Universidade Estadual de Campinas
Faculdade de Ciências Aplicadas

Alejandra Inga Quezada

Using optimization models to achieve solutions in classification and clustering techniques

Utilização de modelos de optimização para obter soluções em técnicas de classificação e agrupamento

Limeira
2020

Alejandra Inga Quezada

Using optimization models to achieve solutions in classification and clustering techniques

Utilização de modelos de optimização para obter soluções em técnicas de classificação e agrupamento

Dissertation presented to the School of Applied Sciences of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Production and Manufacture Engineering, in the area of Operations Research and Processes management.
Dissertação apresentada à Faculdade de Ciências Aplicadas como parte dos requisitos exigidos para a obtenção do título de Mestra em Engenharia de Produção e de Manufatura na área de Pesquisa Operacional e Gestão de Processos.

Orientador: Prof. Dr. Washington Alves de Oliveira

Este exemplar corresponde à versão final da dissertação defendida pela aluna Alejandra Inga Quezada e orientada pelo Prof. Dr. Washington Alves de Oliveira.

Limeira
2020

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Faculdade de Ciências Aplicadas
Renata Eleuterio da Silva - CRB 8/9281

In41u Inga Quezada, Alejandra Jimena, 1994-
Using optimization models to achieve solutions in classification and clustering techniques / Alejandra Jimena Inga Quezada. – Limeira, SP : [s.n.], 2020.

Orientador: Washington Alves de Oliveira.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Ciências Aplicadas.

1. Análise por agrupamento. 2. Classificação. 3. Programação matemática. 4. Aprendizado de máquina. I. Oliveira, Washington Alves de, 1977-. II. Universidade Estadual de Campinas. Faculdade de Ciências Aplicadas. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Utilização de modelos de optimização para obter soluções em técnicas de classificação e agrupamento
Palavras-chave em inglês:
Cluster analysis
Classification
Mathematical programming
Machine learning
Área de concentração: Pesquisa Operacional e Gestão de Processos
Titulação: Mestra em Engenharia de Produção e de Manufatura
Banca examinadora:
Washington Alves de Oliveira [Orientador]
Leonardo Tomazeli Duarte
Romis Ribeiro de Faissol Attux
Data de defesa: 06-11-2020
Programa de Pós-Graduação: Engenharia de Produção e de Manufatura

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0003-1436-3913
- Currículo Lattes do autor: http://lattes.cnpq.br/8293870466255542

COMISSÃO JULGADORA - DISSERTAÇÃO DE MESTRADO

Candidato: Alejandra Inga Quezada
RA: 211717
Data de Defesa: 06 de novembro 2020

Título da dissertacão: ***"Using optimization models to achieve solutions in classification and clustering techniques".***
Título da dissertacão: ***"Utilização de modelos de optimização para obter soluções em técnicas de classificação e agrupamento".***

Prof. Dr. Washington Alves de Oliveira (Presidente)
Prof. Dr. Leonardo Tomazeli Duarte
Prof. Dr. Romis Ribeiro de Faissol Attux

A Ata de Defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no SIGA (Sistema de Fluxo de Dissertação) e na Secretaria de Pós-Graduação da Faculdade de Ciências Aplicadas.

Dedicated to my grandparents: Nelly, Sofia, Ladislao, Juan, and Maria.

Acknowledgements

I would like to thank my parents, Humberto and Ada, for the support and love during the challenges and my little brother, Beto, for being the reason to persevere. Although she is not physically with me, I am very thankful to my lovely grandmother Nelly, who was like a second mother to me. Further, I would like to express my deep gratitude to my supervisor Prof. Washington for his patient guidance, support, and encouragement which has been invaluable throughout this study. I am also thankful to my friends in Perú whose always cheered on my success and to my friends at the CPO laboratory for the support and companionship from the first time I started the master's program. Finally, I would like to express my sincere gratitude to EXTECAMP-UNICAMP for providing financial support.

Abstract

This dissertation aims to study some techniques for handling large scale datasets to extract representative information from the use of mathematical programming. The structural patterns of data provide pieces of information that can be used to classify and cluster them through the optimal solution of specific optimization problems. The techniques used could be confronted with machine learning approaches to supply new numerical possibilities of resolution. Computational tests conducted on two case studies with real data (practical experiments) validate this research. The analyzes are done for the well-known database on the identification of breast cancer tumors, which either have a malignant or have a benign diagnosis, and also for a bovine animal database containing physical and breed characteristics of each animal but with unknown patterns. A binary classification based on a goal programming formulation is suggested for the first case study. In the study conducted on the characteristics of bovine animals, the interest is to identify patterns among the different animals by grouping them from the solutions of an integer linear optimization model. The computational results are studied from a set of descriptive statistical procedures to validate this research.

Keywords: Clusterization,classification,mathematical programming,machine learning

Resumo

Esta dissertação tem como objetivo estudar algumas abordagens de manipulação de bancos de dados em larga escala com o objetivo de extrair informações representativas a partir do uso de programação matemática. Os padrões estruturais dos dados fornecem informações que podem ser usadas para classificá-los e agrupá-los por meio da solução ótima de problemas específicos de otimização. As técnicas utilizadas podem ser confrontadas com abordagens de aprendizado de máquina para fornecer novas possibilidades numéricas de resolução. Testes computacionais conduzidos em dois estudos de caso (dados oriundos de experimentos práticos) validam esta pesquisa. As análises são conduzidas sobre um conjunto de dados relacionados com a identificação de tumores de câncer de mama, com diagnóstico maligno ou benigno, e um banco de dados de animais bovinos que fornecem características físicas e de raça de cada animal, porém sem um padrão previamente conhecido. Uma classificação binária com base em um modelo matemático de programação de metas é usado para o primeiro estudo de caso. No estudo conduzido sobre as características dos animais bovinos, o interesse é identificar padrões entre os diversos animais ao agrupá-los por meio da análise das soluções de um modelo de otimização linear com variáveis inteiras. Os resultados computacionais são estudados a partir de um conjunto de procedimentos estatístico descritivo para validar o estudo proposto.

Palavras-chaves: Clusterização, classificação, programação matemática, aprendizado de máquina

List of Figures

1.1 Interactions between machine learning and optimization. Adapt from Song et al. (2019) 26

1.2 Principal components for two-dimensional subspace 31

1.3 PCA for two-dimension data space and MSE approach. (Bishop 2006) 34

2.1 Classification model with two output labels. Adapt from Alpaydin (2010) 36

2.2 Support vector machine for two classes of data. Adapt from Zaki & Meira Jr. (2014) 39

2.3 Dendrogram illustrating a hierarchical clustering. Adapt from Alpaydin (2010) . . 46

3.1 Illustration of a solution for the classification basic GP model 49

3.2 Illustration of a solution for the classification GP model with standard preference functions 51

3.3 Illustration of a solution for the classification GP model with non-standard preference functions 52

3.4 Data and optimal solution for $n = 30$ and $d = 2$ (Gnagi & Baumann 2017) 54

3.5 Optimal solution graphic with the reduced data (Gnagi & Baumann 2017) 55

4.1 Data set used by Freed & Glover (1981) 59

4.2 GP1 model results considering four tests from data set of Freed & Glover (1981) . 60

4.3 GP2 model results considering four tests from data set of Freed & Glover (1981) . 61

4.4 ROC curves for Tests B,C, and D 62

4.5 Data set used by Freed & Glover (1986) 63

4.6 GP1 model results considering four tests from data set of Freed & Glover (1986) . 65

4.7 GP2 model results considering four tests from data set of Freed & Glover (1986) . 66

4.8 ROC curves for Tests B, and D 67

4.9 GP3 model results considering two tests from data sets DB1 and DB2, and SR or RS methods 68

4.10 Image of a malignant breast with nuclei cells analyzed with *Xcyt*. (Mangasarian et al. 1970) 69

4.11 Pairwise plot of the BCW database 72
4.12 Correlation heatmap of the BCW database for 7 features 73
4.13 Correlation heatmap of the BCW database for 9 features 74
4.14 Correlation heatmap of BCW database for other 9 features 75
4.15 Correlation heatmap of the BCW database for 5 features 76
4.16 Average accuracy level and error rate for the training set 78
4.17 Average accuracy level and error rate for the validation set 78
4.18 Comparative graphic of accuracy level among the training, validation and test sets . 79
4.19 Projected data variance versus number of principal components for the PCA method 80
4.20 Average accuracy level and error rate for the training set obtained from principal components 81
4.21 Average accuracy level and error rate for the validation set obtained from principal components 81
4.22 Comparative graphic of accuracy level for the training, validation, and test sets obtained from principal components 82
4.23 Three-dimensional plot of the bovine animal database 83
4.24 Pairwise plot of bovine animal database 84
4.25 Correlation heatmap of the bovine animal database with 102 observations 84
4.26 Evaluation scores for the Euclidean distance metric 86
4.27 Evaluation scores for the Manhattan distance metric 87
4.28 Evaluation scores for the Chebyshev distance metric 88
4.29 Three-dimensional plot of the bovine animal database 89
4.30 Specific references of a cattle to localize the ECC and Frame scores 89
4.31 Correlation heatmap of the bovine animal database with 526 observations 90
4.32 Evaluation score for the Euclidean distance metric in Test 1 92

List of Tables

1.1 Methods of operations research. Adapt from Rao (2009) 18

4.1 Test scenarios and their respective description 58
4.2 Data used in Figures 4.1 and 4.2 59
4.3 Results from the GP1 model illustrated in Figure 4.2 60
4.4 Results from the GP2 model illustrated in Figure 4.3 61
4.5 Data used in Figures 4.5 and 4.6 63
4.6 Results from the GP1 model illustrated in Figure 4.6 64
4.7 Results from the GP2 model illustrated in Figure 4.7 64
4.8 Description of penalty internal weights for the GP3 model 66
4.9 Penalty internal weights obtained via RS and SR methods 67
4.10 Results from the GP3 model illustrated in Figure 4.9(a) 68
4.11 Results from the GP3 model illustrated in Figure 4.9(b) 68
4.12 Descriptive statistic of the BCW database for 5 features 70
4.13 Descriptive statistic of the BCW database for 15 features 70
4.14 Descriptive statistic of the BCW database for 10 features 71
4.15 Penalty internal weights obtained via RS method for the GP3 model 73
4.16 Confusion matrix for the BCW database 77
4.17 Indicator performance per model 79
4.18 Indicator performance per model obtained from principal components 80
4.19 Descriptive statistic of the bovine animal database with 102 observations 83
4.20 Descriptive statistics of bovine animal database with 526 observations 90
4.21 Evaluation scores and running times 91
4.22 Average ECC, Frame and Weight in each cluster 92
4.23 Optimal cattle weight according to the Frame score 93

Contents

Introduction **14**

1 Preliminaries **17**

1.1 Optimization 17

1.2 Mathematical programming 18

1.3 Goal programming 20

1.4 Machine learning 24

1.5 Connection between optimization and machine learning 26

1.6 Dimensionality reduction 27

1.6.1 Principal component analysis method 28

2 Summary of classification and clustering **35**

2.1 Classification 35

2.1.1 Perceptron algorithm 37

2.1.2 Support vector machine 38

2.2 Clustering 42

2.2.1 K-means clustering 43

2.2.2 Hierarchical clustering 45

3 Methodology **48**

3.1 Classification method based on goal programming 48

3.2 Clustering method based on integer programming 53

4 Numerical experiments **57**

4.1 Numerical experiments for classification 57

4.1.1 Setting the parameters 58

4.1.2 Case of study: Breast Cancer 69
4.2 Numerical tests for clustering 82
4.2.1 Setting the parameters 82
4.2.2 Case of study: Bovine animals database 88

5 Final remarks and some clues for future researches **94**
5.1 Conclusions 94
5.2 Summary of the obtained results 95
5.3 Future researches 96

Bibliography **97**

Introduction

As technology advances, it becomes easier to capture and save different human activities such as buying a product, renting a movie, or going to the cinema. The use of social media or smart devices (smartphones, tablets, personal computers, and others) allows corporations to investigate the general characteristics of users and establish their consumption profiles. The information volume converts to data that are stored in large databases, usually used by companies to understand their customers' preferences. Thus new products/goods and services are adjusted and offered for costumers according to these preferences.

With the purpose of understanding and extracting relevant information from data, machine learning techniques have arisen in the last decades as a powerful tool for this purpose. Alpaydin (2010) explains that machine learning seeks to understand patterns from data that can be helpful to made future predictions of them. In this case, it considers that the past data is similar to future data. Practical problems in various branches of study, such as engineering, manufacturing, management, science, and medicine, have successfully applied machine learning. We note, for example, in engineering design, material behaviour, medical diagnosis, among other applications.

Furthermore, regression analysis, support vector machine, neural networks, *k*-means clustering and perceptron are some examples of well-known algorithms that fit well with these mentioned applications. Note that the existing machine learning techniques are not just to learn from the data. They also need to be able to adapt to data, i.e., to change with the data environment in the sense that it is learning via a context-aware approach to construct a system that is smart enough.

Additionally, looking for patterns in data to try to separate them into different groups by using, for example, similarity or dissimilarity rule is a common problem solved through the classification and clustering approaches. The first approach is a supervised method where the aim is to learn from labeled data such that it can predict for new data what group to which it belongs. The second approach is an unsupervised method that works with unlabeled data. It finds the best number of groups such that data points inside of a specific group (similarity) are more similar than data points outside of this group (dissimilarity). These two approaches arise in the context of optimization problems.

Machine learning is a growing field fundamentally based on artificial intelligence that has an enormous number of applications in connection with optimization methods. Although machine learning seeks to learn from data, Song et al. (2019) affirm that it is a successful method only through an algorithm that is specifically designed by experts when selecting parameters that are more appropriate for a specific goal. Then, to decrease human intervention, optimization techniques are being used, for example, by Carrizosa & Romero Morales (2013), Pedro Duarte Silva (2017), and Song et al. (2019) to tackle these configuration issues of the machine learning algorithms. The methods in the machine learning area aim to learn knowledge from data or experience. At the same time, the techniques from the optimization problems search for the best option or solution to a given problem.

The central aspect of this research is to study some ways to tackle large scale databases for which the structural patterns of data could be classified or clustered via the optimal solution of specific optimization problems. Chapter 1 provides the preliminary concepts and definitions to be used throughout the text. The preliminaries cover basic concepts of mathematical programming and present a specific goal programming model that could be applied for a binary classification problem. It indicates certain connections between machine learning and optimization problems and reports an important dimension reduction technique. Chapter 2 gives a brief overview of classification and clustering techniques. The overview includes the classification methods named as perceptron and support vector machine, and the popular clustering methods named k-means and hierarchical methods. The methodology used appears in Chapter 3. It describes a goal programming model to binary classification and an application of integer mathematical programming for a clustering problem with an emphasis on large-scale databases. Chapter 4 presents numerical experiments for classification and clustering in two case studies: the so-called database on breast cancer tumors, which either have a malignant or have a benign diagnosis, and a bovine animal database containing a set of features with no defined pattern respectively. The former is to perform a binary classification based on a goal programming model, and the latter is to search for the patterns in data to cluster them through a specific integer mathematical programming model using a sparse reduction method. Finally, Chapter 5 provides the conclusions obtained from this research with indications for further related works.

CHAPTER 1

Preliminaries

This chapter provides preliminary concepts and definitions used throughout the research. It covers the most basic concepts of optimization, mathematical programming, goal programming, and machine learning. Moreover, it indicates peculiar connections between machine learning and optimization problems and, finally, presents a way of reducing the dimension of data.

1.1 Optimization

Optimization is a large area of study within pure and applied mathematics that, in short, seeks to obtain the best possible solution of a particular mathematical problem. It deals with a specific objective to be minimized or maximized by considering a series of limitations and satisfying a set of constraints. Progressing studies in optimization took place during World War II when the British military faced diverse difficulties with allocating their resources (such as fighter airplanes, radars, and submarines) to perform several activities. In these times of challenges, a group of mathematics developed a methodology that achieved the best result of a linear programming problem, at the moment that emerges the concept of Operation Research (OR).

Rao (2009) explained (see Table 1.1) that the methodologies for OR split up into three groups: mathematical programming or optimization techniques, stochastic process techniques, and statistical methods. Note in the first column of Table 1.1 that the expressions optimization and mathemat-

Mathematical programming or optimization techniques	Stochastic process techniques	Statistical methods
Calculus methods	Statistical decision theory	Regression analysis
Calculus of variations	Markov processes	Cluster analysis, pattern recognition
Nonlinear programming	Queueing theory	Design of experiments
Geometric programming	Renewal theory	Discriminate analysis (factor analysis)
Quadratic programming	Simulation methods	
Linear programming	Reliability theory	
Dynamic programming		
Integer programming		
Stochastic programming		
Separable programming		
Multiobjective programming		
Multi-criteria decision analysis		
Network methods: CPM and PERT		
Game theory		

Table 1.1: Methods of operations research. Adapt from Rao (2009)

ical programming sound to have the same meaning. This dissertation deals with the concepts and applications of the mathematical programming methodology for binary classification and clustering analysis, as detailed in Chapters 2 and 3.

1.2 Mathematical programming

Mathematical programming (MP) and modeling are the keys to the solution methods in OR. It can be applied in a variety of areas: business and industry (K. Brian Haley B.Sc. 1967), military (Fox & Burks 2019), public-sectors (Kose & Karabay 2016), among others. Likewise, some applications in engineering have been made as in production, civil, chemical, mechanical, or aerospace engineering. All those studies intend to achieve the best result of an objective that is limited by using a certain quantity of resources. In overall terms, according to Ozan (1986), MP is the mathematical representation to obtain the best distribution of scarce resources through programming.

