimizes $\mathcal{R}_\alpha(T)$.

$$T(\alpha) = \arg\min_{T \subset T_{max}} \mathcal{R}_\alpha(T) \tag{5.36}$$

$T(\alpha)$ that satisfies equation (5.36) s called the **minimizing subtree**. It is also called as the **optimally pruned subtree** of T_{max}.

Remark 5.7. Note that $T(\alpha)$ for a given α is not unique. How to get around this problem is discussed later.

The relationship between α value and optimal tree "size" is summarized below. The word "size of a tree" is vague. However, for now the size of the tree can be seen as number of nodes and branches. Tree with greater number of nodes and brances is bigger in size.

1. When α is close to zero, the penality $\alpha|\tilde{T}|$ is small. So the dominant term in $\mathcal{R}_\alpha(T)$ is $\mathcal{R}^{re}(T)$. Hence the minimizing subtree is "bigger" in size. When $\alpha = 0$, then the minimizing subtree is T_{max}.

2. On the other hand, if α is large, then $\alpha|\tilde{T}|$ is the dominant term in $\mathcal{R}_\alpha(T)$. Hence the minimizing subtree is "smaller" in size. For all α which are "sufficiently large", the minimizing subtree is τ_{root} i.e. tree consisting of the root node alone.

Therefore, as α value increases the number of nodes in the optimal subtree $T(\alpha)$ reduces i.e. we prune the tree T_{max} upwards.

The number of possible subtrees of a given tree is finite, therefore the set of all possible α values can be partitioned into intervals $[0, \alpha_1] \cup (\alpha_1, \alpha_2] \cup \cdots [\alpha_M, \infty)$ such that $T(\alpha_i)$ is the optimal subtree for all $\alpha \in (\alpha_{i-1}, \alpha_i]$. One of these $M + 1$ optimal subtrees is the tree that best describes our data. Remaining chapter focuses on how a classification tree is constructed for a given data set D. The nodes of the tree T_{max} are pruned systematically to get the "best" decision trees $T(\alpha_i)$.

When the value of α gradually and continuously increases from 0, the output value the function $\arg\min_{T \subset T_{max}} R_\alpha(T)$ is a continuous step function. Let α_1 and α_2 be two possible α values such that $\alpha_1 < \alpha_2$. Then,

$$\mathcal{R}_{\alpha_1}(T) = \mathcal{R}^{re}(T) + \alpha_1|\tilde{T}| \text{ and } \mathcal{R}_{\alpha_2}(T) = \mathcal{R}^{re}(T) + \alpha_2|\tilde{T}|$$

Additional, consider the following notations as well:

$$\mathcal{R}(\alpha_i) = \min_{T \subset T_{max}} \mathcal{R}_{\alpha_i}(T) \text{ and } T(\alpha_i) = \arg\min_{T \subset T_{max}} \mathcal{R}_{\alpha_i}(T) \text{ for } i \in \{1, 2\}$$

5.5 Constructing a sequence of minimizing subtrees

As mentioned earlier, as α value is gradually increased, the branches of the corresponding minimizing subtree are chopped off. The construction of this sequence of subtrees and corresponding α values is shown in this section. The next section will focus on determining the best pruned subtree among the sequence of constructed optimal subtrees.

Using the notation T_i for $T(\alpha_i)$, a sequence of trees $T_1, \cdots, T_M$ are constructed as shown below. Note that T_M stands for τ_{root} while T_0 stands for T_{max}. This is represented pictorially in 5.9.

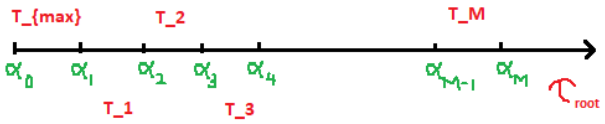


Figure 5.9: Optimal subtrees and α values

Before looking at the construction of T_1, let us first focus on T_{max} i.e. the optimal subtree for $\alpha \in [0, \alpha_1)$. Without loss of generality, T_{max} can be considered as the smallest subtree T' of T_{max} for which $R^{re}(T_{max}) = R^{re}(T')$ hold. It makes sense to use this condition as we have to start with the value $\alpha = 0$. The smallest optimal subtree T' satisfying $R_0(T_{max}) = R_0(T_{max})$ will remain the optimal subtree until some value of α, say α_1. Additionally, it is easy to show that finding the smallest subtree of T_{max} that satisfies the condition $R^{re}(T_{max}) = R^{re}(T')$is equivalent to chopping the daughter nodes τ_1 and τ_2 that satisfy the following equation with respect to their parent node τ_p.

$$\mathcal{R}^{re}(\tau_p) = \mathcal{R}^{re}(\tau_1) + \mathcal{R}^{re}(\tau_2)$$

Now that we have T_{max}, we will now consider the construction of T_1 from T_{max}. In order to understand the construction of T_1, we must understand the concept of "**weakest link node**".

5.5.1 Weakest link node

We will understand the concept of weakest link using a tree T. This will later be applied to the tree T_1 that we were discussing about. Let τ be a non-terminal node of the tree T. Denote by T_τ the maximal subtree of T with τ as its root node. Let the terminal nodes of T_τ be given by $\tilde{T}_\tau = \{\tau'_1, \cdots, \tau'_{L_\tau}\}$. Then,

$$\mathcal{R}^{re}(T_\tau) = \sum_{\tau' \in \tilde{T}_\tau} \mathcal{R}^{re}(\tau') \text{ where } \mathcal{R}^{re}(\tau') = r(\tau_{l'}) \times p(\tau_{l'})$$

Lemma 5.4. *For trees T, T_τ mentioned above, the following is satisfied.*

$$\mathcal{R}^{re}(T_\tau) < \mathcal{R}^{re}(\tau)$$

Proof. Using the lemma 5.3, it is clear that $\mathcal{R}^{re}(T_\tau) \leqslant \mathcal{R}^{re}(\tau)$ holds. This is true because τ can be treated as a trivial subtree of T_τ. We are left to show that the inequality is always strict. Assume on the contrary that $\mathcal{R}^{re}(T_\tau) = \mathcal{R}^{re}(\tau)$ holds.

Let the splits $s_1, s_2, \cdots, s_g$ be required to split the node τ to create T_τ. This can be represented as follows for simplicity:

$$T_0 = \tau \xrightarrow{s_1} T_{s_1} \xrightarrow{s_2} T_{s_2} \xrightarrow{s_3} \cdots \xrightarrow{s_g} T_{s_g} = T_\tau \tag{5.37}$$

By the property of $\mathcal{R}^{re}$, the following holds

$$\mathcal{R}^{re}(\tau) \geqslant \mathcal{R}^{re}(T_{s_1}) \geqslant \cdots \geqslant \mathcal{R}^{re}(T_\tau) \tag{5.38}$$

However, using the assumption $\mathcal{R}^{re}(T_\tau) = \mathcal{R}^{re}(\tau)$ along with the inequality 5.38, a contradiction can be arrived at in the following manner:

$$\tilde{T}_g = \{\tau_1, \tau_2, \cdots, \tau_{l_\tau - 1}, \tau_{l_\tau}\} \text{ and } \tilde{T}_{g-1} = \{\tau_1, \tau_2, \cdots, \tau_{l_\tau - 2}, \tau_{l,l-1}\}$$

In other words, the node $\tau_{l,l-1}$ of T_{g-1} was split into two nodes - $\tau_{l_\tau - 1}, \tau_{l_\tau}$ - to give rise to the tree T_g. Simple manipulations can show that the assumption of $\mathcal{R}^{re}(T_\tau) < \mathcal{R}^{re}(\tau)$ gives the conclusion:

$$\mathcal{R}^{re}(\tau_{l,l-1}) = \mathcal{R}^{re}(T_{\tau_{l,l-1}}) \tag{5.39}$$

The equality in 5.39 gives a contradiction as the tree we are working with is assumed to be the tree which no node tuples following equations of the kind in (5.39). □

Theorem 5.1. *The following statements are equivalent.*

1. *The tree T describes the data better than $T \backslash T_\tau$.*
2. $\mathcal{R}_\alpha(T) < \mathcal{R}_\alpha(T \backslash T_\tau)$
3. $\mathcal{R}_\alpha(T_\tau) < \mathcal{R}_\alpha(\tau)$
4. $\alpha < \dfrac{\mathcal{R}^{re}_\tau - \mathcal{R}^{re}_{T_\tau}}{|\tilde{T}_\tau| - 1}$

Proof. The three statements above can be proved easily by using the definitions of the function $\mathcal{R}_\alpha$. □

Although the proof of Theorem 5.1 is pretty straightforward, the information the theorem provides is extremely crucial for the construction of the sequence of optimal subtrees.

The theorem 5.1 states that the tree T better describes the data $T \backslash T_\tau$ iff $\mathcal{R}_\alpha(T_\tau) < \mathcal{R}_\alpha(\tau)$ holds. Additionally, $\mathcal{R}_\alpha(T_\tau) < \mathcal{R}_\alpha(\tau)$ will hold or T will be better than $T \backslash T_\tau$ if and only if $\alpha < \dfrac{\mathcal{R}^{re}_\tau - \mathcal{R}^{re}_{T_\tau}}{|\tilde{T}_\tau| - 1}$ holds.

Keeping the Theorem 5.1 in mind, we can now try to understand the concept of weakest link node.

Definition 5.11. Let T be a tree with terminal nodes $\tilde{T} = \{\tau_1, \tau_2, \cdots, \tau_L\}$. Let the non-terminal nodes be denoted by $\tilde{T}^C$ Then denote by $g(\tau)$

$$g(\tau) = \frac{\mathcal{R}^{re}_{\tau} - \mathcal{R}^{re}_{T_\tau}}{|\tilde{T}_\tau| - 1} \tag{5.40}$$

Then the weakest link node $\tilde{\tau}$ is defined as follows:

$$\tilde{\tau} = \arg\min_{\tau \in \tilde{T}^C} g(\tau) \tag{5.41}$$

Start with tree T_{max}. For all values of α in $[0, \alpha_1)$, T_{max} is the optimal subtree. From $\alpha = \alpha_1$ onwards, a subtree of T_{max}, call it T_1, is the optimal subtree. The question is how do we find this tree T_1 and the alpha value.

Each non-terminal node τ of the tree has an α value after which the associated T_τ can be chopped off. However, the non-terminal node with smallest alpha value will be the one whose T_τ will be chopped off first as α is gradually increased. The alpha value and its corresponding node are denoted by α_1 and τ_1. Since T_{τ_1} corresponding to τ_1 is the first to be chopped off, τ_1 is called the weakest link node. The same procedure of finding the weakest link node is repeated for the tree T_1 to give rise to α_2 and T_2. This procedure is continued till we find an α_M after which τ_{root} is the best tree. Hence this process creates a sequence of α values and trees as written below and as shown in Figure 5.10:

$$0 = \alpha_0 < \alpha_1 < \cdots < \alpha_{M-1} < \alpha_M$$

$$T_{max} = T_0 \supset T_1 \supset \cdots \supset T_{M-1} \supset T_M = \tau_{root}$$

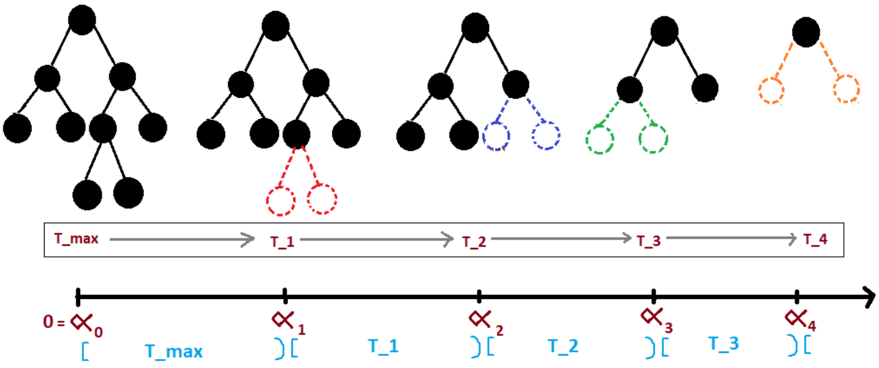


Figure 5.10: Tree Pruning

We earlier addressed the problem of non-uniqueness of optimal subtrees for a given α value. This problem is solved by looking for **smallest minimizing subree** of the tree of interest. It can be proved that smallest minimizing subtree always exists and it is unique.