The aim in MP is to find the optimal value of an objective function, satisfying a set of constraints that represents limited resources and explains the nature of the problem. For the decision-maker, MP is the most attractive approach for aiding to deal with quantifiable variables by searching relationships among them that are not readily perceivable. The optimization problems are structured depending on the existence of constraints (unconstrained and constrained optimization). Luenberger & Ye (2015) stated a constrained optimization problem as follows.

$$
\begin{aligned}
\text{Minimize} \quad & f(x) && (1.1)\\
\text{subject to} \quad & h_i(x) = 0, \quad i = 1, \dots, m, && (1.2)\\
& g_j(x) \leq 0, \quad j = 1, \dots, r, && (1.3)\\
& x \in \mathbb{R}^n, && (1.4)
\end{aligned}
$$

where $f(x)$ is the objective function, $x = (x_1, \dots, x_n)^T$ is an n-dimensional vector with unknown values, and f, h_i, and g_j are real-valued functions of the variable x, for $i = 1, \dots, m$, and $j = 1, \dots, r$. Constraints (1.2) and (1.3) are called of equality and inequalities constraints, respectively; and $S = \{x \in \mathbb{R}^n \mid h_i(x) = 0,\ g_j(x) \leq 0,\ i = 1, \dots, m,\ j = 1, \dots, r\}$ denotes the *set of feasible solutions*. Some common assumptions are used on this problem, as smooth and continuous functions and the n-dimensional space is to be a well-connected region.

The literature provides different paths of classifying mathematical programming problems. For example, Rao (2009) considers the following eight aspects. The existence of constraints (constrained and unconstrained problems). The nature of the design variables (static and dynamic variables). The physical structure (optimal control and nonoptimal control). The core of the equations (linear and nonlinear functions). The permissible values of the design variables (integer and continuous variables). The essence of the variables (deterministic and stochastic variables). The separability of the functions (separable and nonseparable functions). The number of objective functions (scalar and multiobjective problems).

Nevertheless, the approaches studied in this dissertation are based on linear programming, which takes place through the following standard mathematical formulation.

$$
\begin{aligned}
\text{Minimize} \quad & f(x) = \sum_{j=1}^{n} c_j x_j && (1.5)\\
\text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, \dots, m, && (1.6)\\
& x_j \geq 0, \quad j = 1, \dots, n, && (1.7)
\end{aligned}
$$

where the parameters a_{ij}, b_i and c_j are known, and even though this formulation only comes up with equality constraints, the variation for inequality case could be made. If $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$ or $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$ appears in the set constraints, then a slack variable $y_i \geq 0$ is removed or added from each $\sum_{j=1}^{n} a_{ij}x_j$, respectively.

1.3 Goal programming

The goal programming methodology is an excellent tool to deal with conflicting objectives (Jones & Tamiz 2010). The essence of goal programming (GP) is the minimization of unwanted deviation variables, where they need to work together in the form of an achievement function. The purpose is to minimize the achievement function and thus ensure that the solution found is "as close as possible" to the set of desired goals. Mathematical programming and multiple criteria decision making have provided the basis for enhancing the studies of goal programming that started from the works of Charnes et al. (1955), Lee (1972), and Ignizio (1976, 1982, 1985).

Charnes et al. (1955) applied GP for an executive compensation problem. Then, GP was seen as a derivative of linear programming model until 1961 when Charnes & Cooper (1957) made first formal statements, making it one of the most popular techniques in the field of multi-criteria decision making (MCDM) in those years. However, the GP approach was not complete until 1980, when the following list of considerations was suggested to avoid basic errors:

- Pareto-inefficient solutions must be included;
- Redundancy obtained when a high number of priority levels are used;
- Apply a weight sensitivity analysis;
- Direct comparison of incommensurable goals;
- The preferences of the decision-maker(s) have a bad representation.

A more definitive and structure literature in this field was developed by Romero (1991), Tamiz & Jones (1997), and Ignizio (2004).

In short, a goal programming model has six principal elements: decision-maker, decision variables, criterion, objective, goal, and deviation variables. Jones & Tamiz (2010) explained that a decision-maker is a person, organization, or stakeholder whose decision or objective of the problem belongs to and is described by decision variables to know how a decision is going to be taken. Also, the criterion defined by the objective measures the goodness of any solution to a decision problem that has a different scale direction (minimize or maximize). Finally, a goal is a numeric level (also called target level) for which the decision-maker wants to get, and the deviation variables control the difference (positive or negative) between this level and the achieved optimal solution.

A general format for a GP model is stated as follows.

$$\begin{aligned} \text{Minimize} \quad & z = h(n,p) && (1.8) \\ \text{subject to} \quad & f_q(x) + n_q - p_q = b_q, \quad q = 1,\dots,Q, && (1.9) \\ & x \in F, && (1.10) \\ & n_q, p_q \geq 0, \quad q = 1,\dots,Q, && (1.11) \end{aligned}$$

where (1.9) expresses the quantity of goals ($q = 1,\dots,Q$) the decision maker wants to achieve; there are n decision variables $x = (x_1,\dots,x_n)^T$; $f_q(x)$ is the value at the solution x to be achieved for each goal; n_q and p_q are negative and positive deviation variables for each goal, respectively; $n = (n_1,\dots,n_Q)^T$ and $p = (p_1,\dots,p_Q)^T$; b_q is a numeric level that limits each goal; F represents the set of hard linear constrains in (1.10); sign restrictions for deviation variables is stated in (1.11); and the objective function $z = h(n,p)$ to be minimized in (1.8) can be a linear or nonlinear function for the deviation variables n and p. To ensure that the solution is the closest to get the desired goal values, Jones & Tamiz (2010) describe three variants of model: lexicographic goal programming (LGP), weighted goal programming (WGP), and Tchebychev goal programming (THGP).

Priority levels characterize the LGP, where each level contains an achievement function $h_l(n,p)$ on the deviation variables to be minimized according to a predetermined ordering (Silva & Marins 2015): first h_1; second h_2; and so on, up to the last h_L. Lee (1972) presented a mathematical formulation for LGP, for which the objective function (1.8) is replaced with Lex Min z, where $z = [h_1(n,p), h_2(n,p), \dots, h_L(n,p)]$, and $h_l(n,p)$ has the priority level $l = 1,\dots,L$ to be minimized as in the following standard structure.

$$\begin{aligned} \text{Lex Minimize} \quad & z = [h_1(n,p), h_2(n,p), \dots, h_L(n,p)] && (1.12) \\ \text{subject to} \quad & f_q(x) + n_q - p_q = b_q, \quad q = 1,\dots,Q, && (1.13) \\ & x \in F, && (1.14) \\ & n_q, p_q \geq 0, \quad q = 1,\dots,Q. && (1.15) \end{aligned}$$

If each $h_l(n,p)$ is linear and separable then it can be state as $h_l(n,p) = \sum_{q=1}^{Q} \left(\frac{u_q^l n_q}{k_q} + \frac{v_q^l p_q}{k_q} \right)$, where the preferential weights u_q^l and v_q^l are associated to the variables n_p and p_q, respectively, in the l-th priority level. The weight models the preference corresponding to its deviation variable, and the parameter k_q is a normalizing factor for each deviation variable, which for Jones & Tamiz (2010) is to standardize the contribution that each deviation variable gives to the objective function. Three types of normalization could be applied: percentage, zero-one, and Euclidean normalization. Some applications for the LGP variant have been studied. McGregor & Dent (1993) used the LGP model to evaluate the trade-offs between economic, environmental, and energy factors by developing forest energy plantations in Eastern Ontario, Canada. And Nha et al. (2013) utilized a version of LGP, lexicographic dynamic goal programming (LDGP), for implementing time series in a pharmaceutical case of the study, where the optimal solution provides the optimal drug configuration.

Note that sometimes the goals are measured by different units. The WGP variant allows evaluating the compromise among the deviation variables by using normalized weights in the achievement function. The decision-maker informs the importance of each goal to obtain a related optimal solution (Tamiz et al. 1995). Jones & Tamiz (2010) presented a mathematical formulation for WGP that is used when there is a direct comparison among the relative importance of goals, for which $h(n,p) = \sum_{q=1}^{Q} \left(\frac{u_q n_q}{k_q} + \frac{v_q p_q}{k_q} \right)$, u_q and v_q are the assigned weights for the negative and positive deviation from the target value b_q, respectively, and the parameter k_q is a normalizing factor for each deviation variable. Note that the WGP variant allows an equilibrium between all unwanted deviation variables. Assuming linearity of the achievement function, WGP can be represented from the following formulation.

$$\text{Minimize} \quad \sum_{q=1}^{Q} \left(\frac{u_q n_q}{k_q} + \frac{v_q p_q}{k_q} \right) \tag{1.16}$$

$$\text{subject to} \quad f_q(x) + n_q - p_q = b_q, \quad q = 1, \ldots, Q, \tag{1.17}$$

$$\underline{x} \in F, \tag{1.18}$$

$$n_q, p_q \geq 0, \quad q = 1, \ldots, Q. \tag{1.19}$$

Recent applications for the WGP variant appear in the literature. Zografidou et al. (2016) used the WGP variant to found the optimal design of a Greek renewable energy production network considering social, environmental, and economic criteria. Moreover, Jayaraman et al. (2017) developed a mathematical formulation to determine the optimal allocation across various industrial sectors by taking into consideration four goals: economic development, electricity consumption, greenhouse emissions, and the total number of employees.

Flavell et al. (1976) introduced the THGP variant that uses the Tchebychev metric (L_∞) to measure distances, i.e., the maximal normalized weighted deviation from amongst the set of unwanted deviation variables is minimized. For this reason, it is sometimes referred to as goal programming Min-Max, and the decision-maker is seeking the balance of the achievement function. Unlike the LGP model, instead of establishing a priority preference level, it looks for a balance between all the desire goals values. If $\lambda \geq 0$ is the maximal normalized weighted deviation from the targets, THGP can be represented by using $z = \lambda$ with the addition of the following constraints $\frac{u_q n_q}{k_q} + \frac{v_q p_q}{k_q} \leq \lambda$, $q = 1, \ldots, Q$, to the previous model, according to the following formulation.

$$
\begin{aligned}
\text{Minimize} \quad & z = \lambda && (1.20)\\
\text{subject to} \quad & \left(\frac{u_q n_q}{k_q} + \frac{v_q p_q}{k_q}\right) \leq \lambda, \quad q = 1, \ldots, Q, && (1.21)\\
& f_q(x) + n_q - p_q = b_q, \quad q = 1, \ldots, Q, && (1.22)\\
& x \in F, && (1.23)\\
& n_q, p_q \geq 0, \quad q = 1, \ldots, Q. && (1.24)
\end{aligned}
$$

The literature presents practical applications for the THGP variant. For example, Ignizio (2004) used a linear zero-one THGP model to obtain the optimal allocation of maintenance technicians in a factory to optimize the average factory cycle time. And Ghufran et al. (2015) described a THGP variant for the stratified double sampling problem, for which it finds the approximate optimal solution when strata weights are unknown, and non-response is present.

Additionally, in order to encompass the variants LGP, WGP, and THGP into a unique model, Romero (2001) proposed the extended goal programming (EGP) that facilitates to address many ways to minimize the unwanted deviation variables. EGP merges the previous models and provides a useful structure for including different metrics in the achievement function. It can be used to model different problems due to its ability to combine the various underlying philosophies of satisfying (from LGP), optimizing (from WGP), and balancing (from THGP) in a multiobjective environment (Jones & Tamiz 2010). Using a different notation from Romero (2001) and Jones & Tamiz (2010), if $\lambda_\ell \geq 0$ is the maximal normalized weighted deviation from the targets, EGP has the objective function Lex Min $z = [h_1(n,p), h_2(n,p), \dots, h_L(n,p)]$, where $h_\ell(n,p) = \alpha_\ell \lambda_\ell + (1-\alpha_\ell)\{\sum_{i=1}^{Q}(\frac{u_q^\ell n_q}{k_q} + \frac{v_q^\ell p_q}{k_q})\}$, $h_\ell(n,p)$ has the priority level $\ell = 1,\dots,L$ to be minimized, and the constraints $\frac{u_q^\ell n_q}{k_q} + \frac{v_q^\ell p_q}{k_q} \leq \lambda_\ell$, $\ell = 1,\dots,L$, $i = 1,\dots,Q$, are added to the model (1.12)–(1.15). EGP has been improved to address the most diverse practical applications. García et al. (2010) compared WGP, THGP, and EGP models in the ranking of companies. In the forestry sector, Giménez et al. (2013) presented a model for achieving the consensus decision on a forest management problem. In people management, De Andres et al. (2010) described a EGP model for evaluating performance. There are other variants for goal programming. For example, Uría et al. (2002) proposed the meta goal programming, Chang (2008) presented the multiple-choice goal programming, and Tiwari et al. (1987) studied the fuzzy goal programming. The latter deals with different levels of uncertainty.

1.4 Machine learning

Machine Learning (ML) is a multidisciplinary field that draws on the results from artificial intelligence, probability and statistics, computational complexity theory, control theory, philosophy, among others (Mitchell 1997). In the literature, ML may present different definitions according to the diversity of application fields. For this research, we explore an overview of optimization models with ML by using experiences or example data whose patterns are taken as an input to learn from and optimize. Hence, it leads to predictions if the model is predictive, extract knowledge if it is a descriptive one, or it does both.

There are different areas where machine learning is applied. In finance, to predict the credit risk of clients in a bank (Baesens et al. 2003), fraud detection or to investing in the stock market (Gavrishchaka & Banerjee 2006). In medicine, ML is used for medical diagnosis, and in science to manage big databases in physics, astronomy, and biology. According to Simeone (2017), machine learning approaches can be classified into three groups: supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning works with previously known information (the input data). It seeks an association rule that learns the relationship among the input data, whose output interpretation is already known, to predict the expected answer (the output data) of new data. The classification algorithms and regression techniques typically produce this associative learning (Shobha & Rangaswamy 2018).

Classification algorithms predict discrete responses with input data that can be categorized, tagged, or separated into groups or classes. For example, when mails are directed to a specific mailbox, a classification algorithm is applied to distinguish if an email is genuine or spam (Shafigh & Sourati 2016). Some well-known applications appear for bank credit risk (Pandey et al. 2017), speech recognition (Koolagudi et al. 2018), and medical imaging (Giger 2018). Alternatively, regression techniques predict continuous responses by achieving an association rule between two or more variables, for which a linear equation should fit the observed data. For example, a supermarket collected data from costumers' most preferred dates to buy groceries, and a regression finds a relationship between the two following variables: how much a customer buys and when it does it (Larivière & Van den Poel 2005).

Unsupervised learning operates with input data whose labels or answers are unknown, and sometimes they are also used for preliminary data exploration. In this case, the aim is to find some patterns on the input data, see how often they occur, and learn what generally happens or not. Moreover, this type of learning finds the structure of data and the relationship between their variables, often with the help of visual analytic tools.

The k-means clustering, hierarchical clustering, and principal component analysis (PCA) are the most popular unsupervised algorithms. The first two seek to group similar samples according to which variables the data are molded. PCA allows making a dimension reduction by combining the variables to the most representative ones. Several applications have been studied via unsupervised learning. Kakushadze & Yu (2017) used a k-means clustering algorithm to group types of cancers. Qureshi & Ahamad (2018) proposed a clustering method using k-means based on image segmentation with neutrosophic logic. Liu & Ge (2018) applied a hierarchical clustering based on randomly weighting forests to classify complex industrial processes. And Wei et al. (2019) studied several hierarchical divisive clustering for categorical data.

In reinforcement learning, an agent learns by interacting with the environment which generates a certain state. Then, the result of this action is a reward that defines if we are closer to the goal or if the agent should take another action (to interact again with the environment) in order to maximize the total reward (Alpaydin 2010). Typical practical applications of reinforcement learning including video games (Sethy et al. 2015), robotics (Martínez-Tenor et al. 2018), and traffic-signal control (Aragon-Gómez & Clempner 2020).

1.5 Connection between optimization and machine learning

Machine learning (ML) and optimization (OP) techniques belong to different areas, and according to Song et al. (2019), they both are based fundamentally on artificial intelligence. Moreover, they also interact with each other and with themselves to overcome their limitations by working together. Figure 1.1 shows the two types of cooperation between ML and OP. Machine learning for optimization (Interaction 1) and optimization for machine learning (Interaction 2).

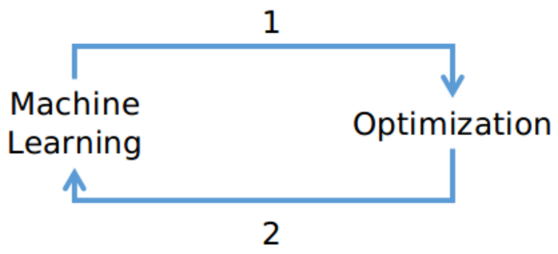


Figure 1.1: Interactions between machine learning and optimization. Adapt from Song et al. (2019)

Machine Learning for Optimization, as illustrated by Interaction 1, involves procedures that incorporate machine learning techniques into optimization, extracting patterns in data and transforming them into information to set parameters and components of the optimization algorithm. Machine learning techniques increase the process and performance of optimization algorithms. They help the speed-up of search processes and improve the quality of solutions. According to Song et al. (2019), there are three types of procedures that machine learning can improve optimization. It improves metaheuristic, algorithm selection, and enhance hyper-heuristics.

Optimization for Machine Learning (Interaction 2) involves approaches that incorporate optimization techniques into machine learning to overcome some design errors in machine learning algorithms since it must need some expert intervention to choose and set parameter values. Optimization improves the machine learning algorithm by decreasing human participation, and it can be incorporated into any process step of a machine learning algorithm. Song et al. (2019) highlighted the following procedures: data preprocessing, algorithm selection, hyper-parameter tuning, and model training.

Before applying any machine learning algorithm, *data preprocessing* transforms the raw data into just the best representative of them, considering data cleaning, dimensionality reduction, and instance reduction. *Algorithm selection* involves choosing the best machine learning algorithm that best fits the problem.

Gambella et al. (2020) also writes about the interaction between machine learning and optimization into three types: machine learning applied to management science problems, machine learning to solve optimization problems, and machine learning problems formulated as optimization problems. The first interaction incorporates machine learning data predictions into management science for making optimal decision making (Kraus et al. 2020); the second interaction uses machine learning techniques to solve, for example, hard optimization problems or to complement existing approaches of combinatorial optimization problems (Bottou et al. 2016). Finally, the third interaction is when machine learning problems are defined as optimization problems and which objectives are, for example, optimizing the training error, the measure of fit, or cross-entropy.

Thereby, Interaction 1 of Song et al. (2019) is similar to the second interaction from Gambella et al. (2020) called machine learning to solve optimization problems, and Interaction 2 (Song et al. 2019) has its alike with the third interaction of Gambella et al. (2020) named as machine learning problems formulated as optimization problems.

1.6 Dimensionality reduction

Dimensionality reduction techniques of data are another class of predictor transformations. These methods reduce the data by generating a smaller set of predictors that seek to capture a majority of the information in the original variables (Kuhn & Johnson 2013). Note that these methods are helpful when modeling a considerable number of variables. Therefore, a reasonable fidelity of the original data must be provided from fewer variables.

Reduction techniques enable data exploratory analyses by reducing the complexity of the dataset but approximately preserving essential properties, such as retaining the distances between cases or subjects. If they can reduce the complexity to a few dimensions, then it is possible to plot the data and explore its intrinsic properties (Dinov 2018). The most data reduction techniques have the new predictors as functions of the original predictors. Then, all the original predictors are still needed to create representative variables. This class of methods is often called signal extraction or feature extraction techniques.

Thus, several dimensionality reduction techniques have been proposed to handle large databases derived from well-known classical methods. Linear discriminant analysis (LDA) and principal component analysis (PCA) are classical linear techniques to reduce the data. The former works with the feature selection from a d-dimension database and selects a k-dimension feature ($k < d$) that gives the most information. The latter finds a new set of attributes of k-dimension generated by combining the original features from a d-dimension database (Alpaydin 2010). For example, canonical correlation analysis (CCA) and independent component analysis (ICA) are linear techniques derived from PCA. Moreover, nonlinear dimensionality reduction methods have also been developed, for example, Kernel PCA (Alpaydin 2010). The next section presents more details about the recognized PCA method.

1.6.1 Principal component analysis method

PCA method is an unsupervised method used in a big database to find a lower dimension by reducing the number of features while keeping as much information as possible. A new set of features is created as a combination of the original ones without require information about the classes. It is commonly used for data visualization, anomaly detection, lossy data compression, and feature extraction.

An algorithm for PCA method is usually defined as the projection of the data onto a suitable lower-dimensional feature subspace (Watt et al. 2016), while it maximizes the variance of the projected data. Another interpretation is that the PCA method seeks a projection that minimizes the square error between the projected data and the original data. Next, we describe these two interpretations of the PCA method.

Maximum projected data variance approach

Let $\{x^{(1)}, \ldots, x^{(m)}\}$ be a given set of data points, where $x^{(i)} \in \mathbb{R}^n$. An algorithm that maximizes the variance of the projected data points seeks typically to determine a new and reduced subspace of k-dimension ($k \ll m$) to project each $x^{(i)}$ onto it while maximizing its variation relative to the average value of points (Zaki & Meira Jr. 2014). In advance, the data points must be pre-processing to make each feature contributes similarly. Thus the data are standardized to have the median equal to zero.

First, a projection onto one-dimension space ($k = 1$) is done in the unitary direction u. As $||\mathsf{u}|| = 1$, the length of the projection of $x^{(i)}$ along u is $\mathsf{u}^T x^{(i)}$. PCA looks for a direction u that maximizes the variance of data points that can be seen as the following optimization problem. Considering centered data, so it has a mean μ equal to zero and, if it is not, subtract the mean of each feature from the data points to make $\mu = 0$.

$$\begin{aligned} &\text{Maximize} && \frac{1}{m}\sum_{i=1}^{m}\left(\mathsf{u}^T x^{(i)}\right)^2 && (1.25)\\ &\text{subject to} && ||\mathsf{u}|| = 1. && (1.26)\end{aligned}$$

The expression in the objective function (1.25) gives the following relation for the variance:

$$\sigma_{\mathsf{u}}^2 = \frac{1}{m}\sum_{i=1}^{m}\left(\mathsf{u}^T x^{(i)}\right)^T\left(\mathsf{u}^T x^{(i)}\right) = \mathsf{u}^T\left[\frac{1}{m}\sum_{i=1}^{m} x^{(i)} x^{(i)T}\right]\mathsf{u}.$$

Thus, $\Sigma \equiv \frac{1}{m}\sum_{i=1}^{m} x^{(i)} x^{(i)T}$ is the covariance matrix of the data, and the previous expression is reformulated as follows.

$$\sigma_{\mathsf{u}}^2 = \mathsf{u}^T \Sigma\, \mathsf{u}. \tag{1.27}$$

Using the expression in (1.27) the optimization problem is redefined as follows.

$$\begin{aligned} &\text{Maximize} && \mathsf{u}^T \Sigma\, \mathsf{u} && (1.28)\\ &\text{subject to} && \mathsf{u}^T \mathsf{u} = 1. && (1.29)\end{aligned}$$

The previous optimization problem can be solved via the Lagrangian multiplier approach, including the multiplier ρ in the constraint and replacing the objective function to obtain the following unconstrained maximization problem.

$$\underset{\mathsf{u}}{\text{Maximize}} \quad J(\mathsf{u}) = \mathsf{u}^T \Sigma\, \mathsf{u} - \rho\left(\mathsf{u}^T \mathsf{u} - 1\right) \tag{1.30}$$

Setting the derivative of $J(\mathsf{u})$ from (1.30) with respect to u equal to zero, we obtain

$$\Sigma \mathsf{u} = \rho \mathsf{u}. \tag{1.31}$$

Note that ρ is an eigenvalue of the covariance matrix Σ for the associated eigenvector u. Furthermore, taking the dot product with u on both sides of (1.31) yields $\mathsf{u}^T\Sigma\,\mathsf{u} = \mathsf{u}^T\rho\mathsf{u}$. From (1.27) and (1.29), we then have $\sigma_{\mathsf{u}}^2 = \rho$. Therefore, to maximize the projected variance σ_{u}^2, a direction $\mathsf{u} = \mathsf{u}_1$ (first principal component) must be chosen that is equal to the eigenvector with the largest eigenvalue $\rho = \lambda_1$ of the covariance matrix Σ.

For the second principal component, it must be determined another direction ν that also maximizes the projected variance of data points and has magnitude $\|\nu\|^2 = \nu^T\nu = 1$ but is orthogonal to u_1. For the direction ν, we have the following optimization problem.

$$\begin{aligned} &\text{Maximize} && \nu^T\Sigma\nu \\ &\text{subject to} && \nu^T\nu = 1, \\ &&& \nu^T\mathsf{u}_1 = 0. \end{aligned}$$

Lagrange multipliers are also applied to obtain the unconstrained maximization problem

$$\underset{\nu}{\text{Maximize}} \quad J(\nu) = \nu^T\Sigma\nu - \rho(\nu^T\nu - 1) - \phi(\nu^T\mathsf{u}_1 - 0). \tag{1.32}$$

Taking the derivative of $J(\nu)$ from (1.32) with respect to ν, and setting it equal to zero, gives $2\Sigma\nu - 2\rho\nu - \phi\mathsf{u}_1 = 0$. The operation $(2\Sigma\nu - 2\rho\nu - \phi\mathsf{u}_1)^T\mathsf{u}_1 = 0$ implies to $2\nu^T\Sigma\mathsf{u}_1 - \phi = 0$. As $\Sigma\mathsf{u}_1 = \lambda_1\mathsf{u}_1$, then $\phi = 2\lambda_1\nu^T\mathsf{u}_1 = 0$. Thus, we have $2\Sigma\nu - 2\rho\nu = 0$, and $\Sigma\nu = \rho\nu$.

This result means that ν is another eigenvector of Σ associated with the eigenvalue ρ. Also, to maximize the projected variance, a second principal component $\nu = \mathsf{u}_2$ must be found, which is also an eigenvector of Σ but associated to the second largest eigenvalue $\rho = \lambda_2$.

The illustration of the two principal components u and ν appears in Figure 1.2 as a two-dimensional subspace spanned by the orthonormal vectors u and ν.

In summary, the principal components of a m-dimension database provide k basis vectors for a k-dimension subspace. The k vectors also are the ones that maximize the variance of the projected data, and they are formed by the eigenvectors $\mathsf{u}_1, \mathsf{u}_2, \ldots, \mathsf{u}_k$ of the covariance matrix Σ (positive semi-definite) that have the respective eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_k$. The eigenvalues must all be non-negative, and we can thus sort them in decreasing order as follows.

$$\lambda_1 \geq \lambda_2 \geq \cdots \lambda_k \geq \lambda_{k+1} \cdots \geq \lambda_m \geq 0 \tag{1.33}$$

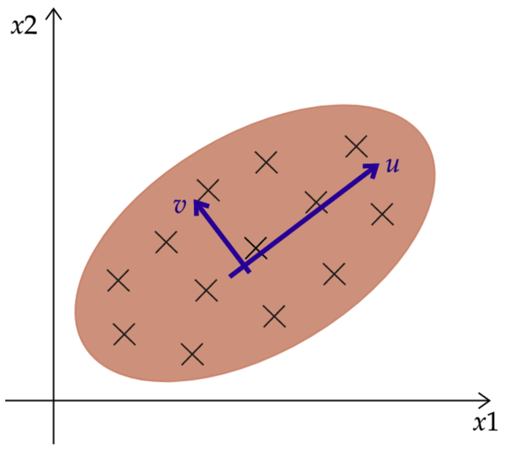


Figure 1.2: Principal components for two-dimensional subspace

We then select the k largest eigenvalues and their corresponding eigenvectors to form the best k-dimensional approximation. In order to find how many dimensions will be useful for a reasonable estimate, the following criterion explained in Zaki & Meira Jr. (2014) for choosing k defines a function $f(k)$ as the fraction of the total variance captured by the first k principal components from the original m-dimension dataset.

$$f(k) = \frac{\lambda_1 + \lambda_2 + \ldots + \lambda_k}{\lambda_1 + \lambda_2 + \ldots + \lambda_m} = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{m} \lambda_i} \tag{1.34}$$

Given a certain desired variance threshold θ and starting from the first principal component, the function $f(k)$ iterates on adding additional components, and stops at the smallest value k, for which $f(k) \geq \theta$, is reached. Note that we select the fewest number of dimensions such that the subspace spanned by these k vectors captures at least θ fraction of the total variance. For standard practices, θ is usually set to 0.9 or higher.

Minimum squared error approach

An alternative formulation of PCA is based on projection error minimization. The minimum squared error approach (MSE) looks for an orthogonal projection that minimizes the overall projection error. Using the notation of Bishop (2006), let $\{x_n\}$ be a dataset of observations where $n = 1, \ldots, N$, and x_n is a Euclidean variable with dimensionality D, and consider a complete orthonormal set of D-dimensional basis vectors $\{u_i\}$ where $i = 1, \ldots, D$ that satisfy $u_i^T u_j = \delta_{ij}$. Thus, each data point can be represented exactly by a linear combination of the basis vectors.

$$x_n = \sum_{i=1}^{D} \rho_{ni} u_i, \tag{1.35}$$

where the coefficients ρ_{ni} will be different for different data points. For each data point x_n, this simply corresponds to a rotation of the coordinate system to a new system defined by the $\{u_i\}$, and the original D components $\{x_{n1}, \ldots, x_{nD}\}$ are replaced by an equivalent set $\{\rho_{n1}, \ldots, \rho_{nD}\}$. Considering the orthonormality property of set $\{u_i\}$, we obtain $\rho_{nj} = x_n^T u_j$, and each x_n can be express as

$$x_n = \sum_{i=1}^{D} (x_n^T u_i) u_i \tag{1.36}$$

However, the objective here is to approximate this data point using a representation involving a restricted number $M \ll D$ of variables corresponding to a projection onto a lower-dimensional subspace. The M-dimensional linear subspace can be represented by the first M of the basis vectors $\{u_1, u_2, \ldots, u_M\}$, and we approximate each data point x_n by

$$\tilde{x}_n = \sum_{i=1}^{M} z_{ni} u_i + \sum_{i=M+1}^{D} b_i u_i, \tag{1.37}$$

where the set $\{z_{ni}\}$ depends on the particular data point, wheres the set $\{b_i\}$ is constant that is the same for all data points. The main goal is to minimize the squared distance between each original data point x_n and its approximation $\tilde{x}_n$, averaged over the data set. Thus, the following function J is also defined as distortion error must be minimized.

$$J = \frac{1}{N} \sum_{n=1}^{N} ||x_n - \tilde{x}_n||^2. \tag{1.38}$$

If (1.37) is replaced into (1.38), it is easy to find z_{ni} from the partial derivate of J with respect to z_{ni} and equal to zero, and making use of orthonormality conditions, we obtain $z_{nj} = x_n^T u_j$, $j = 1, \ldots, M$. Similarly, setting the derivative of J with respect to b_i to zero, which it leads to $b_j = \bar{x}^T u_j$, $j = M+1, \ldots, D$. Now, if we substitute for z_{ni} and b_i, and make use of the expansion (1.35), the approximation error only plays a role in dimension $M+1,\ldots,D$ as follows.

$$x_n - \tilde{x}_n = \sum_{i=M+1}^{D} ((x_n - \bar{x}^T)u_i)u_i \tag{1.39}$$

The displacement vector from x_n to $\tilde{x}_n$ lies in the space orthogonal to the principal subspace, which is a linear combination of $\{u_i\}$ for $i = M+1,\ldots,D$. It can be said that the minimum error (1.38) is given by the orthogonal projection of x_n onto the principal subspace spanned by $\{u_i\}$. Therefore, we obtain an expression for the distortion measure J as a function purely of the $\{u_i\}$ in the form

$$J = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=M+1}^{D} (x_n^T u_i - \bar{x}^T u_i)^2 = \sum_{i=M+1}^{D} u_i^T \Sigma u_i. \tag{1.40}$$

There remains the task of minimizing J with respect to the $\{u_i\}$, which must be a constrained minimization to avoid vacuous $u_i = 0$. The constraints arise from the orthonormality conditions, and the solution is expressed concerning the eigenvector expansion of the covariance matrix. For the case of a two-dimensional data space $D = 2$ and a one-dimensional principal subspace $M = 1$, to choose a basis vector u_1 the following formulation is made

$$\begin{aligned} &\text{Minimize} \quad && J = u_1^T \Sigma u_1 \\ &\text{subject to} \quad && u_1^T u_1 = 1. \end{aligned}$$

Figure 1.3 illustrates a representation in a two-dimensional data space that reduces to one-dimension principal subspace.

The latter model can be reformulated by using the following Lagrangian multiplier approach

$$\text{Minimize} \quad J = u_1^T \Sigma u_1 + \lambda(1 - u_1^T u_1), \tag{1.41}$$

which is solved by making the partial derivate with respect to u_1 equal to zero, to obtain $\Sigma u_1 = \lambda u_1$. The general solution to the minimization of J for arbitrary D and arbitrary $M \ll D$ is obtained by choosing the $\{u_i\}$ to be eigenvectors of the covariance matrix given by

$$\Sigma u_i = \lambda u_i \tag{1.42}$$

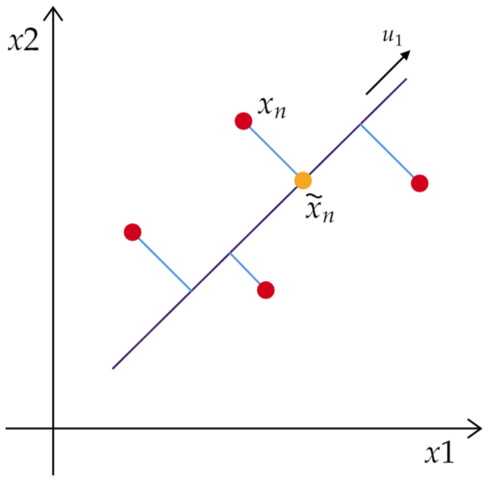


Figure 1.3: PCA for two-dimension data space and MSE approach. (Bishop 2006)

where $i,\dots,D$, and as usual the eigenvectors $\{u_i\}$ are chosen to be orthonormal. The corresponding value of the distortion measure is then given in terms of the eigenvalues of the data covariance matrix Σ as follows.

$$J = \sum_{i=M+1}^{D} \lambda_i. \tag{1.43}$$

This last equation represents the sum of the eigenvalues of those eigenvectors that are orthogonal to the principal subspace (Bishop 2006). Therefore, the minimization of J requires to choose the M eigenvectors as the principle subspace that are associated with the M largest eigenvalues.

In order to calculate the size of M, we use the following ratio between the fraction of the mean square size of error and the mean square size of data whose the output is the smaller possible value

$$\frac{\text{mean square size of error}}{\text{mean square size of data}} = \frac{\sum_{i=M+1}^{D} \lambda_i}{\sum_{i=1}^{D} \lambda_i}. \tag{1.44}$$

CHAPTER 2

Summary of classification and clustering

This chapter gives a brief overview of some classification and clustering approaches, where we explain how a linear classifier works, and describe the perceptron algorithm and support vector machine). Moreover, to outline the clustering technique, we present a description of two well-known methods, the *k*-means and hierarchical clustering.

2.1 Classification

The classification aims to assign an unknown pattern to one of many classes that are considered to be known (Theodoridis 2015). For example, banks predict the risk associated with a loan that relates to the probability of a customer not pay their credit (Pandey et al. 2017). Using the bank's record past data, and after a previous selection of features related to the costumer's attributes, an association between them and their associated class of risk will be found. This connection allows preventing when a future customer applies, so the bank will know if they are suitable or not to receive a loan.

Other applications have been made with classification systems: face recognition (Shailaja & Anuradha 2016), to classify if an image contains or not a face; in medicine, specifically to detect if a tumor could be benign or malign (Wolberg & Mangasarian 1989); and in recognition of letter to identify the authorship of a given text (Frey & Slate 1991). Usually, the classifier's output is often a discrete value; however, it is also common to see continuous variables as to made probability prediction.