Definition 5.12. Fix an α value and a tree T. Let the minimizing subtree be denoted by $T(\alpha)$. The smallest minimizing subtree T_{sm} is the subtree of T that satisfies the following two conditions.

1. T_{sm} is minimizing subtree of T.
2. If T'_{sm} is any other minimizing subtree, then T_{sm} is a subtree of T_s.

It can be proved that the smallest minimizing subtree always exists and it is unique for a given tree. The proof of this statement has not been provided in this chapter. This then solves the problem of having non-unique optimal subtrees for a given α value.

5.5.2 The best pruned subtree

Now that we have a sequence of subtrees where each subtree is the smallest minimizing subtree, we have to find a way to find the **best pruned subtree** among the set of all possible subtrees.

We consider the method of **using independent test sample**. If D denotes the entire set, then divide the set into two subsets - $\mathcal{L}$ will be the learning set or the training set while $\mathcal{T}$ will be the testing set such that $|\mathcal{L}| = n_L$ and $|\mathcal{T}| = n_T$. We assume that the observations in the test set are drawn independently from those in $\mathcal{L}$. The following procedure is used to find the best pruned subtree.

- Use the learning set $\mathcal{L}$ to grow a sequence of trees as described earlier.
- Now drop the n_T test set observations down each of the trees in the sequence.
- For each tree in the sequence, calculate the proportion of observations that were miss-classified.
- Best pruned subtree is the one with least miss-classification error.

Although this methods works well for large data sets, this might not be a good approach for small data sets. Additionally, the other problem lies in the fact that our final answer depends on how the data set was divided into learning and testing sets. Not using the elements of $\mathcal{T}$ to construct a tree implies loss of important information.

5.6 Data Analysis

In this section, we put into use the classification tree construction technique in order to differentiate fake notes from original notes. The dataset has been taken from the UCI machine learning repository.

The data set we use is found as the “data_banknote_authentication” data set at the UCI machine learning repository. It is a dataset of dimension 1371×5. The first four variables represent different properties of fake and original banknotes. These four variable values were computed using a wavelet transform tool. The four variables are **continuous** and denoted as follows:

1. "var" - variance of Wavelet Transformed image

2. "skew" - skewness of Wavelet Transformed image

3. "kurt" - kurtosis of Wavelet Transformed image

4. "entropy" - entropy of image

The fifth variable denotes the class label. "Class" variable takes one of the two values - 0 or 1. However, I could not find information regarding which variables implies a fake note and which one implies a real note. Let us assume for now that all notes with label1 are real while all those with label 0 are fake.

```
> names(data_banknote_authentication)
> colnames(data_banknote_authentication) = c("var", "skew", "kurt", "entropy", "cl
> dim(data_banknote_authentication)
[1] 1372    5

> fac = as.factor(data_banknote_authentication[,5])
> table(fac)
fac
  0   1
762 610
```

Therefore there are 762 fake notes and 610 real notes included in this dataset. This dataset, as mentioned in this chapter, must be divided into two parts - testing set and training set. It makes sence to approach this method as number of data points we have is sufficiently large. We divide the entire set into two subsets. The subset “train” denotes the training set while the set “test” denotes the testing set.

```
> test = data_banknote_authentication[753:772,]
> dim(test)
[1] 20  5
> train = data_banknote_authentication[-c(753:772),]
> dim(train)
[1] 1352    5
```

Now that we have a clear testing and a training set, use the function “rpart()” from the library “rpart” to construct a decision tree. Note that we use entropy function to split the nodes. The tree that we get after running the code is shown in figure 5.11

```
> library(rpart)
> tree = rpart(class ~., data = train, method = "class", parms = list(split = "inf
> plot(tree)
> text(tree)
```

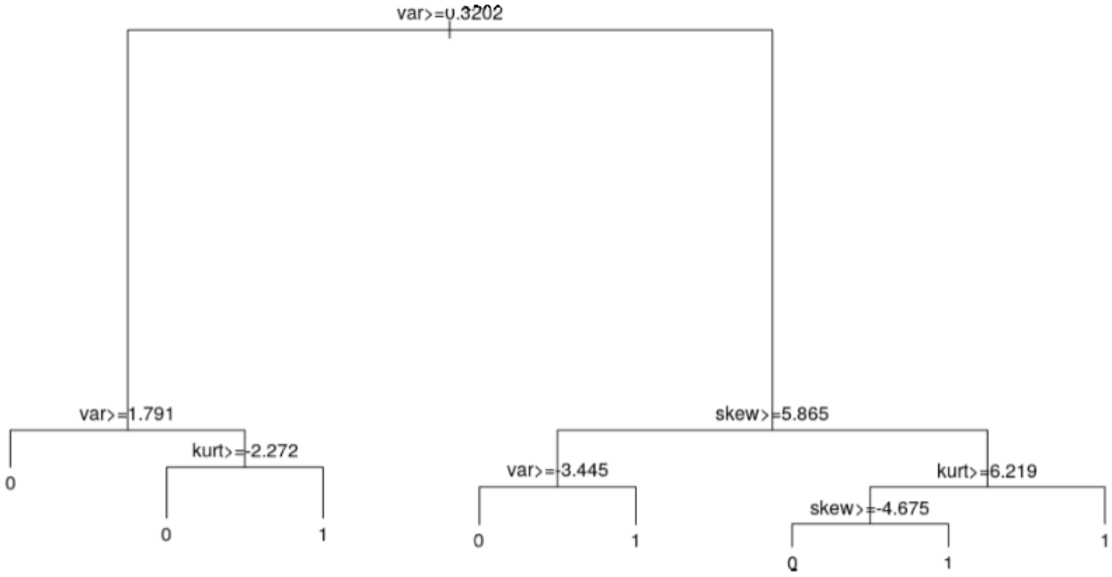


Figure 5.11: Classification tree to classify real and fake notes

Running the code associated with the function rpart() gives the following result.

```
> tree
n= 1352

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 1352 600 0 (0.556213018 0.443786982)
   2) var>=0.320165 704  75 0 (0.893465909 0.106534091)
     4) var>=1.7907 474   5 0 (0.989451477 0.010548523) *
     5) var< 1.7907 230  70 0 (0.695652174 0.304347826)
      10) kurt>=-2.2722 173  16 0 (0.907514451 0.092485549) *
      11) kurt< -2.2722 57   3 1 (0.052631579 0.947368421) *
   3) var< 0.320165 648 123 1 (0.189814815 0.810185185)
     6) skew>=5.86535 135  39 0 (0.711111111 0.288888889)
      12) var>=-3.4449 95   0 0 (1.000000000 0.000000000) *
      13) var< -3.4449 40   1 1 (0.025000000 0.975000000) *
     7) skew< 5.86535 513  27 1 (0.052631579 0.947368421)
      14) kurt>=6.21865 153  25 1 (0.163398693 0.836601307)
        28) skew>=-4.6745 25   1 0 (0.960000000 0.040000000) *
        29) skew< -4.6745 128   1 1 (0.007812500 0.992187500) *
      15) kurt< 6.21865 360   2 1 (0.005555556 0.994444444) *
```

The line written against 1) gives information about the root node or the first node. The root node has no split criteria defined yet. The line mentions that there are 1352 observations in the data set.

```
> factor = as.factor(train[,5])
> table(factor)
> factor
  0   1
752 600
```

The code above lets us know that there are 752 fake and 600 real notes at the root node. Therefore, we would have to assign the class label 0 to this node as majority of the data points in the node are of class 0. This information has been suggested in the line written against "1)" in the code output given above. The word "root" under the variable "split" suggests that it is a root node and therefore no splitting criteria has been defined yet. However, in the remaining lines a splitting criterion is mentioned under the variable "split". The variable "n" in the code is the value $n(\tau_{root})$ i.e. the number of observations associated with the root node. The variable "loss" gives the number of observations that we consider miss-classified. Since $600 < 752$, we assume the 600 points to be mis-classified (since the node is assigned the class label 0). The variable "yval" is nothing but the class label i.e.0 or 1. The numbers given in the bracket are the proportion of points that belong to each class for a given set of points associated with a node. Observe that the proportions mentioned in the first line are 0.556213018 and 0.443786982 which are nothing but $\frac{752}{1352}$ and $\frac{600}{1352}$ respectively. Observe the indentation in the lines of the code output and the graph in 5.11. Then it can be observed that the indentation in the output gives information about the structure of the final tree.

Now that we have constructed a tree, we must check how well the tree works with respect to the testing data. This is done using the "predict()" function found in the rpart package. Note that the serial numbers given in the output below is the serial number of the data point in the original dataset.

```
 > predict(tree, test)
              0           1
753 0.907514451 0.09248555
754 0.989451477 0.01054852
755 0.989451477 0.01054852
756 0.989451477 0.01054852
```

```
757 0.989451477 0.01054852
758 0.989451477 0.01054852
759 0.989451477 0.01054852
760 1.000000000 0.00000000
761 0.989451477 0.01054852
762 0.989451477 0.01054852
763 0.005555556 0.99444444
764 0.907514451 0.09248555
765 0.007812500 0.99218750
766 0.007812500 0.99218750
767 0.007812500 0.99218750
768 0.005555556 0.99444444
769 0.005555556 0.99444444
770 0.005555556 0.99444444
771 0.907514451 0.09248555
772 0.005555556 0.99444444
```

The true class labels of these 20 testing points can be obtained using the original dataset. The output below suggests that the first ten data points correspond to a fake note while the last ten data points correspond to real notes.

```
> data_banknote_authentication[753:772, 5]
[1] 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1
```

Comparing this with the outcome that we got using our decision tree, we observe that all the fake notes were correctly classified as being fake. However two of the real notes have been miss-classified as being fake. These incorrectly classified observations are the 764^{th} and 771^{th} observations of the original data set. This implies that 90% of the test set observations were classified correctly which means that the tree can act as a good classifier.

Chapter 6

Artificial Neural Networks

The following chapter is based on [RS20], [Roj96], [BN12], [Kha19], and [Ize08]. The dataset has been taken from [DG17].

This chapter focuses on "Artificial Neural Networks" which is one of the many classification tools that we know of so far. The name "artificial" neural network comes fact that this theory was first developed in an attempt to model how the neurons in human brain work. Nowadays, however, ANNs are dealt with in a more abstract fashion. This chapter focuses on how a particular type of ANN can be constructed, using a training set, to solve classification problems. We first look at how biological neurons inspired the first artificial neuron models. My inherent interest in biology, made me quite inclined to include a section on the biological inspiration behind ANNs.

6.1 Biological Neurons inspired Artificial Neurons

6.1.1 Biological neurons and neural networks

In this section, we focus on how biological neurons work and how the function of these network of neurons inspired the first mathematical neuron model.