As a supervised learning method, a classifier has two consecutive steps: firstly, a training step builds the model from a prediction function; secondly, a classification step sets the model according to its accuracy rate. In advance of the procedure steps, a data treatment process should be done by selecting the best combination of features from an enormous collection. A statistic analysis does this process to choose the most representative features for each class. If the amount of features is large, a dimension reduction method can be applied to reduce features by picking a subset of the most representative ones or by creating a set with new features that represent the original data but with a reduced dimension.

The training step attempts to "train" the model by using a training set composed of a vector $x = (x_1, x_2, \dots, x_n)^T$ as an input data, whose output (or class label) is already known. For example, take a set of pair $\{(x_i, y_i)\}_{i=1}^{n}$ where y_i is the output variable denoting the class related to the input data x_i. According to Theodoridis (2015), the vector y declares the class labels whose entries belong to the discrete set $\{1, \dots, M\}$, when one has a M-class classification task. $M = 2$ indicates a binary classification (illustrated in Figure 2.1 for a linear case), and $M > 2$ states a multiclass classification. To obtain y for each x, a function f is determined where $y = f(x)$, which can be represented by a classification rule, a decision tree, or a mathematical equation. The function f aims to learn how to separate the data classes.

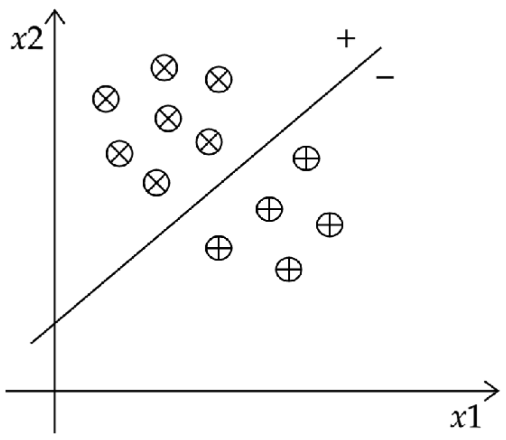


Figure 2.1: Classification model with two output labels. Adapt from Alpaydin (2010)

The classification step operates from a new set of data named test set. The function f is used to predict the class label of a new vector x, whose output class label is known, and with the obtained result, it can measure the classifier accuracy rate. This rate represents the percentage of output class labels that the model classifies correctly. If it is not considered acceptable, the model must be adjusted by returning to the training step, and then the model is tested again. When the model has a fair accuracy rate, it is ready to be used in a new set of data whose output does not know.

Watt et al. (2016) highlighted some practical applications in this field. The detection of faces from images for organizational purposes (object detection). Sentimental analysis that learns and identifies customer's feelings, either positive or negative. And the classification as a diagnostic tool in medicine, which is growing in detecting cancer diagnoses by taking DNA characteristics as features.

2.1.1 Perceptron algorithm

An example of a linear classifier is the perceptron algorithm proposed by Rosenblatt (1961), which seeks to find a hyperplane that separates two types of classes. This algorithm works with linearly separable databases because it will work or iterate until it finds a perfect solution, i.e., a hyperplane that divides both classes. Let x be a n-dimensional input vector; the perceptron algorithm looks for a linear combination that gives the output class for each instance, denoted by y and stated as

$$y = \sum_{i=1}^{n} w_i x_i + w_0, \tag{2.1}$$

where the weight w_i determines the contribution of each x_i to the perceptron output y, and w_0 is generally modeled as the weight coming from an extra *bias unit* ($x_0 = 1$). The equation (2.1) is also expressed as $y = w^T x + w_0$, where the output will take value $y = +1$ if the instances belong to one class on one side of the hyperplane, or take value $y = -1$ if the instances belong to the other class, i.e., lie to the other side of the hyperplane.

The learning process for a perceptron algorithm begins with the vector w formed of random weights that uses the previous equation to see if the instance is correctly classified. If it is not, the following expression modifies the vector w.

$$\Delta w_i = \eta d x_i, \tag{2.2}$$

where η is the learning rate that measures how fast the algorithm converges, d is the difference between the target output and the generated output, and x_i is the current instance being evaluated. Then, the new value of w is as follows.

$$w_i = w_i + \Delta w_i, \tag{2.3}$$

which only will be affected if the instance is not correctly classified. After all the instances go through this process, the algorithm iterates until no instance are miss classified, so the final vector w defines the correct hyperplane that divides both classes. It is important to note that the final result depends on the initial values of w and η.

One way of implementing the separation hyperplane is using perceptron as a neural network, specifically a single-layer network that works only for linear cases. However, multiple-layer perceptrons can be implemented for nonlinear cases with several applications in areas such as handwriting recognition, image recognition, pattern classification, among others.

2.1.2 Support vector machine

Support vector machine (SVM) is a discriminant-based method proposed by Cortes & Vapnik (1995) and linear classifier that aims to find an optimal hyperplane that separates the feature vectors in an n-dimensional space. Although several hyperplanes can separate the data, the SVM procedure looks for the optimal hyperplane, which does not only classify the training set correctly but also generalizes for unseen data.

Let $D \equiv \{(x_1,y_1),(x_2,y_2),(x_3,y_3),\ldots,(x_n,y_n)\}$ be a classification dataset containing n points $x_i \in \mathbb{R}^d$, where $\{x_i\}$ is a set of training observations with associated class labels y_i. The separate decision bound for two-class problems can be defined as follows.

$$h(x) = w^T x + b = 0, \tag{2.4}$$

where $w = (w_1,\ldots,w_d)^T$ is a weight vector in d-dimensional space, d is the number of features or attributes, and b is a scalar named the bias. For two-class problems, y_i could take one of two values in $\{-1,+1\}$, where $y_i = +1$ if $x_i \in C_1$ and $y_i = -1$ if $x_i \in C_2$.

$$\begin{aligned} w^T x_i + b &\geq +1 \quad \text{for} \quad y_i = +1 \\ w^T x_i + b &\leq -1 \quad \text{for} \quad y_i = -1 \end{aligned} \tag{2.5}$$

The two inequalities from (2.5) can be written as

$$y_i(w^T x_i + b) \geq +1, \quad \forall i. \tag{2.6}$$

As to the perceptron algorithm, the optimal solution (separate hyperplane) depends on the initial values of w and b. Therefore if there exist multiple solutions, the smallest generalization error helps to choose one. For SVM, to deal with this, the concept of margin is proposed as the shortest distance from the separating hyperplane to any of the samples. So for each point x_i, it can be said that the distance to the hyperplane $h(x)$ is

$$\delta_i = \frac{y_i h(x)}{||w||} = \frac{y_i(w^T x_i + b)}{||w||} \tag{2.7}$$

And the set of points with minimum distance solve the following problem.

$$\underset{x_i}{\text{Minimize}} \quad \frac{y_i(w^T x_i + b)}{||w||}. \tag{2.8}$$

The set of samples x^* that reaches the previous condition contains the support vectors, and for a

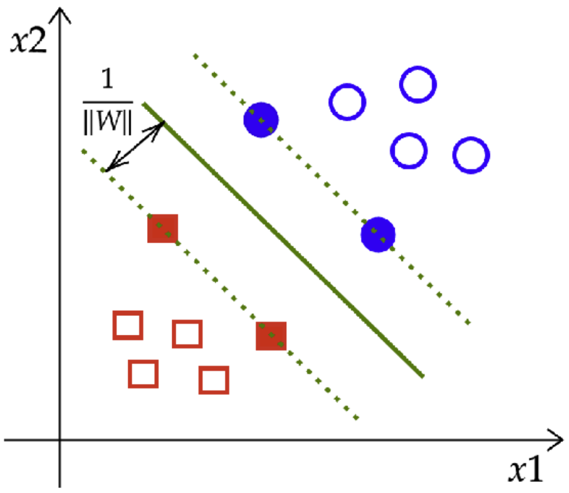


Figure 2.2: Support vector machine for two classes of data. Adapt from Zaki & Meira Jr. (2014)

better generalization, the optimal hyperplane is the one that maximizes the margin given by

$$\delta^* = \frac{y_i^*(w^T x_i^* + b)}{||w||}, \tag{2.9}$$

where y^* is the class label for x^*; the numerator $y^*(w^T x_i^* + b)$ is the absolute distance from support vector to the hyperplane; and the denominator $||w||$ is the relative distance in terms of w. Therefore, the distance in (2.9) also can be express as $\frac{1}{||w||}$ and to maximize the margin, it is formulate as follows.

$$\begin{aligned} &\text{Maximize} && \frac{1}{||w||} \\ &\text{subject to} && y_i(w^T x_i + b) \geq 1, \quad \forall x_i \in D. \end{aligned}$$

Figure 2.2 illustrates the distance between any point to the separator hyperplane is $\frac{1}{||w||}$ for both sides. However, instead of maximize the margin $\frac{1}{||w||}$, an equivalent reformulation can be done to minimize $||w||$. Moreover, for a mathematic convenient, the following is an equivalent minimization formulation.

$$\begin{aligned} &\underset{w,b}{\text{Maximize}} && \frac{||w||^2}{2} \\ &\text{subject to} && y_i(w^T x_i + b) \geq 1, \quad \forall x_i \in D. \end{aligned}$$

This quadratic problem could be solved by standard optimization algorithms, however, it is easier to solve if it does not depend on the dimension d, and instead on the number of samples n. According to Karush-Khun-Tucker conditions, the Lagrange multipliers α_i are applied to obtain the following model.

$$\text{Minimize } L = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(w^T x_i + b) - 1 \right). \tag{2.10}$$

To minimize L, it has to be done with respect w and b, and it also should be maximized with respect α_i. Then, the weight vector w is expressed as $w = \sum_{i=1}^{n} \alpha_i y_i x_i$, i.e., is a linear combination of the data points, with the Lagrange multipliers $\alpha_i y_i$, serving as coefficients (Zaki & Meira Jr. 2014). Also, it is known that $\sum_{i=1}^{n} \alpha_i y_i$ must be zero. With these expressions, a dual Lagrange optimization problem is built as follows.

$$\underset{\alpha}{\text{Maximize}} \quad L_{dual} = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \tag{2.11}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \tag{2.12}$$

$$\alpha_i \geq 0, \quad \forall x_i \in D. \tag{2.13}$$

Then, the dual Lagrange problem depends on n (the number of samples) and not on d (the dimension of inputs). Once the values of α_i are obtained for $i = 1, \ldots, n$, thus w and b also could be found. Vector w is obtained as a linear combination of the support vectors with its α_i when it is bigger than zero, and to compute b, it can be done as the average of $b_i = y_i - w^T x_i$ values for all the support vectors.

Finally, the SVM classifier works with the optimal hyperplane $h(x) = w^T x + b$, and given a new point called p, its class can be predicted as follows.

$$\hat{y} = sign\{w^T p + b\}, \tag{2.14}$$

where $sign\{.\}$ is a function that predicts the class of $\hat{y} \in \{-1, +1\}$, considering +1 belongs to the class group that is above the hyperplane and -1 belongs to the class group that is under the hyperplane.

The previous approach uses SVM to classify data that is linearly separable. However, this case is an ideal one, as most of the data is non-linear separable. An alternative approach is to map the original data into a higher dimensional space using a nonlinear mapping ϕ and subsequent that, search for a linear separating hyperplane in the new space (Han et al. 2011).

Then, the original database $D \equiv \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n)\}$ receives a non-linear transformation which generates a new database in a higher-dimensional space:
$D_\phi \equiv \{(\phi(x_1), y_1), (\phi(x_2), y_2), (\phi(x_3), y_3), \ldots, (\phi(x_n), y_n)\}$
After applying this transformation, the dual Lagrangian in (2.11) depends only on the dot product between two vectors in the new space:

$$\underset{\alpha}{\text{Maximize}} \quad L_{dual} = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \tag{2.15}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \tag{2.16}$$

$$\alpha_i \geq 0, \quad \forall x_i \in D. \tag{2.17}$$

To solve this, instead of computing the dot product in the transformed database, a mathematical equivalent is to apply a kernel function, $K(x_i, x_j)$, in the original database (Bishop 2006) define by:

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \tag{2.18}$$

This avoids the mapping in a new space and instead, all the calculations are made in the original space using the correct kernel function $K(x_i, x_j)$, for example the polynomial kernel, the Gaussian kernel, the sigmoid kernel, among others. However, as supervised learning knows the label of data, unsupervised learning or clustering seeks to classify the data without knowing the types of classes or how many.

2.2 Clustering

The clustering goal is to partition an unlabeled dataset into homogeneous groups or clusters, considering just the information obtained solely from the data. Because it works with no class to be predicted, clustering belongs to the unsupervised learning techniques with applications in data mining (Berkhin 2006), pattern classification (Sah et al. 2018), image segmentation (Dhanachandra & Chanu 2017), among others.

Each cluster is composed of observations that are similar between each of them (similarity) but different from the ones in other groups (dissimilarity). The criteria used differ in the knowledge of the data being studied (James et al. 2014). However, the similarity is usually expressed in terms of the sum of squares via Euclidean distance between the samples and its cluster centroid.

For a dataset D which is conformed of $\{x_1, \ldots, x_n\}$ points in a d-dimensional space ($D = \{x_i\}_{i=1}^{n}$) in a number of groups given by k clusters desired denoted as $C = \{C_1, \ldots, C_k\}$. Usually, to estimate the quality of a cluster and its representation for all points inside one, the mean or centroid

μ_i is calculated for each C_i. Denoted as $\mu_i = \frac{1}{n_i} \sum_{x_i \in C_i} x_j$, where the number of points in cluster C_i is denoted by $n_i = |C_i|$. Then, the similarity for each C_i (considering euclidean distance) is

$$\text{Similarity}(C_i) = \sum_{i=i}^{k} \sum_{x_j \in C_i} ||x_j - \mu_i||^2. \tag{2.19}$$

To find the best arrangement of partition the n points into k clusters, an optimization problem could be made that generates every possible partition and assign a score to each one so the best will be chosen. According to Zaki & Meira Jr. (2014), the following Stirling numbers of the second kind gives the exact number of how to made each partitioning.

$$S(n,k) = \frac{1}{k!} \sum_{t=0}^{k} (-1)^t \binom{k}{t} (k-t)^n. \tag{2.20}$$

However, to compute this technique, it turns exhaustive as it is not possible to reach all possible clusterings. To deal with this issue, different approaches have been proposed where the two most popular are k-means clustering and hierarchical clustering.

2.2.1 K-means clustering

In k-means clustering (Lloyd 1982), a data set is partitioned in k distinct non-overlapping clusters, which are specified in advance, where each observation is assigned to exactly one of the clusters. Moreover, the clustering seeks to find the best k clusters while minimizing the distance of each point to its cluster centroid.

Given a dataset $D = \{x_i\}_{i=1}^{n}$ and a k define number of clusters, its clustering is denoted as $C = \{C_1, \ldots, C_k\}$. The algorithm's goodness and quality is measured by the within-cluster variation that seeks to be small as possible (James et al. 2014). $W(C_i)$ is the defined score to measure how the observations differ one to another in their respective cluster.

$$\underset{C_1,\ldots,C_k}{\text{Minimize}} \sum_{i=1}^{k} W(C_i). \tag{2.21}$$

Thus, (2.21) minimize the total sum of the variations between observations in each cluster after the data get partitioned in k clusters. One approach to calculate the variation within-cluster, is the sum squared errors (SSE), defined as follows:

$$W(C_k) = SSE(C_k) = \sum_{x_j \in C_i} \|x_j - \mu_i\|^2. \tag{2.22}$$

It initializes by generate random k points as clusters centers. Then, each point $x_j \in D$ is assigned to the closest mean, which induces a clustering, with each cluster C_i comprising points that are closer to μ_i than other cluster mean. From equations (2.21) and (2.22), the optimization problem for k-means clustering is built as follows.

$$\underset{C_1,\dots,C_k}{\text{Minimize}} \quad \sum_{i=1}^{k} \sum_{x_j \in C_i} \|x_j - \mu_i\|^2. \tag{2.23}$$

Then, each cluster center C_i is update where new values of variation within-cluster (SSE) is computed iteratively until reaching a fixed point (where the cluster center do not change between iterations) or local minima.

However, to solve this problem, the algorithm finds all the k^n ways to partition n observations into k clusters but for an n and k bigger it is almost impossible. To made it easier, an alternative approach is proposed which founds a local optimum instead of the globally optimum clustering. Consider the binary variable y_{ji}:

$$y_{ji} = \begin{cases} 0, & \text{if data point } j \text{ belong to cluster } i; \\ 1, & \text{otherwise.} \end{cases} \tag{2.24}$$

The k-means clustering is formulated as a mixed interger nonlinear program (Gambella et al. 2020):

$$\text{Minimize} \quad \sum_{j=1}^{n} \sum_{i=1}^{k} y_{ji} \|x_j - \mu_i\|^2 \tag{2.25}$$

$$\text{subject to} \quad \sum_{i=1}^{k} y_{ji} = 1, \quad j = 1, \dots, n, \tag{2.26}$$

$$\sum_{j=1}^{n} y_{ji} \leq 1, \quad i = 1, \dots, k, \tag{2.27}$$

where the objective function (2.25) minimize the sum of the distance between all data points j and the center of their cluster i, considering the decision variable y_{ij}. The constraint (2.26) ensures that each point j is assigned only to one cluster i; and the constraint (2.27) ensures that each cluster i have at least one point. As the problem have a nonlinear and nonconvex objective function with discrete constraints, a linearized formulation by adding big-M constraints is obtained:

$$\text{Minimize} \quad \sum_{j=1}^{n}\sum_{i=1}^{k} d_{ji} \tag{2.28}$$

$$\text{subject to} \quad \sum_{i=1}^{k} y_{ji} = 1, \quad j = 1,\dots,n; \tag{2.29}$$

$$d_{ji} \geq \|x_j - \mu_i\| - M(1-y_{ji}) \quad j = 1,\dots,n, \quad i = 1,\dots,k; \tag{2.30}$$

$$\sum_{j=1}^{n} y_{ji} \leq 1, \quad i = 1,\dots,k; \tag{2.31}$$

$$d_{ji} \geq 0, \quad j = 1,\dots,n, \quad i = 1,\dots,k. \tag{2.32}$$

Here, the variable d_{ji} represent the distance (in this case, euclidean distance) between each point j and the center of their cluster i. The new constraint (2.30) state the upper bound of d_{ij} forms a reasonalble tight value for M. Here, $d_{ij} = 0$ only happens when i is not assigned to the cluster k, i.e. assuring that $d_{ij} \geq 0$.

Therefore, the clustering process can be summarize in two steps: i) after specify the number of k cluster desired, each observation is assign to a C_i cluster, and ii) update each centroids, reassign the observations and iterates until reached a fixed point or local minima.

Furthermore, the algorithms iterations may have two different ways to reach an optimal result or a stop criteria. The first, an optimal value is reached when the centroids from one iteration to another no longer changes; and the second, define an ε value, where $\sum \|\mu_i^{(t)} - \mu_i^{(t-1)}\|^2 \leq \varepsilon$ is the stopping criteria, for $\varepsilon \geq 0$ and t represent the current iteration.

It is important to remember that the change in result for k-means is because the initial values of centroids have a random nature (this is the reason to get a local rather than global optimum). Hence it is recommended that the algorithm should be run several times to select the best solution i.e., the one with lowest SSE value.

2.2.2 Hierarchical clustering

Despite k-means clustering, hierarchical clustering does not require to establish a fixed number k clusters and it can be shown in a tree-based representation called “dendogram”. This algorithm seeks to create groups between observations that are more similar to each other than others in different groups. The most popular similarity measure is Euclidean distance, which is calculate as

$$d(x^r, x^s) = \sum_{j=1}^{d} |x_j^r - x_j^s|. \tag{2.33}$$

This algorithm approached how the dendrogram built: agglomerative or divisive. The former makes it from the bottom to the top, where each observation is a single cluster that starts to fusing between the more similar ones until there is a single cluster. The latter, works in the opposite way, it begins with a single cluster which divided until each observation is its cluster. A basic dendrogram is shown in Figure 2.3.

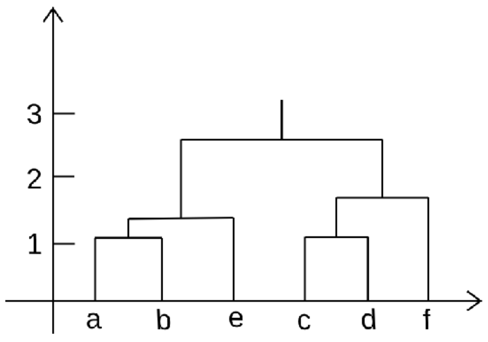


Figure 2.3: Dendrogram illustrating a hierarchical clustering. Adapt from Alpaydin (2010)

To construct the dendrogram, first it establishes the type of similarity between each pair of observations, for example, Euclidean distance is the most popular. Then, an iteration process begins by building the bottom of the dendrogram (if it is an agglomerative type) and each observation its is own cluster, so it begins with n clusters. After, the two more similar observations, fused in a new cluster so there will be $n-1$ clusters. For the next iteration, another two similar clusters fused, so now there are $n-2$ clusters; the following iterations continue until there exists one single cluster and the dendrogram is complete.

This similarity between clusters is based on the notion of linkage among observations. Three most common types are: single, complete and average. The single linkage used the smallest distance between all the combination of possible pair of observations or clusters.

$$d(G_i, G_j) = \underset{x^r \in G_i,\ x^s \in G_j}{\text{Minimize}}\ d(x^r, x^s) \tag{2.34}$$

The complete linkage is the opposite as the single linkage because it take the largest distance.

$$d(G_i, G_j) = \underset{x^r \in G_i,\ x^s \in G_j}{\text{Maximize}}\ d(x^r, x^s) \tag{2.35}$$

For the average linkage, the distance which is used takes the average between all the pairs.

Each dendogram could have 2^{n-1} possible re orderings, and to interpreted it in order to know how many clusters are the best, an horizontal cut in the dendogram is done control by the height of it in the vertical axis.

CHAPTER 3

Methodology

This chapter presents the methodology and the particular models used in this research to classify and cluster databases. It describes three variants of a classification model based on goal programming technique, and a clustering model based on integer mathematical programming for a defined/certain number of clusters.

3.1 Classification method based on goal programming

We use a binary classification model based on goal programming (GP) proposed by Jones et al. (2007), which is characterized by distance metrics and preference modeling techniques. Furthermore, it allows defining weights and parameters as well as the modeler has the option to vary the weight and parameter values, then the probability of obtaining correct classification in each level increases and is more accurate.

Let A and B be two groups with n_1 and n_2 observations, respectively. For each observation i, let $x_{ij}^{(a)}$ and $x_{ij}^{(b)}$ be associated scores with an attribute j, $j = 1, \ldots, m$, of group A and B, respectively. Jones et al. (2007) presented from these definitions the following classification basic model.

$$
\begin{aligned}
\text{Minimize} \quad & z = \sum_{i=1}^{n_1} \left(n_i^{(a)}\right) + \sum_{i=1}^{n_2} \left(p_i^{(b)}\right) && (3.1)\\
\text{subject to} \quad & \sum_{j=1}^{m} x_{ij}^{(a)} w_j + n_i^{(a)} - p_i^{(a)} = w_0, \quad i = 1, \ldots, n_1, && (3.2)\\
& \sum_{j=1}^{m} x_{ij}^{(b)} w_j + n_i^{(b)} - p_i^{(b)} = w_0, \quad i = 1, \ldots, n_2, && (3.3)\\
& \sum_{j=1}^{m} w_j = 1, && (3.4)\\
& -\alpha \leq w_j \leq \alpha, \quad j = 1, \ldots, m, && (3.5)
\end{aligned}
$$

where $(w_1, \ldots, w_m)^T$ is the weight vector in m-dimensional space, and the discriminant line w_0 is defined by the following linear combination.

$$w_0 = w_1 y_1 + w_2 y_2 + \ldots + w_m y_m \tag{3.6}$$

Moreover, $y = (y_1, \ldots, y_m)^T$ gives the coordinates of a generic observation, and α is a user parameter defined so as to be significantly larger than the largest absolute value of the x vector (Jones et al. 2007).

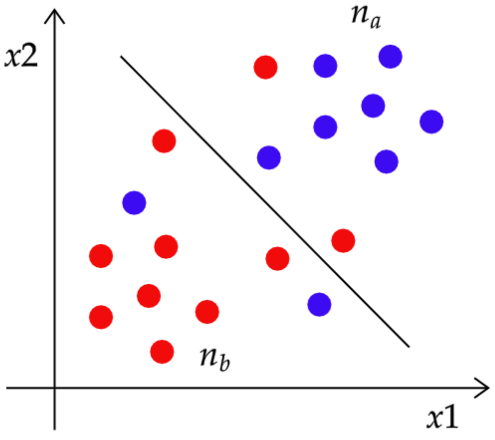


Figure 3.1: Illustration of a solution for the classification basic GP model

The basic model, represented in Figure 3.1, is made up of one objective function and four set of constrains. The first set of constraints (3.2) came from the equation $\sum_{j=1}^{m} x_{ij}^{(a)} w_j \geq w_0$, which represents all the data points which belong to group A should fall into the positive side of the discriminant line w_0 (blue points in Figure 3.1). Otherwise, the second set of constraints (3.3) came from the equation $\sum_{j=1}^{m} x_{ij}^{(b)} w_j \leq w_0$, and it represents the observations that belong to group B, and should fall into the negative side of w_0 (red points in Figure 3.1). Both (3.2) and (3.3) have negative ($n_i^{(a)}$, $n_i^{(b)}$) and positive ($p_i^{(a)}$, $p_i^{(b)}$) deviational variables that lead to the most accurate classification. For this, the objective function (3.1) seeks to minimize the deviational variables: the negative one $n_i^{(a)}$ for group A and the positive one $p_i^{(b)}$ for group B. Another both constrains are added for computational convenience only: (3.4) forces the sum of decision variables (except w_0) be equal to 1, and (3.5) states that each w_j will be between $-\alpha$ and α.

The model (3.1)–(3.5) tries to minimize the sum of distances between each observation and the separation hyperplane represented by the deviational variables $n_i^{(a)}$ and $p_i^{(b)}$. All three models presented in this section are based on Manhattan distance. However, it will lead to pull of some errors, as several observations could fall exactly into the discriminant line or deviational positive or negative variables may keep value of zero.

In order to avoid those errors, the next model was proposed.

$$\text{Minimize} \quad z = \sum_{i=1}^{n_1} \left(n_i^{(a)}\right) + \sum_{i=1}^{n_2} \left(p_i^{(b)}\right) \tag{3.7}$$

$$\text{subject to} \quad \sum_{j=1}^{m} x_{ij}^{(a)} w_j + n_i^{(a)} - p_i^{(a)} = w_0 + \beta, \quad i = 1, \ldots, n_1, \tag{3.8}$$

$$\sum_{j=1}^{m} x_{ij}^{(b)} w_j + n_i^{(b)} - p_i^{(b)} = w_0 - \beta, \quad i = 1, \ldots, n_2, \tag{3.9}$$

$$\sum_{j=1}^{m} w_j = 1, \tag{3.10}$$

$$-\alpha \leq w_j \leq \alpha, \quad j = 1, \ldots, m. \tag{3.11}$$

A new division zone with size 2β is created which is added to constrains (3.8) and (3.9); β is a parameter defined by the modeler. Figure 3.2 represents two parallel lines to w_0 (one to the left and the other to the right) with a division size of β so that five classification levels are declared as: "definitive A", "probable A", "unclassified", "probable B", "definitive B". Depending on β value, the probability of erring to classify each observation decreases since they are more likely to join the "probable" group. However, it also increases the probability of any observation relay in this class when it does not belong here.

As the penalization of non-achievement deviational variables is just a standard one in the previous model; another variation was proposed with a non-standard preference function (Jones &

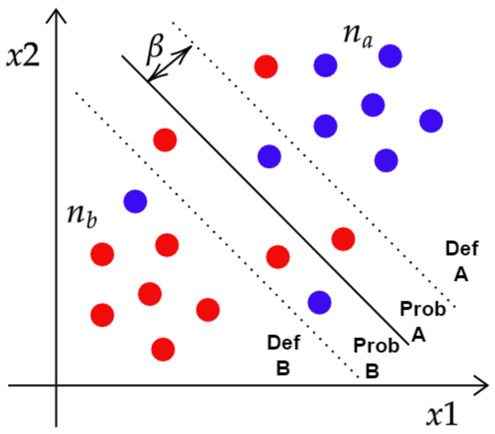


Figure 3.2: Illustration of a solution for the classification GP model with standard preference functions

Tamiz 1995). It establishes penalty weights and discontinuous penalties (Jones et al. 2007) that allows the modeler has more control in the classification probability (Figure 3.3).

$$\text{Minimize} \quad z = W_a \sum_{i=1}^{n_1} \left(u_1 n_{i1}{}^{(a)} + u_2 n_{i2}^{(a)} + u_3 n_{i3}^{(a)} \right) + W_b \sum_{i=1}^{n_2} \left(v_1 p_{i1}{}^{(b)} + v_2 p_{i2}^{(b)} + v_3 p_{i3}^{(p)} \right) \tag{3.12}$$

$$\text{subject to} \quad \sum_{j=1}^{m} x_{ij}^{(a)} w_j + n_{i1}^{(a)} - p_{i1}^{(a)} = w_0 - \beta, \quad i = 1, \dots, n_1, \tag{3.13}$$

$$\sum_{j=1}^{m} x_{ij}^{(a)} w_j + n_{i2}^{(a)} - p_{i2}^{(a)} = w_0, \quad i = 1, \dots, n_1, \tag{3.14}$$

$$\sum_{j=1}^{m} x_{ij}^{(a)} w_j + n_{i3}^{(a)} - p_{i3}^{(a)} = w_0 + \beta, \quad i = 1, \dots, n_1, \tag{3.15}$$

$$\sum_{j=1}^{m} x_{ij}^{(b)} w_j + n_{i1}^{(b)} - p_{i1}^{(b)} = w_0 - \beta, \quad i = 1, \dots, n_2, \tag{3.16}$$

$$\sum_{j=1}^{m} x_{ij}^{(b)} w_j + n_{i2}^{(b)} - p_{i2}^{(b)} = w_0, \quad i = 1, \dots, n_2, \tag{3.17}$$

$$\sum_{j=1}^{m} x_{ij}^{(b)} x_j + n_{i3}^{(b)} - p_{i3}^{(b)} = w_0 + \beta, \quad i = 1, \dots, n_2, \tag{3.18}$$

$$\sum_{j=1}^{m} w_j = 1, \tag{3.19}$$

$$-\alpha \leq w_j \leq \alpha, \quad j = 1, \dots, m, \tag{3.20}$$

where W_a and W_b are weights assigned for each class such that the assigned value will tell how important is to classified correctly one group respect another; further, it is also useful to give equal weight in the case of $n_1 \gg n_2$. For example, if the decision-maker wants to give major importance to classify group A than B, then $W_a > W_b$. Moreover, interior weight vectors u and v are related with the miss-classified penalty according to the associated levels: $w_0 + \beta$, w_0, and $w_0 - \beta$.

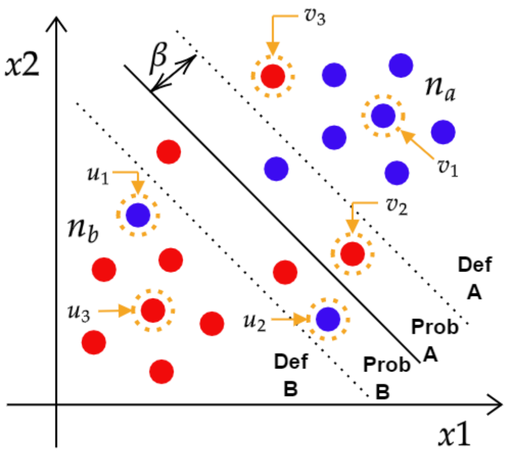


Figure 3.3: Illustration of a solution for the classification GP model with non-standard preference functions

The earlier model has a similar structure to the other two previously seen; however, new constrains are created for both groups: (3.13), (3.14), and (3.15) are related to the observations of group A; and (3.16), (3.17), and (3.18) are associated to data points on group B. Also, new negative ($n_{ik}^{(a)}$, $n_{ik}^{(b)}$) and positive ($p_{ik}^{(a)}$, $p_{ik}^{(b)}$) desviation variables are created, where k is related to the three objectives.

The objective function (3.12) seeks to minimize the sum of deviational variables, which are related to their respective internal weights and also the preferential class preference. Figure 3.3 shows that: v_1 and u_3 represent the penalization of miss-classification for “A defined as B” and “B defined as A” respectively; also, v_2 and u_2 represent the penalization of miss classified “A as probably B” and “B as probably A”, respectively; finally, u_1 and v_3 are associated to the classification of “A defined as A” and “B defined as B”, respectively.

The model (3.12)–(3.20) allows the modeler to establish his preferences concerning miss-classification weights. This choice improves the accuracy level after comparing the rates of correctly label objects as the previous models do not consider the difference amount of objects in each class or the importance of classifying one group from the other.

Observe that even though these models are based on Manhattan distance, Jones et al. (2007) also proposed an extension to distance metrics other than Manhattan. The following equation (3.21) represents the objective function for a generalized distance metric.

$$\text{Minimize} \quad z = \left[W_a^{\rho} \sum_{i=1}^{n_1} \left[u_1 n_{i1}^{(a)} + u_2 n_{i2}^{(a)} + u_3 n_{i3}^{(a)} \right]^{\rho} + W_b^{\rho} \sum_{i=1}^{n_2} \left[v_1 p_{i1}^{(b)} + v_2 p_{i2}^{(b)} + v_3 p_{i3}^{(b)} \right]^{\rho} \right]^{1/\rho}, \tag{3.21}$$

where ρ takes values in $[1,\infty)$. For the previously explained models, ρ takes value 1 in the objective function, i.e., the models are based on Manhattan distance and may give little sensitivity to outliers in the dataset. For $\rho = 2$, each model is based on Euclidean distance, and for $\rho = \infty$, each model is based on Chebychev distance. Jones et al. (2007) explained that the solution would vary depending on the value ρ in $[1,\infty)$, but if ρ is lesser than 1, then the model will take a non-convex structure, being harder to solve.

3.2 Clustering method based on integer programming

A clusterization model based on integer mathematical programming to solve the k-means clustering problem was proposed by Gnagi & Baumann (2017). In the proposed model, the input is a set of observations with size n and d attributes, and the modeler fixes the number of clusters. The following model assigns each object to a particular cluster center by minimizing the distances among objects.

$$\text{Minimize} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} \delta_{ij} z_{ij} \tag{3.22}$$

$$\text{subject to} \quad \sum_{i=1}^{n} z_{ij} = 1, \quad j = 1, \ldots, n, \tag{3.23}$$

$$\sum_{j=1}^{n} z_{ij} \leq n y_i, \quad i = 1, \ldots, n, \tag{3.24}$$

$$\sum_{i=1}^{n} y_i = k, \tag{3.25}$$

$$y_i, z_{ij} \in \{0,1\}, \quad i = 1, \ldots, n, \quad j = 1, \ldots, n, \tag{3.26}$$

where δ_{ij} represents the similarity between the objects i and j, and k is the number of clusters. The similarity δ_{ij} can be calculated using the generalized distance-metric, i.e., by using the distance metric parameter ρ, for $1 \leq \rho < \infty$. The binary variable z_{ij} represents the decision to choose the object i as a cluster center of the object j, and the binary variable y_i represents the decision to select the object i as a cluster center. Here, the objective function (3.22) seeks to minimize the total distance between the objects and each cluster center, considering three sets of constraints: (3.23) guarantees that each object is only assigned to one cluster center; (3.24) assures that at most n objects are assigned to each center cluster i; and (3.25) ensures that just k objects are selected as cluster centers.

Gnagi & Baumann (2017) illustrated in Figure 3.4 the optimal solution of the model (3.22)–(3.26) for an example containing 30 objects and 2 attributes.