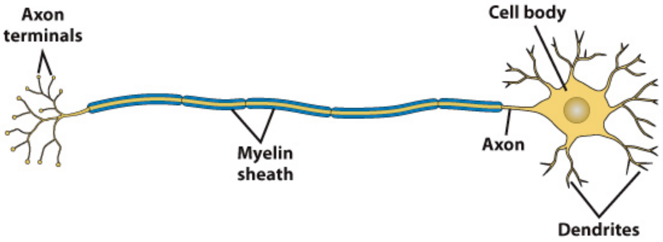


Figure 6.1: Structure of a biological neuron (©The Alcohol Pharmacology Education Partnership)

Coordination between billions of neurons in the brain is the reason why we are able to sense, move, feel emotions and much more. In order to execute these functions, our brain uses an extensive networks of neurons to rapidly communicate via electrical signals. Each neuron comprises a cell body, dendrites, axon and axon terminals as shown in Figure 6.1. An electrical impulse (action potential) is generated in the cell body which propagates along the axon to the axon terminal. The electrical impulse in the axon terminal of one neuron is relayed onto another adjacent neuron through the dendrite-terminal bridge as shown in 6.2. This terminal-dendrite connection between two neurons is called a synapse. Therefore, dendrites receive the inputs while the axon terminals receive the output. A given neuron can send and receive signals to multiple neurons as both dendrites and axon terminals are connected to several other neuron's axon terminals and dendrites respectively.

Now that we have some understanding of how neurons communicate, it is useful to know about action potential. Action potential is a temporary change in neuron's membrane potential that is caused due to external stimulus. A neuron "fires" or sends the electrical message/signal to the next neuron only if the change in membrane potential crosses a certain threshold value of -55mV. The neuron returns to its resting state if the stimulus or change in membrane potential is not strong enough. Interestingly, the strength of the action potential is always the same irrespective of how severe or mild the stimulus is. However, a weak stimulus triggers a less frequent action potentials while intense stimuli trigger more frequent action potentials. Signals that a neuron receives can either be excitatory or inhibitory. As the name suggests, these signals change the membrane potential in a direction that is either towards or away from the fixed threshold value of -55mV.

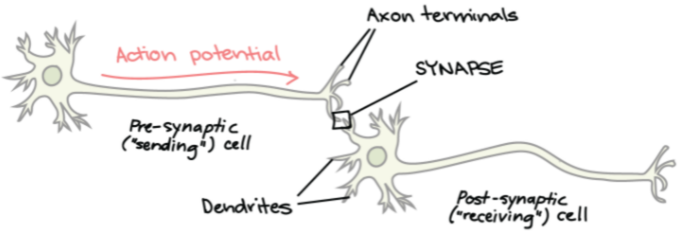

Axon terminals
Action potential
SYNAPSE
Pre-synaptic ("sending") cell
Dendrites
Post-synaptic ("receiving") cell

Figure 6.2: Communication between neurons (©Khan Academy)

6.1.2 Artificial neurons

Now that we understand how a biological neuron works, the statistical model of a neuron will appear to be obviously defined based on how the neurons work. This section is devoted to given a brief introduction to artificial neurons. Various terminology related to ANNs will also be introduced in this section. Artificial neural networks are statistical models that are used to solve classification problems. Therefore ANNs, like decision trees, are one of the possible classifier options we have. Although ANNs can be used to solve regression problems, we focus only on ANN based classification in this chapter.

ANNs,just like decision trees, are trained using a set of labeled observations i.e. ANN is a supervised learning algorithm. We first look at what constitutes a single neuron. We then look at how a network of such single neurons work.

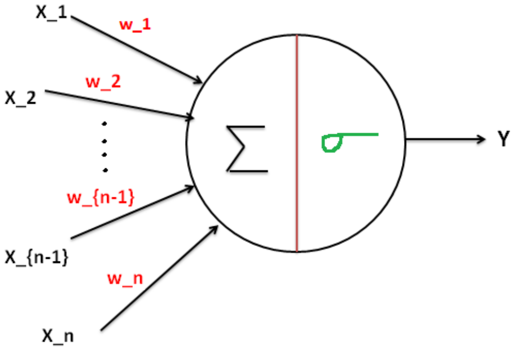


Figure 6.3: An artificial neuron

6.1.3 Single neuron

An individual neuron or **computing unit** consists of n input nodes. This implies that each observation $\tilde{X} \in \mathbb{R}^n$.The circle in 6.3 plays the role of the cell body and axon i.e. transmit the information received at the "dendrites". Dendrites here are the arrows pointing into the circle. The label Y given at the end is the output we get after the input is processed in the neuron. The "processing" of the input is done by the two functions $\Sigma : \mathbb{R}^n \longrightarrow \mathbb{R}$ and $\sigma : \mathbb{R} \longrightarrow \mathbb{R}$. Σ adds up all the weighted inputs $w_1 \times X_1, w_2 \times X_2, \cdots, w_n \times X_n$ while σ acts as a normalizer. Since the input values can vary in magnitudes depending on which scale they are measured on. Therefore, σ is used to normalize the sum of input values to some value between 0 and 1 or between -1 and 1. σ is also called as the **activation function**.

In conclusion, the "processing" that happens in the neuron can be summed up as follows:

$$(x_1, x_2, \cdots, x_n) \mapsto w_1x_1 + w_2x_2 + \cdots + w_nx_n \mapsto \sigma\left(\sum_{i=1}^{n} w_ix_i\right)$$

Activation functions must be bounded, continuous, monotonic and continuously differentiable with respect to the vector of weights $\tilde{w}$. The activation function must be bounded because it normalizes input sums. On the other hand the conditions of continuity and differentiability are useful for optimization purposes. Monotonicity is important because activation functions are a measure of how negative or positive the sum of inputs is. Activation functions are sometimes continuous functions but not differentiable as well. One of the examples of such an activation function is the step function at origin i.e. $\sigma(x) = \mathcal{I}_{x \geqslant 0}$ where $\mathcal{I}$ stands for indicator function.

Some commonly used continuous and differentiable activation functions are:

1. $\sigma : \mathbb{R} \longrightarrow [0, 1]$ such that
$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{6.1}$$

2. $\sigma : \mathbb{R} \longrightarrow [-1, 1]$ such that
$$\sigma(x) = \frac{2}{\pi} \arctan(x)$$

3. $\sigma : \mathbb{R} \longrightarrow [-1, 1]$ such that
$$\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

However, the most commonly used activation function is the **sigmoid function**, i.e. the one in (6.1). Sigmoid function is a measure of how positive or negative the sum of inputs is. The graph of each of these activation functions is provided in 6.4. The activation function we use throughout this chapter is the sigmoid function unless otherwise stated.

The weights w_i are a measure of the frequency or strength with which an input message x_i is received. Every individual neuron is also associated with a threshold θ, just like in the case of a biological neuron, only after which the neuron fires. This threshold is called as the **bias** of the neuron. Once the inputs $x_1, x_2, \cdots, x_n$ are entered into the neuron for processing, the first step involves the Σ function calculating the weighted sum $\sum_{i=1}^{n} w_i x_i$. The bias θ is then subtracted from the weighted sum before using it as an input for the function σ. This bias θ is a measure of the minimum value the weighted sum can take.

$$(x_1, x_2, \cdots, x_n)^T \xrightarrow{\Sigma} \sum_{i=1}^{n} w_i x_i \longrightarrow \sum_{i=1}^{n} w_i x_i - \theta \xrightarrow{\sigma} \sigma\left(\sum_{i=1}^{n} w_i x_i - \theta\right)$$

Let us assume that the weighted sum of all the inputs have to be at least 10 to consider the stimulus/signals to be strong enough. Then we choose $\theta = 10$. Subtracting $\theta = 10$ from each data point would give us a measure of high or low the combined signal strength is with respect to 10. In other words, we are shifting the co-ordinates such that the real number 10 is now the origin of the real line.

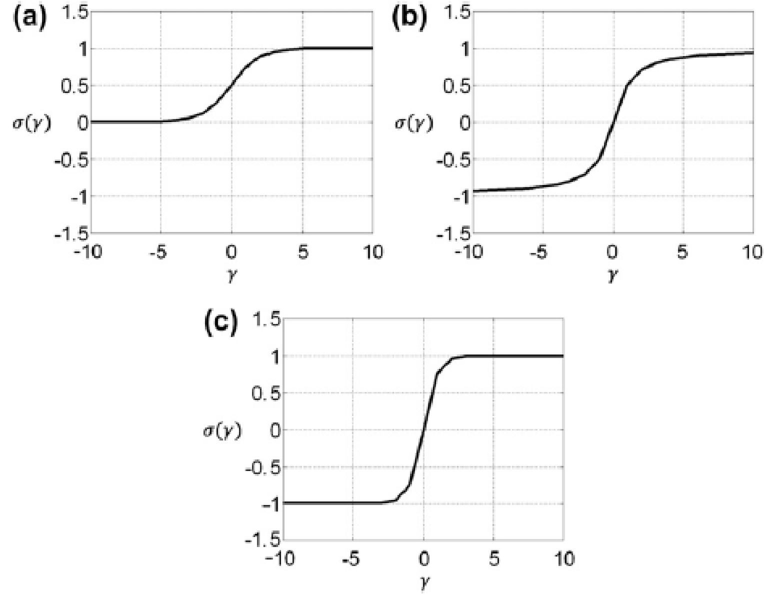


Figure 6.4: (a)The sigmoid function, (b)The arc-tangent function, (c)The hyperbolic tangent function [Reference: [BN12]]

6.1.4 Artificial Neural Network

Just like a network of biological neurons work in coordination, a network of artificial neurons, called as artificial neural networks are used to address classification problems. Several different types of ANNs have been defined in the literature so far. However, the ANN we will focus on is a most frequently used ANN called **feed-forward multilayer perceptron** or MLP in short. The pictorial representation of the MLP is given in Figure 6.5.

As the name suggests, MLP is a neural network that is organized in "layers". Each layer is group of nodes that processes the inputs it gets and gives an output. These outputs act as inputs to the next layer of nodes. Hence the nextwork is **layered** and **feeds forward**. The first layer of nodes is called the **input layer**. This is called a "layer" although the inputs are taken in as they are without any processing. The last layer of nodes which produces the estimated class label is called the **output layer**. All the other layers inbetween input and output layers are called **hidden layers**. The number of nodes present in different layers can be different as shown in Figure 6.5. The complexity of a neural network is based on the choice of number of hidden layers and the number of nodes in each hidden layer.

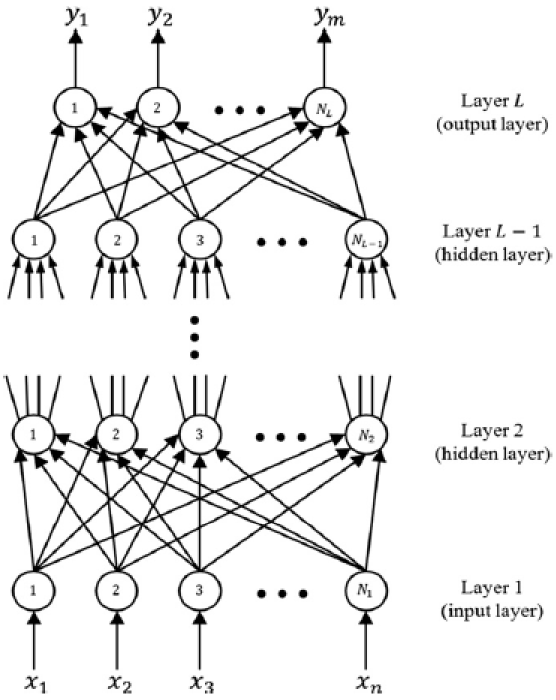


Figure 6.5: Feed-forward multi-layer perceptron taken from [BN12]

6.2 The Classical Perceptron

In the previous section we focused on understanding the structure of the feed-forward multilayer perceptron (MLP) and how biological neural networks inspired the modelling of the MLP. Now, we take a step back to look at the artificial neural network models that were proposed before the MLP and how these earlier models were modified to give the model that is often used today.