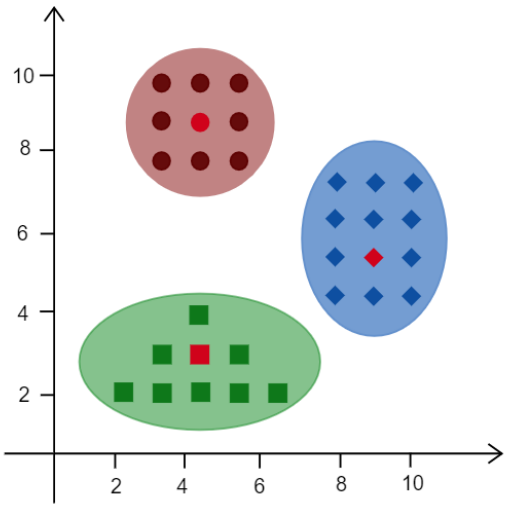


Figure 3.4: Data and optimal solution for $n = 30$ and $d = 2$ (Gnagi & Baumann 2017)

Note in Figure 3.4 that the modeler assumed three "natural" clusters. This example's objective function took value 37.1, which represents the total sum of Euclidean distances ($\rho = 2$) from all points belongs to an individual cluster to the center of this cluster (in red points). This solution had perfect accuracy by coinciding with the original label group, i.e., the number of clusters $k = 3$ fits the original groups.

However, since this model is object-oriented, the number of variables and constraints increases as the size of the data does, so it also spend more running time to obtain an optimal solution. To overcome this issue, an scaling approach of the previous model is proposed by Gnagi & Baumann (2017), which seeks a set of representatives that play correctly the original objects to decrease the number of variables and constrains.

Gnagi & Baumann (2017) uses a methodology based on sparse-reduced computation. It begins with the standardization of the data, which means that the set of features has average equal to zero and standard deviation equal to one. Then, a dimension reduction of the data set is made by using

Principal Component Analysis (PCA). The following step is to divide into ranges the values of the new features into intervals of equal length so that each object will belong to a single block. Finally, a single representative object from all non-empty blocks is computed as the center of gravity of the corresponded objects in the original dimensional space, i.e., before the reduced dimension.

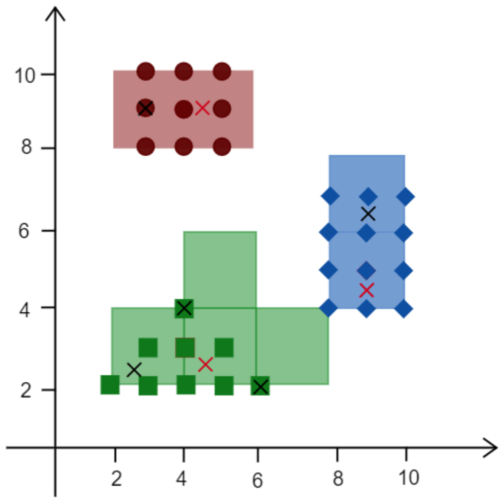


Figure 3.5: Optimal solution graphic with the reduced data (Gnagi & Baumann 2017)

The scale methodology seeks to use a clustering model based on the representatives to find a cluster center for the representatives and then transferred it on to the original objects. Applying this concept to the previous example, Figure 3.5 shows the scaling approach; and since the original data is in a small dimension ($d = 2$) non-projection to a smaller one is needed. The number of intervals of equal length is set as $g = 4$, for which the partition of the space is calculated as g^p grid blocks. In this case, the number of grids is $4^2 = 16$, with a length of 2. After mapping the 30 original objects, from each non-empty grid block, a representative is chosen. Then, $q = 8$ is represented by a cross maker in Figure 3.5. Those representatives were used to solve the following scaling formulation.

$$\text{Minimize} \quad z = \sum_{r=1}^{q} \sum_{l=1}^{q} \rho_{rl} z_{rl} \tag{3.27}$$

$$\text{subject to} \quad \sum_{r=1}^{q} z_{rl} = 1, \quad l = 1, \ldots, q, \tag{3.28}$$

$$\sum_{l=1}^{q} z_{rl} \leq q y_r, \quad r = 1, \ldots, q, \tag{3.29}$$

$$\sum_{r=1}^{q} y_r = k, \tag{3.30}$$

$$y_r, z_{rl} \in \{0, 1\}, \quad r = 1, \ldots, q, \quad l = 1, \ldots, q, \tag{3.31}$$

where ρ_{rl} is the distance between the representative r and l, and both z_{rl} and y_r are binary variables, where z_{rl} represents the decision to choose an object r as cluster center of object l, and y_r represents which object r is design as cluster center.

As the model remains with the same structure as the model (3.22)–(3.26), the objective function and constraints seek the same goal but in a different dimension. Likewise, in Figure 3.5 the cluster center for the representatives is represented as red cross makers, and it attaches a perfect clustering accuracy after transferred the cluster representative on to the original objects. The number of variables and constraints greatly decreases when the scale approach to the clustering model based on integer mathematical programming is applied. Moreover, this scale model (3.27)–(3.31) used less running time and the solution reaches the same level of accuracy or even improved the ones obtained from the object oriented model (3.22)–(3.26).

CHAPTER 4

Numerical experiments

This chapter presents the computational experiments for the classification and clustering methods previously explained. Additionally, it gives a series of illustrations, tables, and statistics summarizing the collected results. The set of numerical examples includes two case studies. In the first case, the well-know database “Breast Cancer Wisconsin (BCW)” contains 569 observations with 30 features that characterize malignant and benign breast cancer tumors. A new database was selected in the second case, which corresponds to 526 bovine animals with their respective characteristics.

The mathematical models and routines were coded in Julia language using IBM ILOG CPLEX 12.9 as the solver. The tests were executed on a computer with an Intel Core i7-6500 processor, 7.89GB of RAM, and a Windows operating system.

4.1 Numerical experiments for classification

This section analyzes different tests in order to establish the best way to choose the parameter values for the three classification models based on the goal programming technique described in Section 3.1. Hence, the obtained parameters were used for setting the classification models applied on the first database BCW.

4.1.1 Setting the parameters

The user parameters α and β that appear in the models described in Section 3.1 are essential components to build the classification models based on goal programming. In particular, see α in (3.5) and β in (3.8). This subsection presents preliminary numerical tests for setting the parameters α and β. Two simple examples of Freed & Glover (1981, 1986) provide the data for these tests. The data listed in Tables 4.2 and 4.5, respectively. Two classes of inputs form each preliminary numerical example, and each class contains two features and five observations, making it easier to represent graphically.

To facilitate the presentation of tests, we label the three goal programming models studied in Section 3.1. GP1 is the goal programming model (3.1)–(3.5) without the parameter β. GP2 is the model (3.7)–(3.11) that adds the parameter β. Finally, GP3 is the model (3.12)–(3.20) that considers all preference weights. The two numerical examples were used to solve the three models.

We handled four tests concerning the variable vector $w = (w_1, \ldots, w_m)^T$, more specifically, the range for w_j. Table 4.1 outlines each one of these tests. In Test A, the range for w_j is $[0,1]$. In Test B, a normalization of w states as $|w_j| \leq 1$, referring to research of Nakayama & Kagaku (1998) on classification problem for Support Vector Machine based in goal programming. In Test C, w_j is a free variable. And in Test D, we consider the normalization of w described by Jones et al. (2007), where $|w_j| \leq \alpha$ and α takes a value significantly larger, for example, the largest absolute value among the input data.

Freed & Glover (1981) used the data of Table 4.2 to study different approaches for the discriminant problem via goal programming technique. In short, they consist of finding a discriminant hyperplane that entirely separates the input classes. Note that in this example, since we have two

Test	Description
A	$0 \leq w_j \leq 1$
B	$|w_j| \leq 1$ and constraint $\sum_{j=1}^{n} w_j = 1$ is deleted
C	w_j is a free variable
D	α is the largest input in absolute value

Table 4.1: Test scenarios and their respective description

Class 1

x1	x2
1	1
2	2
3	1
3	3
6	3

Class 2

x1	x2
4	1
5	2
7	2
8	4
9	1

Table 4.2: Data used in Figures 4.1 and 4.2

features **x1** and **x2**, then the hyperplane is a linear function. Figure 4.1 illustrates the data belong to Class 1 and Class 2 by red and blue points, respectively.

Tables 4.3, 4.4, 4.6, 4.7, 4.10, and 4.11 summarize the numerical outputs of Tests A, B, C, and D concerning to used models. The results are shown for the objective function (OF), weight vector components $w = (w_0, w_1, w_2)^T$, the error in classification Class 1 (error_c1), the error in classification Class 2 (error_c2), the false positive rate (FP), and the true positive rate (TP).

Table 4.3 gives the outputs of tests realized by GP1 model. In test A, the OF achieves the value 4, which does not indicate a perfect classification; error_c1 and error_c2 confirm the presence of observations from each class wrongly classify. Furthermore, since w_1 took value equal to zero, the classification line is a horizontal line (see Figure 4.2(a)). In test B, OF, w_0, w_1, and w_2 took values equal to zero, which evidence, for these conditions, that the model does not have a hyperplane that divides both classes (see Figure 4.2(b)). Furthermore, the quantity of observations wrongly classified is zero because it does not occur any classification.

Distinctly test A and B, the tests C and D achieve the same results Freed & Glover (1981), where $w_0 = -2$, $w_1 = -1.49$, and $w_2 = 2.49$ bring together a perfect division line for both classes

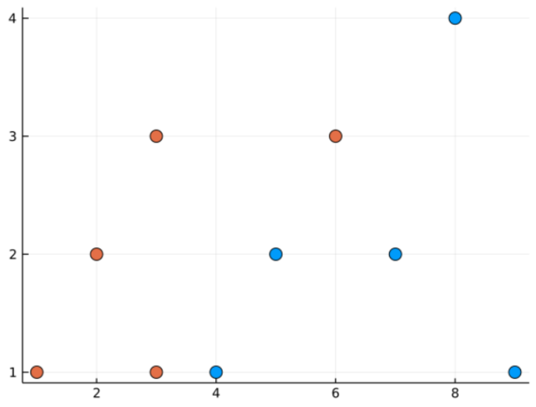
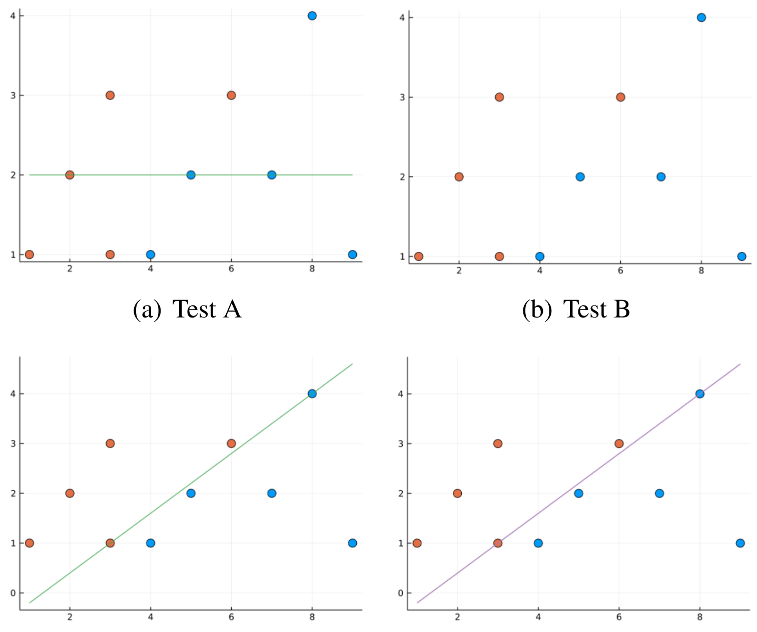


Figure 4.1: Data set used by Freed & Glover (1981)

	Test			
	A	B	C	D
OF	4	0	0	0
w_0	2	0	-2	-2
w_1	0	0	-1.49	-1.49
w_2	1	0	2.49	2.49
error_c1	1	0	0	0
error_c2	2	0	0	0
FP	0.33	0	0	0
TP	0.75	1	1	1

Table 4.3: Results from the GP1 model illustrated in Figure 4.2

with a minimal value for OF. Figure 4.2(c) and 4.2(d) illustrate both solutions graphically.



(a) Test A (b) Test B

(c) Test C (d) Test D

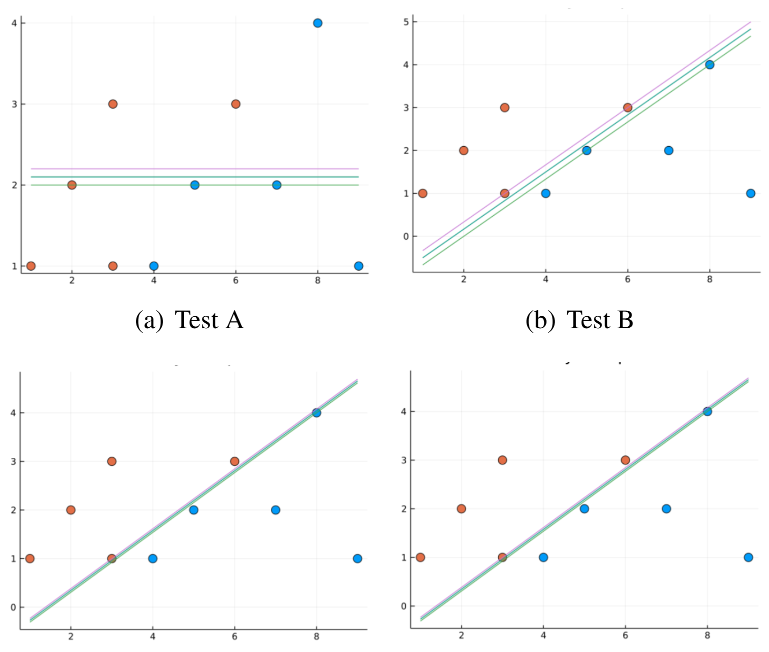
Figure 4.2: GP1 model results considering four tests from data set of Freed & Glover (1981)

As previously observed, the GP1 model considers only the parameter α in its structure; thus GP2 model introduces the parameter β in the constraints (3.8)–(3.9) creating two parallel division lines. Table 4.4 shows the results of applying GP2 model in the same four tests seen above considering $\beta = 0.1$. In Test B, the results of Table 4.3 differ from Table 4.4 significantly, where is found a hyperplane that correctly separates both classes. Moreover, the graphic results for four tests have shown that another two joined parallel lines establish the five levels of classification explained in Chapter 3.

From the graphics in Figure 4.3, just one scenario does not generate a classification line with

	Test			
	A	B	C	D
OF	4.06	0	0	0
w_0	2.01	-0.07	-2.03	-2.03
w_1	0	-0.04	-1.51	-1.51
w_2	1	0.06	2.51	2.51
error_c1	1	0	0	0
error_c2	3	0	0	0
FP	0.43	0	0	0
TP	0.67	1	1	1

Table 4.4: Results from the GP2 model illustrated in Figure 4.3



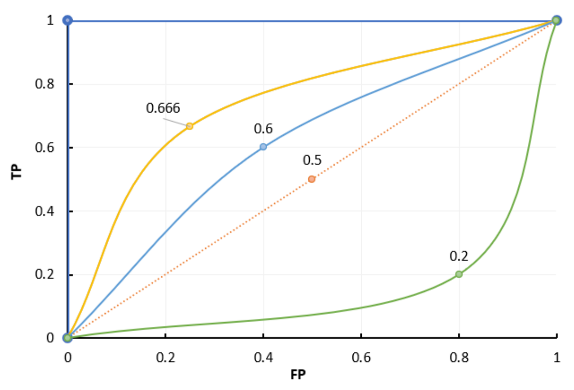
(a) Test A (b) Test B

(c) Test C (d) Test D

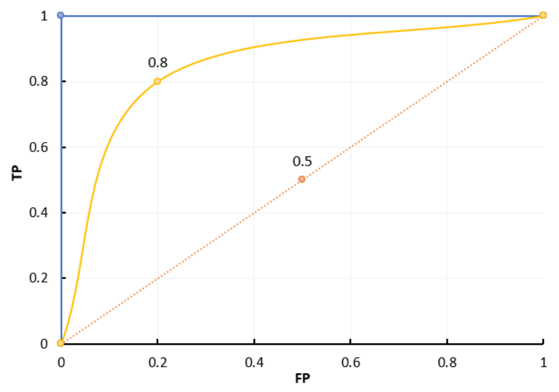
Figure 4.3: GP2 model results considering four tests from data set of Freed & Glover (1981)

a slope, which is the case of Test A as it still is a horizontal line. Although all other tests have a precise classification, it is necessary to evaluate the performance when β changes of values. For this purpose, four additional scenarios were applied considering $\beta = 0.5, 1, 2, 5$. For each β, a curve ROC is given, which evaluates the performance between False Positive and True Positive rates. If a perfect classification exists, then TP=1 and FP=0, and only a curve with TP greater than 0.5 (the orange diagonal line in Figure 4.4) could be considered as acceptable classifier.

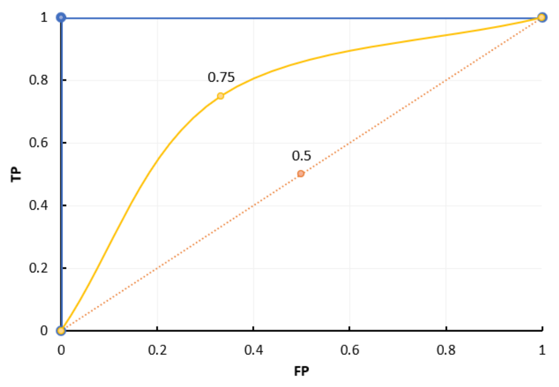
Figure 4.4 exhibits the ROC curves for the four different values of β in Test B, C, and D. In Test B three curves are in the acceptable zone determined by the orange diagonal line with TP=0.5. If $\beta = 0.5$, the model has a perfect classification with TP=1 and FP=0. The following yellow curve



(a) ROC curve for Test B



(b) ROC curve for Test C



(c) ROC curve for Test D

Figure 4.4: ROC curves for Tests B,C, and D

with TP=0.66 represents the scenario $\beta = 1$. Moreover, if $\beta = 2$ the blue curve is not a perfect classification and has TP=0.6. Although the last two scenarios do not make a perfect classification, they are considered suitable classifiers. However, if $\beta = 5$, the green curve with TP=0.2 takes place under the orange diagonal line that certifies that the model for this scenario is no longer considered a good classifier. This is also observed from the values of variables n_a and p_b since they have taken positive values, then those points are miss classified.

Test C and D have a similar performance at the moment of changing β since both achieve a stable TP and FP above the orange line when β is higher than 1. This results correspond to which observations were miss classified, being only two from Class 1 and one from Class 2, that corresponds to the ones closer to the division line (Figure 4.3(c) - 4.3(d)). GP1 and GP2 performed better in Test C and D as they all attained the same optimal solution presented in the research of Freed & Glover (1981). Furthermore, we notice that GP2 has optimal solution when β is higher than 1, so we establish a relationship between the data, and β must correspond to the data variance, in this case of 1.43.

Class 1

x1	x2
-1	3
-1	4
1	3
4	9
2	7

Class 2

x1	x2
0	0.25
2	3
-4	-9
3	4
-1	-3

Table 4.5: Data used in Figures 4.5 and 4.6

To validate the previous analysis, another set of data is used to solve the GP1 and GP2 model. Freed & Glover (1986) uses the data in Table 4.5 to resolve certain difficulties founded in his previous research (Freed & Glover 1981) by including an appropriate normalization. Note that the data belong to Class 1 and Class 2 are represented by red and blue points respectively in Figure 4.5.

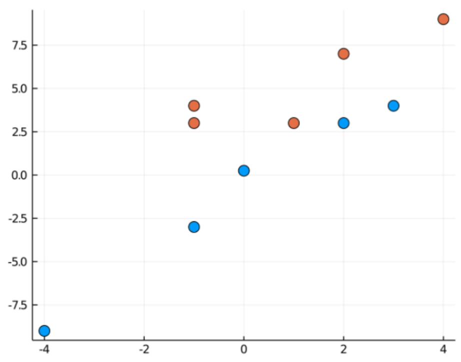


Figure 4.5: Data set used by Freed & Glover (1986)

The same four tests were used in this experiment, and its results of model GP1 are given in Table 4.6. Similarly, than the previous data set, Test A is not a reliable classifier as its objective function (OF) does not attain the minimum, even though just one observation is miss classified. Freed & Glover (1986) disclose in his research that the optimum solution would be when $w_0 = 5$, $w_1 = 2$, and $w_2 = 4$ which comparing to Table 4.6, only Test C and D achieve these values. Despite that Test B finds a vector w that makes a perfect separation line in Figure 4.6(b) is convenient to come with a definitive decision after seeing the results of the model GP2 including the value β.

Similarly to the first analysis using the previous data set, the behavior of model GP2 is evaluated according to the variation of β. In Table 4.7, the results for $\beta = 0.1$ are displayed and confirm that Test A is not suitable for our model because it does not achieve a perfect classification. Figure 4.7 illustrates the graphic representations for the four tests. Additionally, no bigger changes were detected in Test B, C, and D compared to Table 4.6, therefore the analysis of impact changing of β

	Test			
	A	B	C	D
OF	1	0	0	0
w_0	3	0.12	5	5
w_1	0	-1	-1	-1
w_2	1	0.46	2	2
error_c1	1	0	0	0
error_c2	0	0	0	0
FP	0	0	0	0
TP	0.83	1	1	1

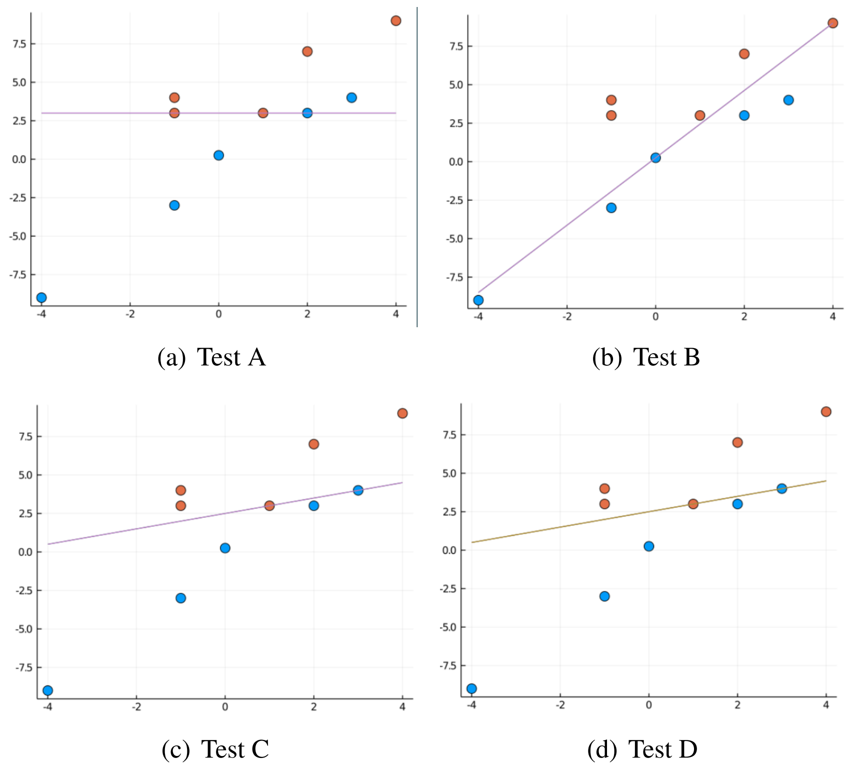
Table 4.6: Results from the GP1 model illustrated in Figure 4.6

helps to choose which scenario performs better.

Figure 4.8 illustrates the ROC curves for Test B and D and shows the impact on the classification power according to the variation of β. Conversely, Test C does not present a ROC curve as its performance did not change in any β and it maintains classifying correctly. Again, Test B has an acceptable level of classification for $\beta = 0.5, 1, 2$ because those three curves are above the diagonal orange line that divides the two regions of acceptance in the graphic. However, when $\beta = 5$, the model reaches TP=0.43 that makes the classifier weaker as it misclassifies two and four observation points from Classes 1 and 2, respectively. ROC curve for Test D presents a different performance from the previous data set. For the first three scenarios, the model found a perfect classification line, but when β change to 5, its FP value decreases to 0.166. This variation only made one miss classification related to an observation of Class 1 that is closer to the division line (Figure 4.7(d)); however, the ROC curve still is above the orange division line, so the classifier is adequate.

	Test			
	A	B	C	D
OF	1.8	0	0	0
w_0	3.2	0.33	5.03	5.03
w_1	0	-1	-1.02	-1.02
w_2	1	0.51	2.02	2.02
error_c1	1	0	0	0
error_c2	2	0	0	0
FP	0.33	0	0	0
TP	0.75	1	1	1

Table 4.7: Results from the GP2 model illustrated in Figure 4.7



(a) Test A (b) Test B

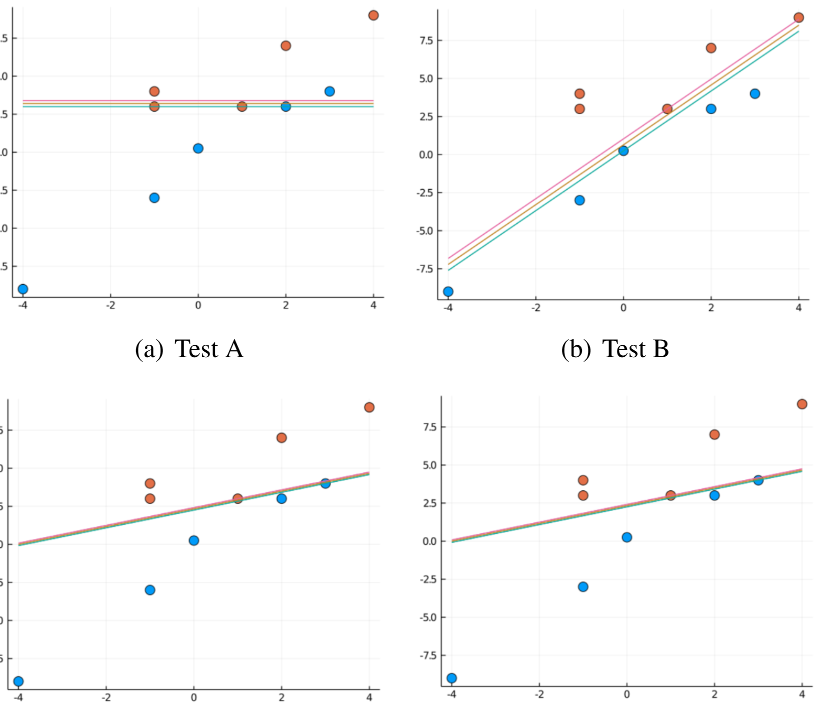
(c) Test C (d) Test D

Figure 4.6: GP1 model results considering four tests from data set of Freed & Glover (1986)

Finally, we can find a relationship between the variation of the data and the value of β. As analyzed previously, we have chosen to vary β to identify an approximate value for which the model starts to lose its classification power. Note that the addition of parameter β on the goal programming model improves the model's accuracy rate. Thus, we evaluated the growth of β and compared it with the standard deviation of data. This last data set has the standard deviation equal to 2.81, which coincides with the larger value β that the model started the incorrect classification, i.e., after analyzing GP1 and GP2 models, the best two scenarios for both models are Test C and D for which β is no bigger than the standard deviation of data set.

Therefore, to analyze the GP3 model, we only consider those statements but with a further overview of establishing the preference weights. GP3 model is characterized by using penalty weights applied to the objective function. As detailed in Chapter 3, Jones et al. (2007) introduced two-class of weights, W_a and W_b are preference weights for the objective function, and the internal weights u and v penalize each incorrect classification in addition to the associated boundary. Table 4.9 presents the six internal weights with its corresponding penalization of unclassified one observation defined as or probably to a class that does not belong.

Indeed, increasing any weight value has its pros and cons. For example, the weights u_1,v_3, u_2,


(a) Test A (b) Test B

(c) Test C (d) Test D

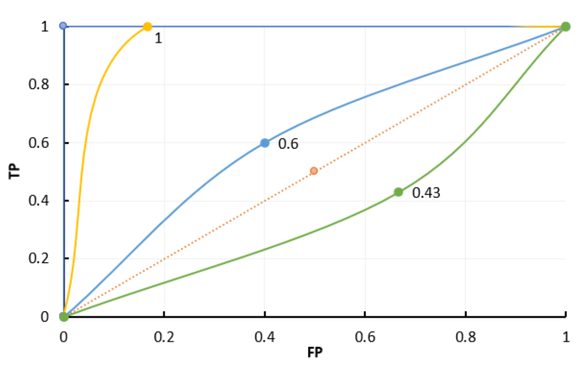
Figure 4.7: GP2 model results considering four tests from data set of Freed & Glover (1986)

and v_2 may improve the correct classification; however, it also may reduce the power classification and, if u_2 or v_2 are too large, the model can classify an observation "A defined as B", i.e., an observation from Class A becomes of Class B. Moreover, the weights u_3 and v_1 enlarge the number of observations that are correctly classified, i.e., "A defined as A", and "B defined as B"; but does not reduce the unclassified observations of type "A defined as B" or "A as probably B". Table 4.8 highlights the additional pros and cons for the GP3 model.

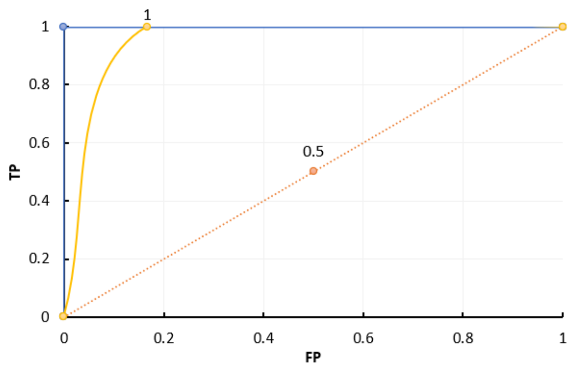
weight	type of error	Pros	Cons
u_1	A defined as B	Lower level of miss classification	Reduce the power of classification
u_2	B defined asA		
u_3	A as probably B		Increase power of classification A defined as B
v_1	B as probably A		
v_2	A defined as A	Increase power of classification A defined as A, and B defined as B	Do not reduce miss classification of A defined as B and A as probably B
v_3	B defined as B		

Table 4.8: Description of penalty internal weights for the GP3 model

The preference weights W_a and W_b indicate the importance of each class to be classified. For both examples, Database 1 (DB1) of Freed & Glover (1981) and Database 2 (DB2) of Freed & Glover (1986), we select Class 2 as the priority class and we use $W_a := W_{C1} = 0.3$ and $W_b := W_{C2} = 0.7$. Moreover, to give the most accurate values for the internal weights u and v, we use the



(a) ROC curve for Test B



(b) ROC curve for Test D

Figure 4.8: ROC curves for Tests B, and D

Ranking Sum (RS) and Sum Reciprocal (SR) methods described in Danielson & Ekenberg (2017). From a ranking known in advance that selects from the more to the less priority error in a correct classification (each internal weight is following a priority level according to a type of error), RS and SR calculate each internal weight as presented in Table 4.9. The RS and SR methods use the equations (4.1) and (4.2), respectively, to obtain each internal weight from a known ranking and by normalizing via the sum of the ranking in (4.1), and via an additive combination of Sum and Reciprocal (Stillwell et al. 1981) weight functions in (4.2).

$$w_i^{RS} = \frac{N+1-i}{\sum_{j=1}^{N}(N+1-j)} \tag{4.1}$$

$$w_i^{SR} = \frac{\frac{1}{i}+\frac{N+1-i}{N}}{\sum_{j=1}^{N}\left(\frac{1}{j}+\frac{N+1-j}{N}\right)}, \tag{4.2}$$

where N is the amount of internal weights, $i = 1,\ldots,N$, and it is assumed that $w_1 > w_2 > \ldots > w_N$, for which $\sum w_i = 1$ and $w_i \geq 0$. Note that the chosen ranking in Table 4.9 considers Class 2 (C2) as a priority in classifying over Class 1 (C1).

weight	type of error	ranking	RS	SR
u_1	C1 defined as C2	3	0.190	0.190
u_2	C1 as probably C2	1	0.286	0.334
u_3	C1 defined as C1	5	0.095	0.074
v_1	C2 defined as C1	6	0.048	0.018
v_2	C2 as probably C1	2	0.238	0.253
v_3	C2 defined as C2	4	0.143	0.131

Table 4.9: Penalty internal weights obtained via RS and SR methods

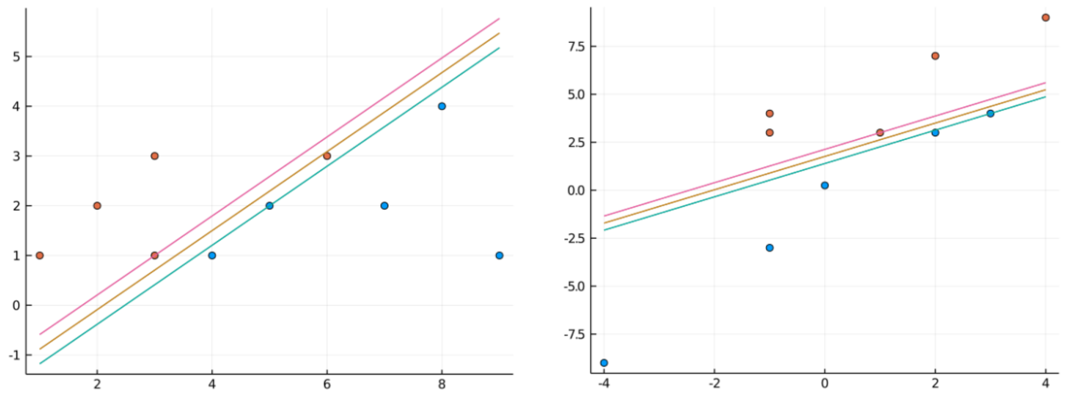
We applied both sets of weights to datasets DB1 and DB2 in the two best scenarios found previously: Test C and D. However, no difference between the results from Tests C and D were found in either RS or SR weights. Figure 4.9 displays a graphic representation of DB1 (Figure 4.9(a)) and DB2 (Figure 4.9(b)); thus is clear that the principal division responds to the classification priority as the distance between observations from Class 2 and the division line is bigger than Class 1.

	Test	
	C	D
OF	0.18	0.18
w_0	-8.15	-8.15
w_1	-3.86	-3.86
w_2	4.86	4.86
err_c1	1	1
err_c2	0	0
FP	0.17	0.17
TP	1	1

Table 4.10: Results from the GP3 model illustrated in Figure 4.9(a)

	Test	
	C	D
OF	0	0
w_0	13.43	13.43
w_1	-6.62	-6.62
w_2	7.62	7.62
err_c1	0	0
err_c2	0	0
FP	0	0
TP	1	1

Table 4.11: Results from the GP3 model illustrated in Figure 4.9(b)



(a) Test C and D for DB1

(b) Test C and D for DB2

Figure 4.9: GP3 model results considering two tests from data sets DB1 and DB2, and SR or RS methods

The conclusion is that the parameter analysis for GP1 and GP2 models lead to validate that either Test C or Test D gives the same results considering β equal or lower than the data set variation. Moreover, the weights structure of the GP3 model explained with two types of weighting for internal weights do not present a significant difference in results; it is suggested by Danielson & Ekenberg (2014) to use the SR method as it is more robust than the RS method. The following section is oriented to apply the configured parameters and weights into a larger breast cancer tumor database, where the three models would try to classify if a tumor mass is benign or malign.

4.1.2 Case of study: Breast Cancer

In this subsection, we use the parameters α and β previously configured to study the well-known database of breast cancer tumors that diagnoses if a breast mass is benign or malign, considering a set of thirty characteristics.