6.2.1 McCullough-Pitts neuron

The first mathematical model of how an individual neuron works was proposed by two people - Warren McCullough and Walter Pitts. The single neuron model they proposed, also called as the McCullough-Pitts neuron (or MP neuron), has the structure as shown in 6.7.

- The MP neuron takes in binary inputs and gives out binary outputs.
- The MP neuron is unweighted.
- Each neuron is associated with a threshold or bias θ.

- The activation function here is the step function at θ i.e. $\sigma(x) = \mathcal{I}_{x \geqslant \theta}$. Choice of activation function ensures that the neuron outputs binary values only i.e 0 or 1.

- The neuron is said to fire when the neuron outputs 1.

- Each input signal is either an excitatory signal or an inhibitory signal.

- Let the inputs $e_1, e_2, \cdots, e_{m_1}$ be m_1 excitatory input signals while $in_1, in_2, \cdots, in_{m_2}$ be inhibitory input signals. Let y denote the output associated with this neuron. Then,

$$y = \begin{cases} 0, & \text{if at least one} in_i = 1, i \in \{1, 2, \cdots, m_2\} \\ 0, & \text{if } in_j = 0 \forall j \text{ and } \sum_{i=1}^{m_1} w_i e_i < \theta \\ 1, & \text{if } in_j = 0 \forall j \text{ and } \sum_{i=1}^{m_1} w_i e_i \geqslant \theta \end{cases} \tag{6.2}$$

Examples of individual MP neurons for AND, OR function is given in Figure 6.6. Note that all boolean functions can be represented with a combination of three different gates - AND, OR and NOT. Since individual AND, OR and NOT functions can be represented using an MP neuron, it is clear that every possible boolean function can be represented using an MP neuron.

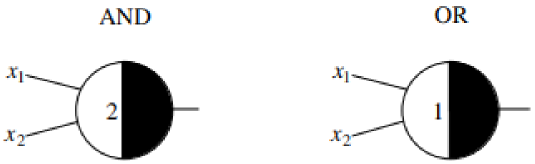


Figure 6.6: MP neurons for AND, OR functions taken from [Roj96]

6.2.2 Rosenblatt's Perceptron

After McCullogh and Pitts came the psychologist Donald Hebb who proposed the idea of **Hebbian Learning**. To put it briefly and simply, Hebbian learning says that if a neuron A repeatedly excites a neuron B, the connection strength between A and B will be strengthened i.e. it gets easier for A to trigger B. It is intuitive, based on the biology of neurons we saw so far, that this idea can be implemented into the neuron model by introducing weights. Rosenblatt's perceptron is a modification of the McCulloch-Pitt neuron that incorporates the idea of Hebbian learning. A pictorial representation of the Rosenblatt's perceptron, also called as the classical perceptron, is given in Figure 6.7. The structure of the classical perceptron can be summarized in the points below:

- Inputs and outputs are binary. Denote the m inputs as $x_1, x_2, \cdots, x_m$.
- The threshold function is a step function at θ, just like it was in the case of an MP-neuron.
- The weights are real valued.
- Each neuron is associated with a threshold θ such that the output y is defined as follows:

$$y = \begin{cases} 0, & \text{if } \sum_{i=1}^{m} w_i x_i < \theta \\ 1, & \text{if } \sum_{i=1}^{m} w_i x_i \geqslant \theta \end{cases} \tag{6.3}$$

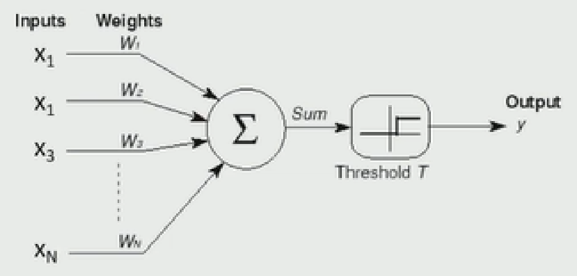


Figure 6.7: The classical perceptron taken from [RS20]

Although it was initially believed that the single perceptron can represent any boolean function, it was later pointed out by Minsky and Papert that the perceptron neuron cannot represent the XOR function. However, the XOR function can be represented with the help of a network of neurons. That is, a network of neurons can represent a larger class of functions than the individual neuron.

Every boolean function can be represented with a network of finite number perceptron units. But the problem with the model is that **real world inputs most often not binary**. This gives rise to a modification of the perceptron called as **the real perceptron**.

The perceptron Since the classical perceptron does not recieve non-binary inputs, it can be modified to allow real values as inputs. The weights, along with the inputs is real valued. The threshold function is still step function at the bias θ. This implies that the output for a single neuron is still binary. The neuron fires only if the weighted average of the input exceeds a given threshold.

Remark 6.1. Note that we refer to Rosenblatt's perceptron as "the classical perceptron" while the perceptron with real valued inputs is called"the real perceptron" or just perceptron when the context is clear.

6.3 Geometric interpretation

Assume m inputs $x_1, x_2, \cdots x_m$. y denotes the output, θ is the threshold and w_i indicates the weight corresponding to x_i for $i \in \{1, 2, \cdots, m\}$. The vector $\tilde{w} = (w_1, w_2, \cdots, w_m)$ is called the weight vector.

Mathematically, the locus of all points $\tilde{x} = (x_1, x_2, \cdots, x_m)^T \in \mathbb{R}^m$ that are assigned 0 or 1 by the single perceptron unit is given as:

$$\mathcal{C}_1 = \{\tilde{x} | \tilde{x}^T \tilde{w} \geqslant \theta\} \text{ and } \mathcal{C}_0 = \{\tilde{x} | \tilde{x}^T \tilde{w} < \theta\}$$

The locus of points in $\mathbb{R}^m$ that separate points in class 1 from those in class zero is given by the set

$$\{\tilde{x} \in \mathbb{R}^m | \tilde{x}^T \tilde{w} = \theta\} = \left\{ \tilde{x} \middle| \sum_{i=1}^{m} w_i x_i = \theta \right\} \tag{6.4}$$

The set of points that separate the points that lie in the two classes is clearly a $m-1$ dimensional **hyperplane** in $\mathbb{R}^m$. In other words, the single perceptron unit with the threshold activation function separates the points of the two possible classes by a **linear decision boundary**.

Example 6.1. Consider for instance the following scenario. Input is two dimensional i.e. all our inputs are of the form $(x_1, x_2)^T$. Let the weights corresponding to x_1 and x_2 be 0.3 and 0.5 respectively. Let the threshold value be $\theta = 1$. Then the decision boundary defined by $0.3x_1 + 0.5x_2 = 1$ is nothing but a 1-dimensional hyperplane in $\mathbb{R}^2$. The graph of the decision boundary is shown in 6.8.

$$y = \begin{cases} 0, & \text{if } 0.3x_1 + 0.5x_2 < 1 \\ 1, & \text{if } 0.3x_1 + 0.5x_2 \geqslant 1 \end{cases} \tag{6.5}$$

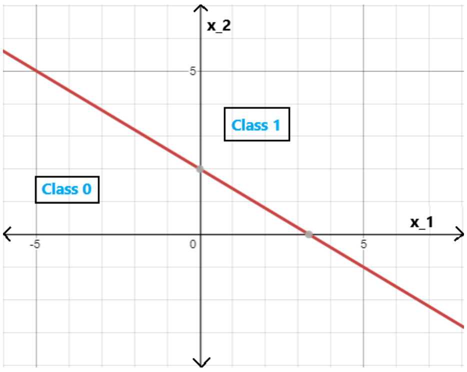


Figure 6.8: $0.3x_1 + 0.5x_2 = 1$

△

Notation

Although the bias θ for each perceptron is a pre-determined and not an input, $-\theta$ is often considered as an $(m+1)^{th}$ weight corresponding to an additional input $x_{m+1} = 1$. Thus,the input and vectors are “extended”. The number written inside a node mentions the θ value in the case when the activation function is step function at θ.

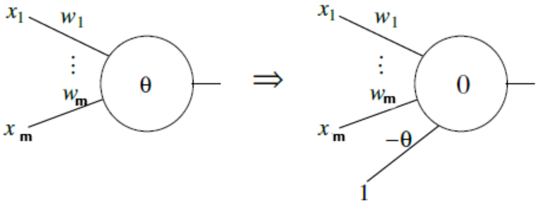


Figure 6.9: Considering bias as an additional input (Figure taken from [Roj96])

The advantage of this is that the separating hyperplane **always passes through the origin**. Reconsider the example 6.1. The extended input and output vectors are $(x_1, x_2, 1)$ and $(0.3, 0.5, -1)$. The threshold for this neuron has to be 0 for it to be equivalent to the original neuron defined in Example 6.1. The separating plane now is:

$$\left\{(x_1, x_2, 1)^T \mid 0.3x_1 + 0.5x_2 - 1x_3 = 0\right\} \subset \mathbb{R}^3 \tag{6.6}$$

The locus in 6.6 is clearly a plane passing through the origin as shown in Figure 6.10. In fact, the locus of points in 6.6 and 6.10 is the plane defined by the corresponding weight vector $(0.3, 0.5, -1)$ as the normal vector.

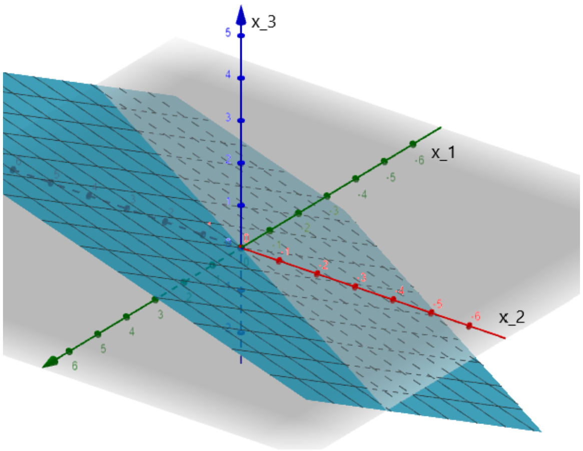


Figure 6.10: Hyperplane $0.3x_1 + 0.5x_2 - 1x_3 = 0$

Generalizing the above information, if the input data $(x_1, \cdots, x_m)^T$ and the corresponding weight vector $(w_1, \cdots, w_m)^T$ lie in $\mathbb{R}^m$, then extending the vectors as follows ensures that the separating hyperplane passes through the origin.

$$\tilde{x} = (x_1, \cdots, x_m, 1)^T \text{ and } \tilde{w} = (w_1, \cdots, w_m, -\theta)^T$$

Note that the original and extended vectors are denoted in the same manner.Depending on the context, I will mention what $\tilde{x}$ and $\tilde{w}$ refer to.

Example 6.2. Consider the boolean functions shown in Figure 6.11. Points lying in the pink side of the plane belong to one class while the others belong to the other class. Since both classical and real perceptron units can only find a hyperplane as the separator, it is clear as to why the boolean function corresponding to (*c*) cannot be represented using a single classical perceptron unit. The boolean function corresponding to (*c*) is nothing but the XOR function we came across earlier.

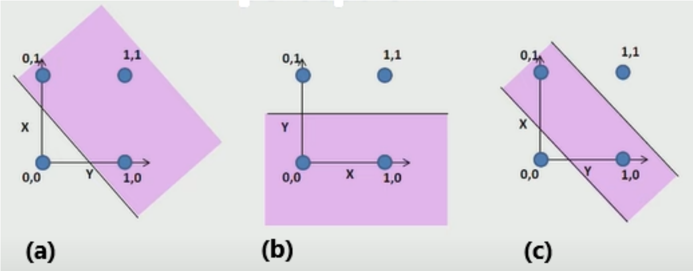


Figure 6.11: (a), (b) have linear separation while (c) does not. Figure taken from [RS20]

The real perceptron cannot act as a classifier unless the points belonging to the two classes are linearly separable. Figure 6.12 is an example of points that a single perceptron unit cannot separate. △

Note that using a single real perceptron unit with threshold activation to classify points from classes 0 and 1 is equivalent to finding a hyperplane passing through the origin that separates points from the two classes. In other words,

The perceptron can be used to as a classifier only if the data points at hand are **linearly separable**.

6.4 Learning a single perceptron

Definition 6.1. Two finite sets of points $\mathcal{A}$ and $\mathcal{B}$ in an m-dimensional space are called linearly separable if $n+1$ real numbers $w_1, w_2, \cdots, w_{n+1}$ exist such that every point $(x_1, x_2, \cdots, x_n)^T \in \mathcal{A}$ satisfies $\sum_{i=1}^{m} w_i x_i \geqslant w_{n+1}$ while every point $(x_1, x_2, \cdots, x_n)^T \in \mathcal{B}$ satisfies $\sum_{i=1}^{m} w_i x_i < w_{n+1}$.

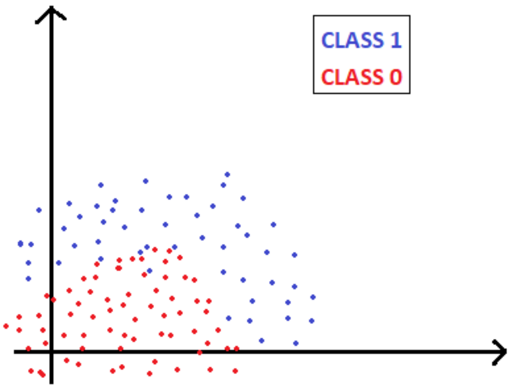


Figure 6.12: Real input values that are not linearly separable

Proposition 6.1. *Let $\mathcal{A}$ and $\mathcal{B}$ be two finite set of points in $\mathbb{R}^m$ such that points in $\mathcal{A}$ belong to class 0 while those of $\mathcal{B}$ belong to class 1. Then the set of points $\mathcal{A}$ and $\mathcal{B}$ are linearly separable iff they are absolutely linearly separable.*

Proof. Absolute linear separability implies linear separability follows directly from definitions. We are left to prove the converse.
The points in $\mathbb{A}$ and $\mathbb{B}$ are linearly separable implies that:

$$\sum_{i=1}^{m} a_i w_i \geqslant w_{m+1} \forall\ (a_1, \cdots, a_m)^T \in \mathcal{A} \text{ and } \sum_{i=1}^{m} b_i w_i \geqslant w_{m+1} \forall\ (b_1, \cdots, b_m)^T \in \mathcal{B}$$

Define ϵ and w' as follows:

$$\epsilon = \max\left\{ \sum_{i=1}^{m} w_i b_i - w_{m+1} \;\middle|\; (b_1, \cdots, b_m) \in \mathcal{B} \right\} \text{ and } w' = w_{m+1} + \frac{\epsilon}{2}$$

Now, for points $(a_1, \cdots, a_n) \in \mathcal{A}$

$$\begin{aligned} \sum_{i=1}^{m} w_i a_i - w_{m+1} \geqslant 0 &\implies \sum_{i=1}^{m} w_i a_i - \left(w' - \frac{\epsilon}{2}\right) \geqslant 0 \\ &\implies \sum_{i=1}^{m} w_i a_i - w' \geqslant -\frac{1}{2}\epsilon > 0 \\ &\implies \implies \sum_{i=1}^{m} w_i a_i > w' \end{aligned}$$

Similarly, it can be shown that all points $(b_1, \cdots, b_m)$ satisfy the following condition:

$$\sum_{i=1}^{m} w_i b_i < w' \forall\ (b_1, \cdots, b_m)^T \in \mathcal{B}$$

□

Perceptron learning

- **Start**: The weight vector $\tilde{w} = \tilde{w}_0$ is generated randomly. Set t denote the iteration number. Start by setting $t = 0$.
- **Test**: A vector $\tilde{x} \in \mathcal{A} \cup \mathcal{B}$ is selected randomly,
  - if $\tilde{x} \in \mathcal{A}$ and $\tilde{w}_t{}^T\tilde{x} > 0$, go to **Test**,
  - if $\tilde{x} \in \mathcal{A}$ and $\tilde{w}_t{}^T\tilde{x} \leqslant 0$, go to **Add**,
  - if $\tilde{x} \in \mathcal{B}$ and $\tilde{w}_t{}^T\tilde{x} < 0$, go to **Test**,
  - if $\tilde{x} \in \mathcal{B}$ and $\tilde{w}_t{}^T\tilde{x} \geqslant 0$, go to **Subtract**,
- **Add**: Set $\tilde{w}_{t+1} = \tilde{w}_t + \tilde{x}$ and replace t by $t+1$, go to **Test**.
- **Subtract**: Set $\tilde{w}_{t+1} = \tilde{w}_t - \tilde{x}$ and replace t by $t+1$, go to **Test**.

Geometric visualization of the perceptron learning algorithm Consider the vectors $\tilde{w}$ and $\tilde{x}$ in $\mathbb{R}^3$. Then the separating hyperplane is given by $w_1x_1 + w_2x_2 + w_3 = 0$. This implies that the separating hyperplane can be be identified by its unique normal vector $(w_1, w_2, w_3)^T$. Thus we want to find the plane or its unique normal $\tilde{w}$, given that we have two sets of points $\mathcal{A}$ and $\mathcal{B}$. Assume that the sets are linearly separable. The proposition 6.1 then implies that the two sets are absolutely linearly separable. Without loss of generality let the following hold,

$$\tilde{w}^T\tilde{x} > 0 \forall\ \tilde{x} \in \mathcal{A} \text{ and } \tilde{w}^T\tilde{x} < 0 \forall\ \tilde{x} \in \mathcal{B}$$

Observe that $\tilde{w}^T\tilde{x} > 0$ implies that the angle between the vectors $\tilde{w}$ and $\tilde{x}$ is acute while $\tilde{w}^T\tilde{x} < 0$ implies that the angle between the vectors is obtuse. Therefore all vectors in set $\mathcal{A}$ are at an acute angle α with the normal vector of the plane while all vectors in $\mathcal{B}$ are at an obtuse angle β with the hyperplane's normal as shown in Figure 6.13.

Perceptron learning algorithm starts with a randomly chosen $\tilde{w}_0$. If a vector $\tilde{x} \in \mathcal{A}$ is such that $\tilde{w}^T\tilde{x} < 0$. This means the hyperplane associated with $\tilde{w}_0$ does not accurately separate points in $\mathcal{A} \cup \mathcal{B}$. This is shown in Figure 6.14. The angle between the vectors is obtuse instead of being acute. The angle can be made "more acute" by pushing the vector $\tilde{x}$ towards the normal $\tilde{w}$. This can be done by updating our weights from $\tilde{w}$ to $\tilde{w} + \tilde{x}$. This update would imply that a new separator, defined by the normal $\tilde{w} + \tilde{x}$, is now there which corrects for the point $\tilde{x}$ we chose initially. Usng similar arguments, one can explain how updating $\tilde{w}$ to $\tilde{w} - \tilde{x}$ would correct for a point in $\mathcal{B}$ that is wrongly classified as being in $\mathcal{A}$.

Continuing this procedure, as mentioned in the algorithm above, would eventually give us a separator that correctly divides the points in $\mathcal{A}$ from points in $\mathcal{B}$. Notice that the weight vector is rotated one way or another every time the weight vector is updated.

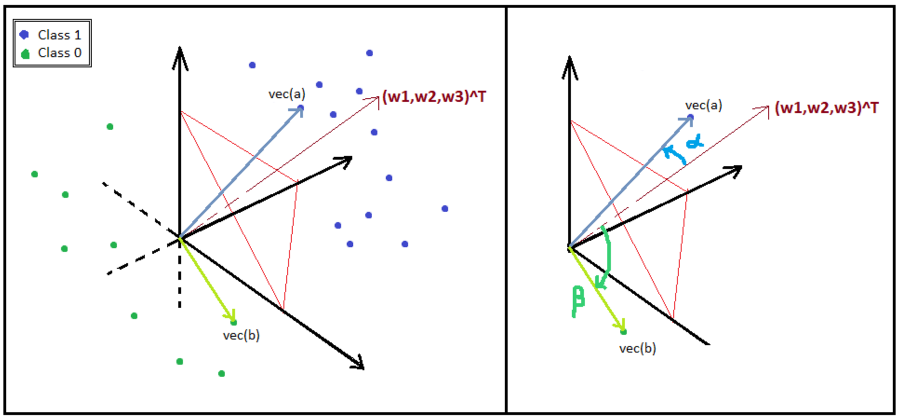


Figure 6.13: Geometric visualization of angles α and β

Remark 6.2. The plane drawn with red boundaries in Figures 6.14 and 6.13 has $(w_1, w_2, w_3)^T$ as the associated weight vectors.

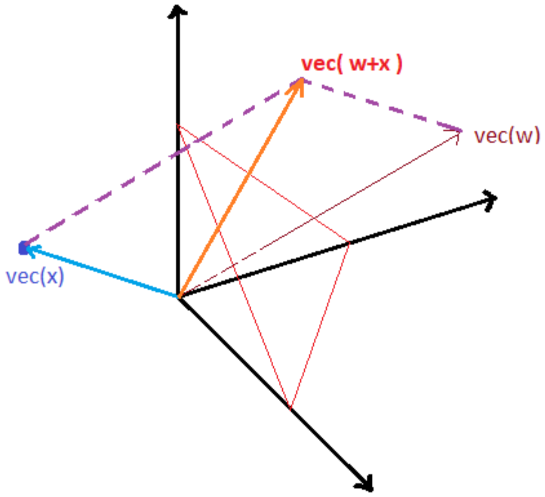


Figure 6.14: Adjusting weights

Convergence of the learning algorithm

The algorithm described above would be practically useful if the number of steps the algorithm takes to arrive at a solution is finite. The proposition below proves that the perceptron learning algoritm takes at most a finite number of steps to arrive at the right answer.

Proposition 6.2. *If the sets $\mathcal{A}$ and $\mathcal{B}$ are finite and linearly separable, the perceptron learning algorithm updates the weight vector $\tilde{w}_t$ a finite number of times.*

Proof. As usual we assume that $\mathcal{A}$ consists of all those $\tilde{x}$ such that $\tilde{x}^T\tilde{w} > 0$ and $\mathcal{B}$ consists of all those points which satisfy $\tilde{w}^T\tilde{x} < 0$. Without loss of generality, the following simplifications can be made:

- The sets $\mathcal{A}$ and $\mathcal{B}$ can be joined to give a single set $\mathcal{S} = \mathcal{A} \cup \mathcal{B}^-$ where $\mathcal{B}^-$ consists of negated elements of $\mathcal{B}$.
- The vectors in $\mathcal{S}$ can be normalized. If a weight $\tilde{w}$ ensures $\tilde{w}^T\tilde{x} > 0$, it holds for any other scaled version of $\tilde{x}$.
- The weight vector can also be normalized. We assume that a solution in the form of a linear separator exists and we denote the normalized solution vector as $\tilde{w}^*$.

Let $\tilde{w}_{t+1}$ be the weight vector at the end of $t+1$ iterations. Let $\tilde{s} \in \mathcal{S}$ such that it was classified incorrectly at the t^{th} iteration. This implies that the weight was updated from $\tilde{w}_t$ to $\tilde{w}_{t+1} = \tilde{w}_t + \tilde{s}$. Then the angle ρ between $\tilde{w}_{t+1}$ and $\tilde{w}^*$ is given by

$$\cos\rho = \frac{\tilde{w}^* \cdot \tilde{w}_{t+1}}{\sqrt{\tilde{w}_{t+1} \cdot \tilde{w}_{t+1}}}$$

Using the notation $\delta = \min_{\tilde{s}}\{\tilde{w}^*\tilde{s} \mid \tilde{s} \in \mathcal{S}\}$, we can derive the following:

$$\tilde{w}^* \cdot \tilde{w}_{t+1} \geqslant \tilde{w}^* \cdot \tilde{w}_t + \delta \geqslant \tilde{w}^* \cdot \tilde{w}_0 + (t+1)\delta \tag{6.7}$$

Since the weight $\tilde{w}_t$ was corrected, it implies that thay the term $\tilde{w}_t\tilde{s} \leqslant 0$. Therefore, it can be deduced that

$$||\tilde{w}_{t+1}||^2 \leqslant ||\tilde{w}_t||^2 + 1$$

Repeating the same prcedure backwards with respect to t gives us:

$$||\tilde{w}_{t+1}||^2 \leqslant ||\tilde{w}_0||^2 + (t+1) \tag{6.8}$$

Using the equations (6.7) and (6.8), the following can be derived easily:

$$\cos\rho \geqslant \frac{\tilde{w}^* \cdot \tilde{w}_0 + (t+1)\delta}{\sqrt{||\tilde{w}_0||^2 + (t+1)}} \tag{6.9}$$

The term on the right in (6.9) grows proportionally to $\sqrt{t}$. The term on the right can grow arbitrarily large, however, the term $\cos\rho$ is restricted to $[-1, 1]$. Therefore, the number of possible updates is atmost finite. □

6.5 Learning the multi-layer feed forward perceptron

Note that the perceptron learning algorithm we discuss in this section is the real perceptron. Before looking at the learning algorithm or how it works, it is important to understand some definitions and properties.

We assume in this section that the architecture of the feed forward neural network is given (i.e. we know the number of hidden layers required etc.) and can be used to model the function of our interest. It is clear that every multilayer perceptron or a feed forward neural network corresponds to a function whose parameters are all the weights and biases we define for each individual neuron in the network. So we are expected to identify the right set of weights and biases in order to enable the network to approximate the desired function. This process of determining the parameter values such that the network computes the desired function is called **learning the network**.

From the universal approximation theorem, it is clear that there always exists an MLP which can approximate any given function to arbitrary precision. Although this theorem proves the existance of MLPs arbitrarily "close" to the function of interest, it does not mention how the MLP can be constructed. Let $g(\tilde{x})$ be the function we want to model while the neural network architecture we are given corresponds to the function $f(\tilde{x};\tilde{w})$. We wish to find the right set of weights and biases such that f is a good approximate of g. Therefore the optimality criteria we choose, to compare the performance of f for different weight values is :

$$\widehat{W} = \arg\min_{\tilde{w}} \int_{\tilde{x}} \mathcal{D}(f(\tilde{x},\tilde{w})||g(\tilde{x}))\mathrm{d}\tilde{x} \tag{6.10}$$

In (6.10), we use the KL diveregence term i.e. $\mathcal{D}(f(\tilde{x},\tilde{w})$. The KL-divergence quantifies the error in using the function $f(\tilde{x};\tilde{w})$ to approximate $g(\tilde{x})$.

But there is a problem - $\mathbf{g}(\tilde{\mathbf{x}})$ is not known. This is because $g(\tilde{x})$ is the true classifier which we do not know. Since we do not know g, we **sample** different $\tilde{x}$ values and their corresponding $g(\tilde{x})$ values. This is nothing but collecting the **training sample**. So the problem now is to learn the function f we need using the training sample alone. To sum up, we would ideally want to optimize the network to represent g everywhere. Since this is not possible, we try to estimate a network that fits at the given set of labelled observations.

Remark 6.3. The bias is not often explicitly mentioned. It is implicitly assumed that the bias has been written as one of the weights corresponding to the input $x_{m+1=1}$.

The previous subsection dealt with how the weights and bias of a single perceptron unit can be learnt using the training data we have. But what about more complex decision boundaries? Consider, for example, the classification problem in 6.15. Although a single neuron cannot model the boundary, it is clear that multiple perceptrons can together model the boundaries of this dataset. The boundaries of the red data points are defined by edges of two polygons. Each polygonal side can be defined with the help of a perceptron as shown in 6.17 **only after the other points have been relabelled** as shown in 6.16. This relabelling of all points to identify the weights of a perceptron unit add a layer of complexity while trying learning MLP using single unit perceptrons.

We have seen earlier that trying to using a single perceptron unit as a classifier is equivalent to finding a hyperplane in $\mathbb{R}^m$ that divides the two sets of points in the data set. The problem is equivalent to finding the appropriate weights $\tilde{w}$ such that $\tilde{w} \cdot \tilde{x} \geqslant 0$ for all $\tilde{x}$ in belonging to one class while $\tilde{w} \cdots \tilde{x} < 0$ for all $\tilde{x}$ in the other class.

Problem with the threshold function: One might be tempted to use the ideas of single perceptron learning to construct more complex decision boundaries as well. However, the idea to update the weight vector in small steps until all points are classified correctly may not always be ideal as the number of iterations to reach the to reach the final step might be very large at times. This is because of the choice of taking threshold function as the activation function. Due to the threshold function, the overall function $f : \mathbb{R}^m \longrightarrow \{0, 1\}$ that the single perceptron unit represents will be a step function at 0. This implies that f has $\tilde{0}$ derivative (with respect to $\tilde{w}$, where $\tilde{w}$ is the weights vector) everywhere except at the point 0. This inturn implies that there will be instances where change in weights will not change the function f. This in turn implies an unchanged error value. Therefore, we have no idea of which direction to move our weight vector to in order to reduce the error. This is a problem as it can increase the number steps needed to stop the perceptron learning algorithm.

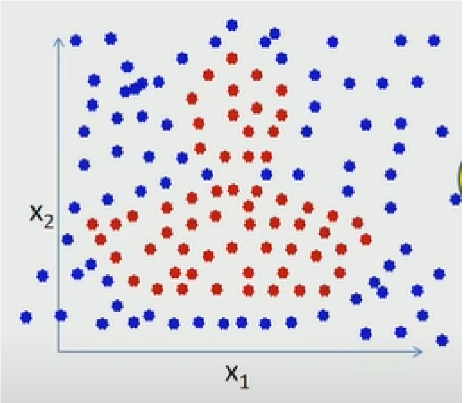


Figure 6.15: Data points with non-linear separation. Figure taken from [RS20].

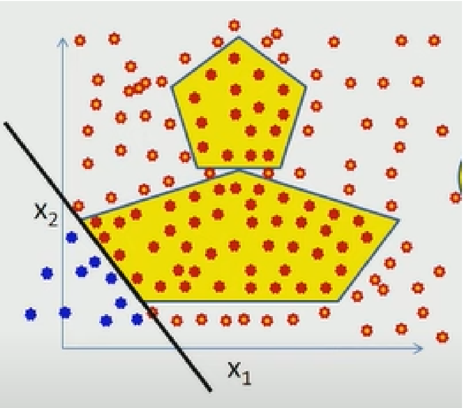


Figure 6.16: Relabelling the points to define hyperplane X1-X2. Figure taken from [RS20].

The problem with the single perceptron learning will be observed in multiple perceptrons as well if the same idea of updating weights is used with the help of a threshold function. This problem with the activation function can be solved by using

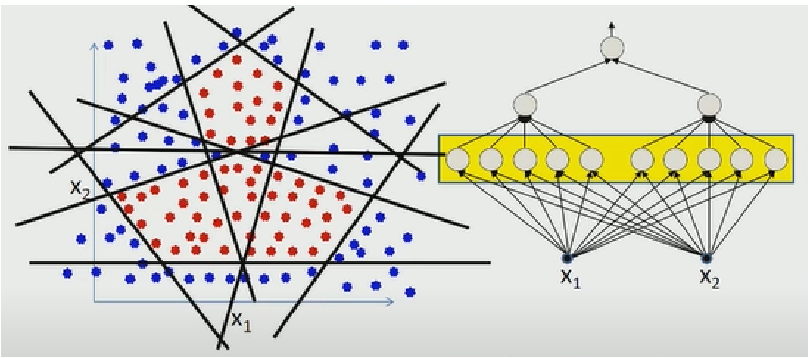


Figure 6.17: Combination of perceptrons can define the boundary. Figure taken from [RS20].

a continuous activation function like the sigmoid function. This function ensures that a change in weights leads to an increase or decrease in error (since the derivative is never zero for real inputs). This would enable us to update the parameters in a way that reduces the error.

Real perceptron cannot always model real world data: Real world data is not always linearly separable. Therefore the real perceptron cannot always model real world data. Understanding the interpretation of sigmoid activation function, which is given later, makes it more intuitive for us to accept sigmoid activation in real world settings where linear separation rarely exists.

6.5.1 Intrepretation of the sigmoid activation

Consider the one dimensional points in Figure 6.18. The two points have been drawn with a separation to show that one set of points belong to class 1 while the rest belong to class 0. Clearly, the two point, if plotted on the same real line, will overlap. As we go from left to right, the likelihood of finding a red dot increases. This likelihood is calculated based on a window around each point on the real line as shown in 6.19.

Therfore, the sigmoid can be considered as the function that **models the overlap** present among data points of different classes that we see in real data sets. The value of the sigmoid function at a given point can be considered as the **probability of a point belonging to class label 1** at a given value of input.

6.5.2 Perceptron with differentiable activation function

Consider the perceptron unit shown in figure 6.22. Since addition and sigmoid are differential functions whose composition corresponds to a given perceptron unit, it is clear that the function represented by a unit perceptron is **differentiable**. This implies that one can compute the change in y for small changes in input values or

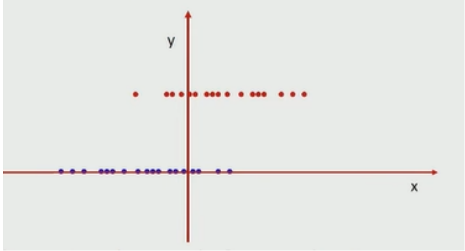


Figure 6.18: One dimensional points

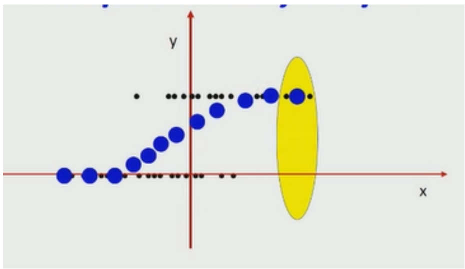


Figure 6.19: Interpretation of the sigmoid function

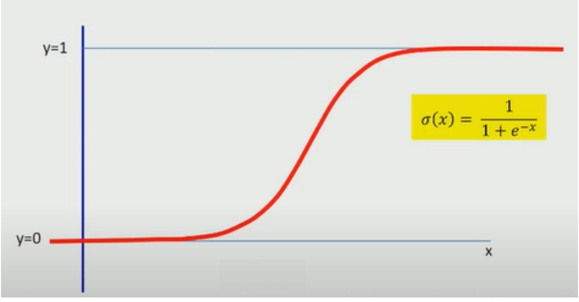


Figure 6.20: The Sigmoid function

Figure 6.21: All three figures were taken from [RS20].

weight values.

$$\frac{\mathrm{d}y}{\mathrm{d}z} = \sigma'(z) \tag{6.11}$$

$$\frac{\partial y}{\partial w_i} = \frac{\mathrm{d}y}{\mathrm{d}z} \times \frac{\partial z}{\partial w_i} = \sigma'(z)x_i \tag{6.12}$$

$$\frac{\partial y}{\partial x_i} = \frac{\mathrm{d}y}{\mathrm{d}z} \times \frac{\partial z}{\partial x_i} = \sigma'(z)w_i \tag{6.13}$$

As mentioned earlier, if we knew the true classifier g we need to model then the optimization problem of our interest would be:

$$\widehat{W} = \arg\min_{\tilde{w}} \int_{\tilde{x}} \mathcal{D}(f(\tilde{x}, \tilde{w})||g(\tilde{x}))\mathrm{d}\tilde{x}$$

Ideally the weights should be such that $f = g$. However, this is not possible because we do not know the exact complexity of the architecture required when we start. So the difference or the error term can never be made 0. Therefore, we try to ensure that the regions in the input space that are more likely to occur have lower error rates than those inputs which are less likely to occur. This is accounted for by multiplying to the divergence term, the probability of observing that input. In other words, X is treated

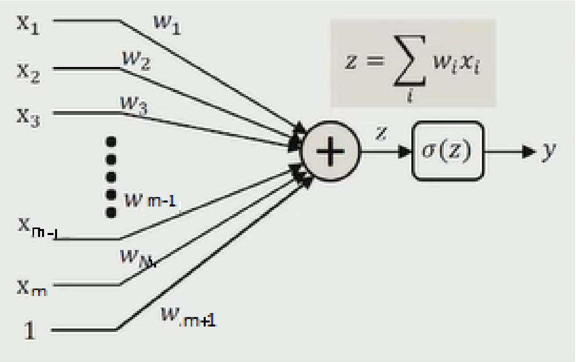


Figure 6.22: Figure taken from [RS20].

as a random variable. Mathematically, our optimization problem now converts to:

$$\widehat{W} = \arg\min_{\tilde{w}} \int_{\tilde{x}} \mathcal{D}(f(\tilde{x},\tilde{w})||g(\tilde{x}))P(x)\mathrm{d}\tilde{x} = \arg\min_{\tilde{w}} \mathbb{E}[\mathcal{D}(f(\tilde{X},\tilde{w})||g(\tilde{X}))] \tag{6.14}$$

Again, since g is not known to us, the above expected error or **risk** is estimated as follows:

$$\mathbb{E}[\mathcal{D}(f(\tilde{X},\tilde{w})||g(\tilde{X}))] \approx \frac{1}{n}\sum_{i=1}^{n}\mathcal{D}(f(\tilde{x_i},\tilde{w})||d_i) \text{ where } d_i = \text{ true class label for } x_i \tag{6.15}$$

Note that the term $\mathbb{E}[\mathcal{D}(f(\tilde{X},\tilde{w})||g(\tilde{X}))]$ is the **expected error or risk** i.e. it is the mean error over the entire input space. The estimate mentioned in equation (6.15) on the other hand is the **empirical risk**.

Assume there are L different classes and the neural network has L nodes in the output layer. Let $y_1, \cdots, y_L$ denote the L outcomes we get using our neural network while $d_1, d_2, \cdots, d_L$ denote the true outputs. For a given input $(x_1, \cdots, x_m)^T$, an output $(y_1, \cdots, y_L)^T$ with $y_i = 0$ for all i except $i = i_0$ implies that the input x has the class label j. Since this is too idea to happen in real world, we get a vector of probabilities in $\tilde{y}$ that add up to 1. The distribution corresponding to the constant function d_i is that the probability of d_i occuring is one while every other point has zero probability to occur.

The KL divergence can be replaced with cross-entropy Let q be the true density which we are trying to approximate using f. Then it can be shown that:

$$\begin{aligned} KL(q||f) = \int q(x)\log\left(\frac{q(x)}{f(x)}\right)\mathrm{d}x &= -\int q(x)\log f(x)\mathrm{d}x + \int q(x)\log(q(x))\mathrm{d}x \\ &= C(q,f) + H(q) \end{aligned}$$

Clearly, since the term $H(q)$ is independent of f, it is clear that minimizing KL-divergence between f and q is equivalent to minimizing cross entropy $C(q,f)$ between q and f.

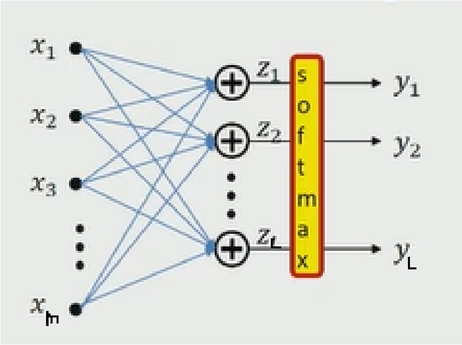


Figure 6.23: Neural network with softmax activation. Figure taken from [RS20].

A couple of important points worth mentioning again are stated below:

- The function we wish to truly optimize over all possible weight vectors is the expected risk. However, since the function g is unknown to us, we minimize the emperical risk estimate of the expected risk.
- Making the activation function differentiable enables us to get a network which represents a differentiable function. This sets up a scenario where we can exploit gradient decent techniques to optimize the functions of our interest.

Remark 6.4. Note that the $\tilde{x}_i$ in (6.15) denotes the i^{th} observation. We assume that out training set consists of n labelled observations. Any given observation is generally denoted by $\tilde{x} = (x_1, x_2, \cdots, x_m)^T$. However, $\tilde{x}_i = (x_{i1}, x_{i2}, \cdots, x_{im})^T$.

Softmax activation

The activation function we saw is a scalar activation function i.e. its output is a scalar. There exist **vector activations** as well which take in a m dimensional input to give out a L dimensional output $\tilde{y}$. An example of such a vector activation function is the **softmax**. Softmax vector activation works as shown in Figure 6.23.

In Figure 6.23, we denote the weight of the edge between x_i and z_j to be w_{ij}. Corresponding to each z_i, there is a bias b_i such that z_i denotes the affine combination of $x_i's$ and b_i.

$$z_i = \sum_{j=1}^{m} w_{ji}x_j + b_i \text{ and } y_i = \frac{\exp z_i}{\sum_{j=L}^{m} \exp z_j} \tag{6.16}$$

Observe that the each y_i can be viewed as the probability of the given input belonging to class one. The only difference between a scalar activation function and a softmax

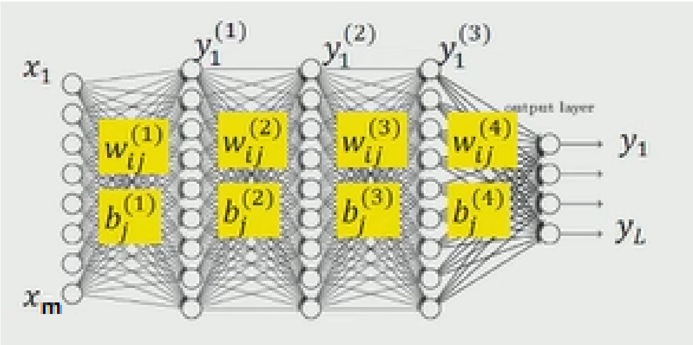


Figure 6.24: Neural network notations. Figure taken from [RS20].

activation lies in how a change in weights of the network change the final values of the network.‘In case of using a sigmoid for each output neuron, a change in weights of a given neuron N will only effect the output neuron associated with neuron N. In the case of softmax, a change in weights of the network can change all output values due to the constraint $\sum_{i=1}^{L} y_i = 1$. If the number of possible classes is L, the we can try to calculate a network that outputs the i^{th} standard basis vector if the input belongs to the i^{th} class. However, this is quite an ideal scenario to occur in real word models. Instead, what we get $(y_1, \cdots, y_L)^T$ can be viewed as a **vector of probabilities**.

Since $(y_1, \cdots, y_L)$ can be treated as a vector of probabilities, therefore the distribution corresponding to Y while calculating KL divergance is $P(x \in \text{class } i) = y_i$

Some points to note

- Given a neural network, it can either have a single output neuron or multiple output neurons. The single output scenario is often used for binary classification problems. For multi-class problems, multiple outputs are required.
- In case of multi-class problems, each output neuron can be associated with a univariate sigmoid or the final layer can be associated with a vector activation function to get a vector output.

Notation

The input layer is called the zeroth layer. The outputs coming out of the i^{th} hidden layer are denoted as $\{y_1^{(i)}, y_2^{(i)}, \cdots, y_{l_i}^{(i)}\}$. The number of nodes in the i^{th} hidden layer are denoted by l_i. There are L output nodes which give the outputs $y_1, y_2, \cdots, y_L$. The weight of the edge connecting the i^{th} unit of $k-1$ layer and j^{th} unit of the k^{th} layer is given by $w_{ij}^{(k)}$. Bias corresponding to the j^{th} unit of the k^{th} layer is denoted by $b_j^{(k)}$.

Problem set-up We use a differentiable activation function so that the **gradient descent algorithm** can be used to minimize the empirical risk.

- We are given a training set of input-output pairs $(\tilde{X}_1, d_1), (\tilde{X}_2, d_2), \cdots, (\tilde{X}_n, d_n)$.
- We wish to minimize the following function:

$$\text{Risk } = R(\tilde{w}) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{D}(f(x_i; \tilde{w}) \mid\mid d_i)$$

Using gradient descent algorithm

- Initialize all weights and biases $w_{ij}^{(k)}$ and $b_i^{(j)}$ for all possible i, j, k.
- For each layer k and for all possible i, j, update weight as follows:

$$w_{i,j}^{(k)} = w_{i,j}^{(k)} - \eta \frac{\mathrm{d}R}{\mathrm{d}w_{i,j}^{(k)}} \tag{6.17}$$

Repeat the above rules until the risk function converges.The term η is called the **learning rate**. That is because it quantifies how big a step we have to take in the direction opposite to the gradient. Clearly,

$$\frac{\mathrm{d}R}{\mathrm{d}w_{i,j}^{(k)}} = \frac{1}{n}\sum_{t=1}^{n} \frac{\mathrm{d}R(y_t, d_t)}{\mathrm{d}w_{i,j}^{(k)}} \text{ where } y_t = \text{ value estimated by the network}$$

Using basic calculus and following the notations carefully helps us apply the gradient descent algorithm to find the appropriate weights that reduce the risk function. This technique of using gradient descent to arrive at a minimum is called the **Back-propagation algorithm**.

Problems associated with the Back-propagation algorithm

- Gradient descent doesn't always take us to the global minima. If we initialize our weights to a point that is in the vicinity of a local minima, then the gradient descent algorithm leads us to the local minima.
- It is quite common for the empirical risk function to have saddle points. Therefore, it might happen that we end up at a saddle point instead of a minimum while using the gradient descent algorithm.
- If we use univariate sigmoid functions as activation functions for each of the neurons in the output layer, then it is not possible for us to interpret the output vector we get. Since the output is not always the exact value (class label) we want, there is no way of interpreting the outputs that might be some values strictly between 0 and 1.

- Adding an extra data point into the training set does not alter the error function much. Therefore, adding an additional point doesn't significantly change the resulting neural network. This can be a problem if the data point added is quite important and must therefore change the structure of the network. Otherwise, this robustness is good because we do not want to entertain any additional noisy points that are added to our training set.

6.6 Data Analysis

In this section we use the data set that we used in the classification tree chapter. The details about this data set can be found in Chapter 5. However, to summarize, the data set we are considering is called the "data_banknote_authentication" data set that has been taken from the UCI machine learning repository. The dimension of the data set 1372 × 5. Each data point in this dataset corresponds to a bank note. Four continuous variables associated with each of the bank note is given. The fifth variable gives the true class label of each data point or bank note. Class label 0 implies that the note is fake while a value of 1 implies that it is a real bank note. We now construct a neural network of one hidden layer in order to use it as a classifier in the future.

```
> test = data_banknote_authentication[753:772,]
> dim(test)
[1] 20  4
> train = data_banknote_authentication[-c(753:772),]
> dim(train)
[1] 1352    5
```

Install the package "neuralnet" in order to get functions that help construct neural networks.

```
> library(neuralnet)
> nn = neuralnet(class ~., data = train, hidden = 3, startweights = NULL,
algorithm = 'backprop', err.fct = 'ce', act.fct = 'logistic',
linear.output = FALSE, learningrate = 0.1)
> plot(nn)
```

The code "hidden = 3" implies that we want a neural network with a single hidden layer such that this hidden layer consists of three neurons. If we wanted a neuron with two hidden layers, then the command we must use is "hidden = (x, y)". The variables x and y denote the number of neurons we want in the first and second hidden layers respectively. Choosing "startweights = NULL" implies that we wish to start with an arbitrary choice of weights. The term "err.fct = 'ce"' implies that the function we wish to optimize uses cross-entropy. The function "neuralnet()" can construct both neural network classifiers and regressors. Specifying that the optimizing function uses cross entropy implies that we wish to construct a neural net classifier. Choosing "linear.output = FALSE" implies that we wish to have activation functions associated with neurons of the output layer as well. "linear.output = TRUE" leaves the output linear without letting it go through the activation function.

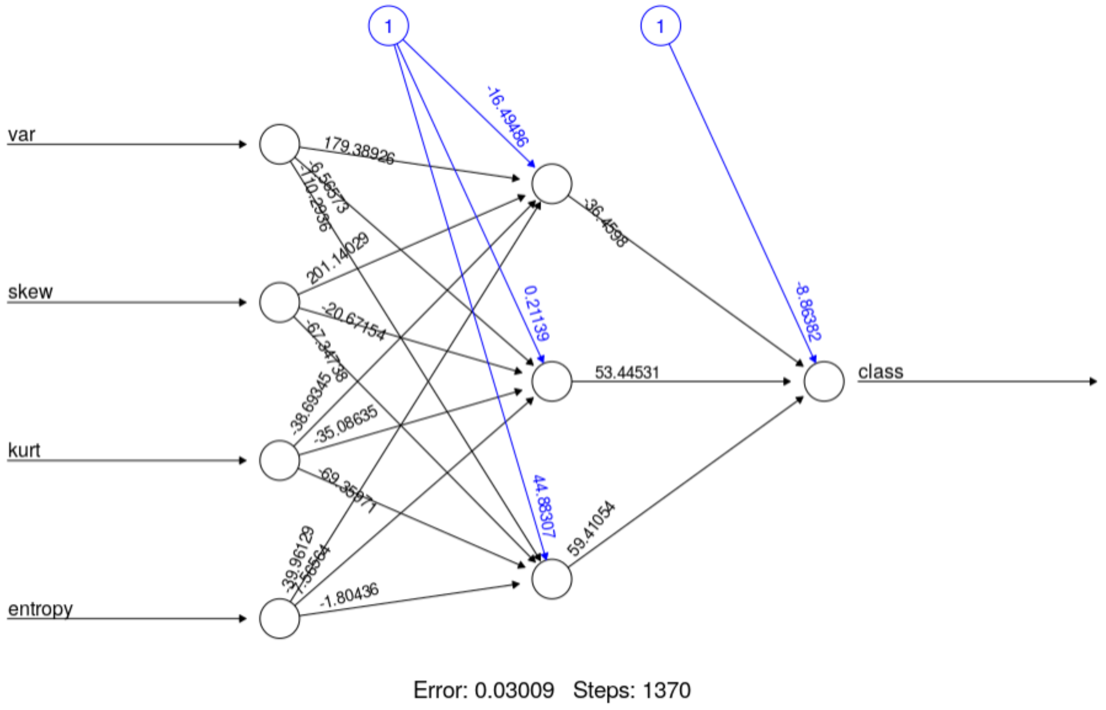


Figure 6.25: Neural network associated with the bank note dataset

The resulting neural network associated with this data set is given in figure 6.25. Observe that the nosed and edges associated with the bias term are given in blue. The "neuralnet()" function used the gradient descent algorithm to arrive at the a minima. At this minima, the error value is 0.03009. This error value along with the number of steps it took to reach this minima is mentioned in the figure. The weight values had to be updated 1370 times in order to reach this point of optimum.

Everytime we run the "neuralnet()" function, we get a different set of weights assuming that we keep the architecture of the network fixed. This is because we set the weights arbitrarily and then use the gradient descent to reach a point of minima. Therefore, the neural network we get and its associated error value are different every time we run the "neuralnet()" function code. I ran the code multiple times until I could get a neural network with significantly low error.

```
> output = compute(nn,test)
> output$net.result
           [,1]
```

```
753 2.071104e-20
754 1.413936e-04
755 2.071104e-20
756 2.071104e-20
757 2.071104e-20
758 2.071104e-20
759 2.071104e-20
760 2.071104e-20
761 2.071104e-20
762 2.071104e-20
763 9.999992e-01
764 1.000000e+00
765 1.000000e+00
766 1.000000e+00
767 1.000000e+00
768 1.000000e+00
769 1.000000e+00
770 9.999992e-01
771 1.000000e+00
772 1.000000e+00
```

Clearly, the neural network predicts all test set points correctly. From the error function value itself, it is clear that the given neural network we have can act as a very good classifier.

Bibliography

[Ana07] Venkat Anantharam. Jointly gaussian random variables. Lecture notes for EECS 223 course, Spring 2007. `https://people.eecs.berkeley.edu/~ananth/223Spr07/jointlygaussian.pdf`. Last visited on 7 Feb 2021.

[And03] T. W. Anderson. *An introduction to multivariate statistical analysis*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Inc., Hoboken, New Jersey, 3 rd edition, 2003.

[Bar06] Randal J. Barnes. Matrix differentiation (and some other stuff). Lecture notes for CE 8361 course, Spring 2006. `https://atmos.uw.edu/~dennis/MatrixCalculus.pdf`. Last visited on 7 Feb 2021.

[BFOS48] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and regression trees*. Chapman & Hall/CRC, Florida, 1st edition, 1948.

[BN12] Mohamed H. Bakr and Mohamed H. Negm. Chapter three - modeling and design of high-frequency structures using artificial neural networks and space mapping. In M. Jamal Deen, editor, *Silicon-Based Millimeter-wave Technology*, volume 174 of *Advances in Imaging and Electron Physics*, pages 223–260. Elsevier, 2012.

[Bur10] Christopher J. C. Burges. Dimension reduction: A guided tour. *Foundations and Trends in Machine Learning*, 2(4):275–365, 2010. `http://dx.doi.org/10.1561/2200000002`.

[BW20] Carl T. Bergstorm and Jevin D. West. *Calling bullshit*. Random House, New York, 1st edition, 2020.

[CBRO] Vittorio Castelli, Mark Brodie, Irina Rish, and Daniel Oblinger. The proof of the optimality of the bayes decision rule. Lecture notes for ELEN E 6880 Statistical Pattern Recognition, Summer 2003, Columbia University. `https://www.ee.columbia.edu/~vittorio/Lecture1.html`. Last visited on 12 July 2021.

[Chi19] Ted Chinburg. Shannon's theorem. Lecture notes for MATH 280 course at University of Pennsylvania, Spring 2019. https://www2.math.upenn.edu/~ted/280S19/math280.html. Last visited on 12 July 2021.

[DG17] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[Fid16] Sanja Fidler. Multiclass classification. Lecture notes for CSC 411, Spring 2016, University of Toronto, 2016. https://www.cs.utoronto.ca/~fidler/teaching/2015/slides/CSC411/07_multiclass.pdf. Last visited on 12 July 2021.

[Gho15] Ali Ghodsi. Statistical learning classification. Youtube lecture videos 1-4 for STAT 441/841 course at University of Waterloo, Fall 2015. https://www.youtube.com/watch?v=L-pQtGm3VS8&list=RDCMUCKJNzy_GuvX3SAg3ipaGa8A&index=3. Last visited on 12 July 2021.

[Ize08] Alan J. Izenman. *Modern Multivariate Statistical Techniques.* Springer Series in Statistics. Springer-Verlag, New York, 1 st edition, 2008.

[Kha19] Mitesh M. Khapra. Mcculloch pitts neuron, thresholding logic, perceptrons, perceptron learning algorithm and convergence, multilayer perceptrons (mlps), representation power of mlps. Lecture notes for CS7015 course at IITM, Spring 2019. https://www.cse.iitm.ac.in/~miteshk/CS6910.html. Last visited on 12 July 2021.

[Koc14] Inge Koch. *Analysis of Multivariate and High-Dimensional Data.* Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, New York, 1 st edition, 2014.

[Lee14] Seungchul Lee. Ellipse and gaussian distribution. Lecture slide from the "machine learning" course taught at Pohang University of Science and Technology, 2014. https://iai.postech.ac.kr/teaching/machine-learning. Last visited on 12 July 2021.

[LM13] Erik G. Learned-Miller. Entropy and mutual information. Lecture notes for CS670 course at University of Massachusetts, Amherst, Fall 2013. https://people.cs.umass.edu/~elm/Teaching/670_F13/. Last visited on 12 July 2021.

[Mas11] David Massey. Multivariable taylor polynomials and series. Lecture video from youtube uploaded by The Worldwide Center of Mathematics, 2011. https://www.youtube.com/watch?v=4-ZYyg3bRvs. Last visited on 12 July 2021.

[OR] Penn State University Open and Affordable Educational Resources. Tree-based methods. Lecture notes available online for the course STAT 508: Applied Data Mining and Statistical Learning. https://online.stat.psu.edu/stat508/lesson/11.

[Ora08] Oracle. Oracle data mining. Oracle, 2008. https://docs.oracle.com/cd/B28359_01/datamine.111/b28129.pdf. Last visited on 12 July 2021.

[Roj96] Raul Rojas. *Neural Networks.* Springer Series in Statistics. Springer-Verlag, Berlin, 1996.

[RS20] Bhiksha Raj and Rita Singh. Introduction to deep learning. Lecture notes and videos for 11-785 course at CMU, Fall 2020. https://deeplearning.cs.cmu.edu/F20/index.html. Last visited on 12 July 2021.

[RTSH08] C. R. Rao, H. Toutenburg, Shalabh, and C. Heumann. *Linear Models and Generalizations.* Springer Series in Statistics. Springer-Verlag, Berlin Heidelberg, 3 rd edition, 2008.

[Sha48] C. E. Shannon. A mathematical theory of communication. *Bell Labs Technical Journal*, 27(3):379–423, 1948. http://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf.

[Shl14] Jonathon Shlens. A tutorial on principal component analysis. arXiv preprint, 2014. https://arxiv.org/abs/1404.1100. Last visited on 7 Feb 2021.

[SL03] George A. F. Seber and Alan J. Lee. *Linear Regression Analysis.* Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., Hoboken, New Jersey, 2 nd edition, 2003.

[Ste] Hans Sterk. Geometry in architecture and building. Lecture notes for 2DB60 Meetkunde voor Bouwkunde (Geometry for Engineering), April 2014, Eindhoven University of Technology. https://www.win.tue.nl/~sterk/Bouwkunde/. Last visited on 12 July 2021.

[TG21] Web Technology and Information Systems Group (Webis Group). Machine learning. Lecture notes and videos for “Machine Learning” course, Spring 2021. https://webis.de/lecturenotes.html#machine-learning. Last visited on 12 July 2021.

[WB] Jevin West and Carl Bergstrom. p-values and the prosecutor’s fallacy. Youtube video part of course INFO 270 / BIOL 270 offered at University of Washington. https://www.youtube.com/watch?v=eesUdFlYMh8&t=400s. Last visited on 12 July 2021.