In 2018, according to Wild et al. (2020), cancer was categorized as the first or second cause of premature death (from 30 to 69 years), and six types of cancer are the most common: lung cancer, breast cancer, colorectal cancer, prostate cancer, stomach cancer, and cervical cancer. For women, breast cancer is the most common, with 2.1 million cases and a mortality rate of 627 000 in 2018.

Since breast cancer has a significant impact on women's health, several studies have been made, like prediction models for breast cancer risk. Shawky et al. (2017) used the mammographic density (MD) for a breast cancer prediction model that delimits who will most benefit from chemoprevention or other prevention efforts. Mavaddat et al. (2015) implemented a breast cancer risk model for women considering the cancer family history from a polygenic risk score (PRS). A most recent study made by Zhang et al. (2018) joined MD, PRS, and postmenopausal endogenous hormone levels to improve existing prediction models. The overview made by Cintolo-Gonzalez et al. (2017) discussed existing models for breast cancer risk that use hormonal and environmental factors focusing on hereditary risk. The problem involves diagnosing breast masses as benign or malignant by using computer software called *Xcyt*, which is an image analysis program that estimates the probability of the malignancy of a breast lump or mass. The process begins by taking a sample directly from the breast lump or mass with a small needle called fine needle aspirate (FNA). The

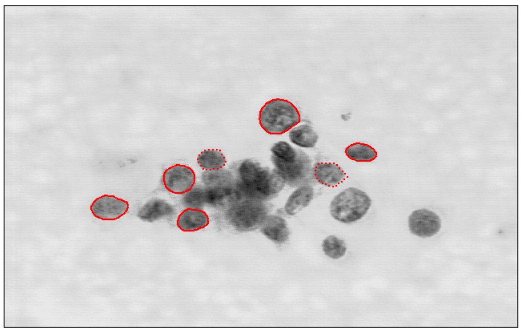


Figure 4.10: Image of a malignant breast with nuclei cells analyzed with *Xcyt*. (Mangasarian et al. 1970)

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
mean	14.13	19.29	91.97	654.89	0.10
std	3.52	4.30	24.30	351.91	0.01
min	6.98	9.71	43.79	143.50	0.05
25%	11.70	16.17	75.17	420.30	0.09
50%	13.37	18.84	86.24	551.10	0.10
75%	15.78	21.80	104.10	782.70	0.11
max	28.11	39.28	188.50	2501.00	0.16

Table 4.12: Descriptive statistic of the BCW database for 5 features

fluid is then analyzed in a microscope, so the constituent cells' nuclei are highlighting, and an image is taken of those cells' nuclei. *Xcyt* is used to fit a curve around the boundaries of the nuclei cells, which are between 10 and 40 per image, as in Figure 4.10; moreover, ten features are computed for each nucleus: area, radius, perimeter, symmetry, number and size of concavities, fractal dimension (boundary), compactness, smoothness, and texture. Finally, calculate the mean value, extreme value (large or worst), and standard error of each nucleus characteristic that, in total, gives thirty features.

	compactness mean	concavity mean	concave points mean	symmetry mean	fractal dimension mean
mean	0.10	0.09	0.05	0.18	0.06
std	0.05	0.08	0.04	0.03	0.01
min	0.02	0.00	0.00	0.11	0.05
25%	0.06	0.03	0.02	0.16	0.06
50%	0.09	0.06	0.03	0.18	0.06
75%	0.13	0.13	0.07	0.20	0.07
max	0.35	0.43	0.20	0.30	0.10

	radius se	texture se	perimeter se	area se	smoothness se
mean	0.41	1.22	2.87	40.34	0.01
std	0.28	0.55	2.02	45.49	0.00
min	0.11	0.36	0.76	6.80	0.00
25%	0.23	0.83	1.61	17.85	0.01
50%	0.32	1.11	2.29	24.53	0.01
75%	0.48	1.47	3.36	45.19	0.01
max	2.87	4.89	21.98	542.20	0.03

	compactness se	concavity se	concave points se	symmetry se	fractal dimension se
mean	0.025	0.032	0.012	0.021	0.004
std	0.018	0.030	0.006	0.008	0.003
min	0.002	0.000	0.000	0.008	0.001
25%	0.013	0.015	0.008	0.015	0.002
50%	0.020	0.026	0.011	0.019	0.003
75%	0.032	0.042	0.015	0.023	0.005
max	0.135	0.396	0.053	0.079	0.030

Table 4.13: Descriptive statistic of the BCW database for 15 features

All these 30 features are in the breast cancer Wisconsin (diagnosis) database published by Wol-

	radius worst	texture worst	perimeter worst	area worst	smoothness worst
mean	16.27	25.68	107.26	880.58	0.13
std	4.83	6.15	33.60	569.36	0.02
min	7.93	12.02	50.41	185.20	0.07
25%	13.01	21.08	84.11	515.30	0.12
50%	14.97	25.41	97.66	686.50	0.13
75%	18.79	29.72	125.40	1084.00	0.15
max	36.04	49.54	251.20	4254.00	0.22

	compactness worst	concavity worst	concave points worst	symmetry worst	fractal dimension worst
mean	0.25	0.27	0.11	0.29	0.08
std	0.16	0.21	0.07	0.06	0.02
min	0.03	0.00	0.00	0.16	0.06
25%	0.15	0.11	0.06	0.25	0.07
50%	0.21	0.23	0.10	0.28	0.08
75%	0.34	0.38	0.16	0.32	0.09
max	1.06	1.25	0.29	0.66	0.21

Table 4.14: Descriptive statistic of the BCW database for 10 features

berg & Mangasarian (1989), which performed the previous analysis in 569 patients. The actual diagnostic outcome is known: 357 benign and 212 malignant breast tumor cases. Table 4.12 only presents the first five features with its mean, standard deviation (std), minimum (min) and maximum value (max), and the first, second, and third quartile (25%, 50%, and 75%). The set of supplementary tables presented in Tables 4.13 and 4.14 help understand the distribution of data in the database to execute the classification correctly. For example, variables with a high standard deviation indicate the data points are extensively spread around the central point (mean). This observation helps the class model as it is better to apply it in an outspread area. Conversely, low standard deviation leads to data points be closer to the center point, as it is appreciated in Table 4.13 and Table 4.14 for the following variables: concavity mean, and concave points mean.

Also, a pairwise plot is made to look up for a possible relationship between the features. In Figure 4.11, for the first five features (the mean values of radius, texture, perimeter, area, smoothness, compactness, and concavity), three linear relationships were found that suggest a possible correlation between them too. Those associations (with their correlation value) are the mean values of: radius with area (0.99), radius with perimeter (0.99), and area with perimeter (1).

A correlation heatmap presented in Figure 4.12 confirms that some features had high correlation between them. For example, the correlation between radius mean and area mean is 0.99, and it takes a darker color as it is closer to 1. Thus, the correlation heatmaps in Figures 4.13, 4.14,

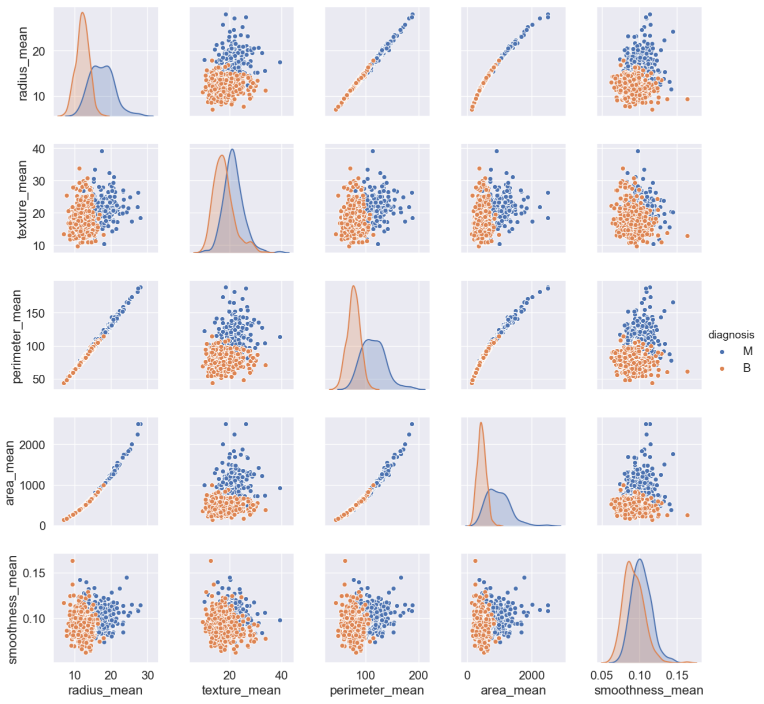


Figure 4.11: Pairwise plot of the BCW database

and 4.15 can be read just by seeing how the color intensity change to a more darker one.

The correlation between variables were: from Figure 4.13 perimeter mean and area mean, radius mean and perimeter mean, area mean and radius mean; from Figure 4.14 radius se and perimeter se, radius se and area se, and perimeter se and area se; and from Figure 4.15 perimeter worst and radius worst, radius worst and area worst, and perimeter worst and area worst.

The binary classification model based on goal programming was used: a basic Model 1 called as GP1 formulated by (3.1)–(3.5); a second Model 2 named as GP2 which considers a division zone of 2β formulated by (3.7)–(3.11) and a third Model 3 called GP3 that considers penalization via internal weights following the structure formulated by (3.12)–(3.20). As the database has a higher number of benign than malign cases, the preference weights are used to compensate this difference by setting $W_a = 0.9$ and $W_b = 0.3$. Table 4.15 shows the rank of importance of each internal weight and its corresponding values after applying the Sum Reciprocal (RS) method.

These three models were tested in the two test scenarios (Tests C and D) presented in the previous section and considering that β in GP2 and GP3 must be no higher than the database variation. A general view of how the classification algorithm works is described in Algorithm 1.

The database was divided into two sets, the training and test sets. The former is composed of 80% of the data, and the latter has 20% of the remaining data. A K-fold cross-validation is used to prevent an overfitting model by partitioning the training set into k subsets, and it uses $k-1$ to train

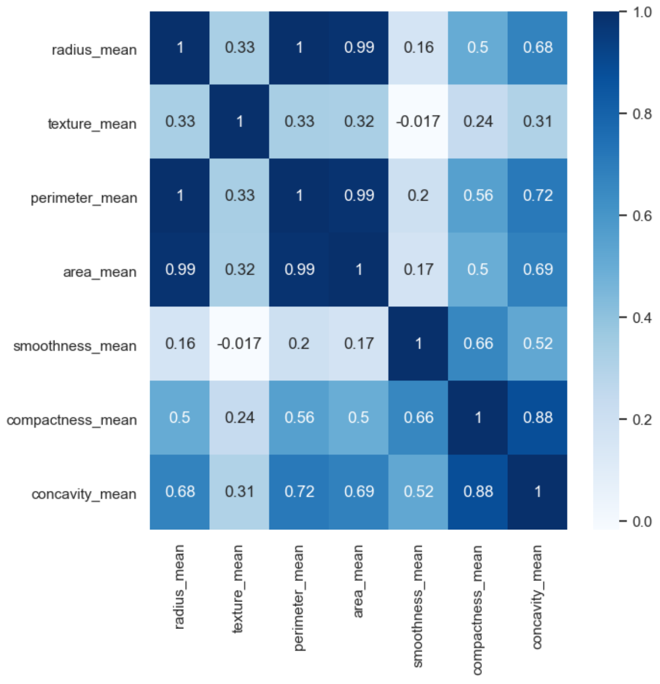


Figure 4.12: Correlation heatmap of the BCW database for 7 features

weight	type of error	ranking	RS
u_1	Benign defined as Malign	5	0.074
u_2	Benign as probably Malign	6	0.018
u_3	Benign defined as Benign	2	0.253
v_1	Malign defined as Malign	3	0.190
v_2	Malign as probably Benign	4	0.131
v_3	Malign defined as Benign	1	0.334

Table 4.15: Penalty internal weights obtained via RS method for the GP3 model

and the remaining k subsets to validate. Moreover, a confusion matrix (Table 4.16) was built to validate each developed model. It is composed of four variables: True positive (TP), True negative (TN), False negative (FN), and False positive (FP).

We have the following for this database: true positive (TP) relates to the benign cases which were correctly classified; false negative (FN) express the benign tumors classified as malign; false positive (FP) is the number of malign cases which were predicted as benign; and True negative (TN) represents the malignant tumors correctly classified. Each one of the four variables calculates the following ratios: the accuracy rate measures how correctly a model classified each group; the error rate or incorrect classification rate represents the fraction of error; the sensitivity and specificity rate are the portion of recognize for benign and malign classes respectively; the precision and recall rate measure the exactness and the completeness of the model respectively.

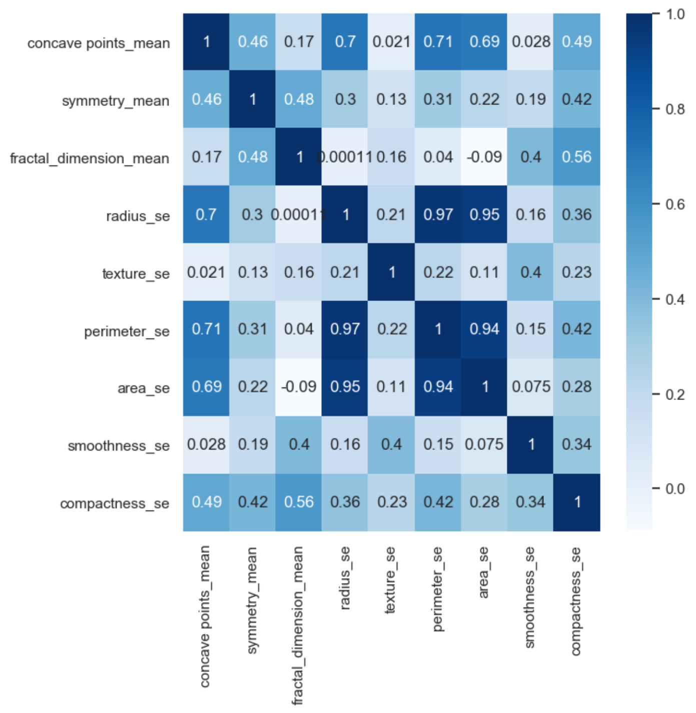


Figure 4.13: Correlation heatmap of the BCW database for 9 features

Since $k = 10$ on the K-fold cross-validation, then the model evaluates nine different subsets as a training set. Figure 4.16 shows the average value of accuracy and error rate for each model, where the former is the blue curve, and the latter is the orange curve. The accuracy levels are higher, and the error rate levels are lower, indicating the model can find a hyperplane that classifies correctly. However, this cannot be taken as a performance measure because it only considers the training phase.

Figure 4.17 displays both curves of average accuracy and error rate for all k validation subsets. Similarly to Figure 4.16, it has higher and lower values for accuracy and error rate, respectively, indicating that each model does not have overfitting. Moreover, the accuracy level performance increases from GP1 to GP2, but it decreases on GP3; the error rate converges with the accuracy level as it decreases from GP1 to GP2, but it increases on GP3.

After proving no overfitting exist in all models, we used the 20% remaining of data to test each model, including 114 observations. A comparative chart of accuracy level in the training, validation, and test phases is displayed in Figure 4.18, where all models reach acceptable accuracy levels that are higher than 0.93. Additionally, Table 4.17 presents six indicators to analyze the performance of the GP1, GP2, and GP3 models when Test C or Test D are applied. Next, we summarize the obtained results for each indicator of measure.

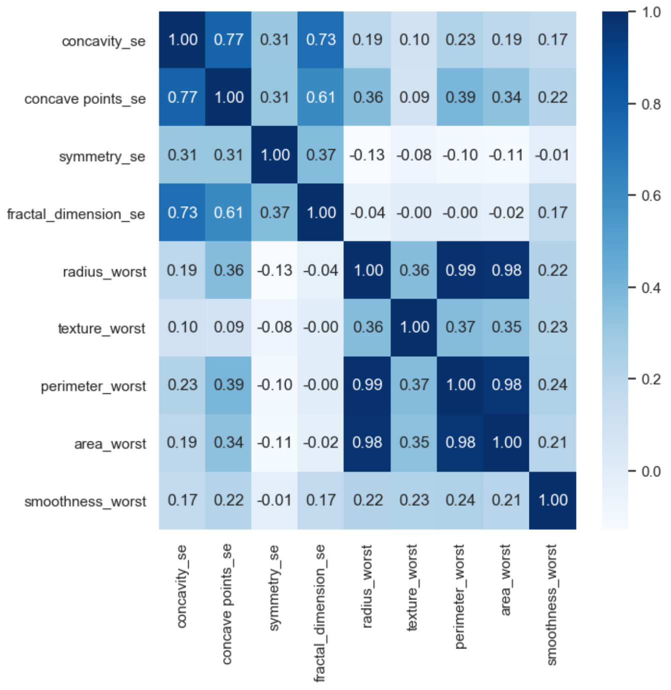


Figure 4.14: Correlation heatmap of BCW database for other 9 features

Accuracy Level:

The GP2 model achieves the highest values of 0.95, proving that a division zone of beta helps to find a hyperplane separating both classes. Alternatively, the GP3 model achieves the lowest of 0.85 in Test D that is maybe related to the range of the weights limited by α, making the model more restricted. Moreover, GP1 also has higher accuracy levels but is not correct to assume that the classifier is better than another without observing other indicators.

Error Rate:

GP1 applied to Test C, and GP2 applied to Tests C and D obtained the lowest error rates, which is reasonable since both models have the highest accuracy level. However, GP3 has the highest error rates that correspond to its lower accuracy level explained above. Although it cannot be detailed in which class was most unclassified, in general, the classification of both classes has a lower level of error rate.

Sensitivity and Specificity:

These indicators are rates that represent the portion of recognizing benign (sensitivity) and malignant (specificity) tumors, i.e., the number of observations correctly classified concerning their class. Thereby, the highest value of sensitivity was obtained by GP3 applied to Test D that indicates all the benign cases were correctly classified but comparing to the specificity rate, it is clear that a class of benign tumors were unclassified too. It would then be most accurate to look

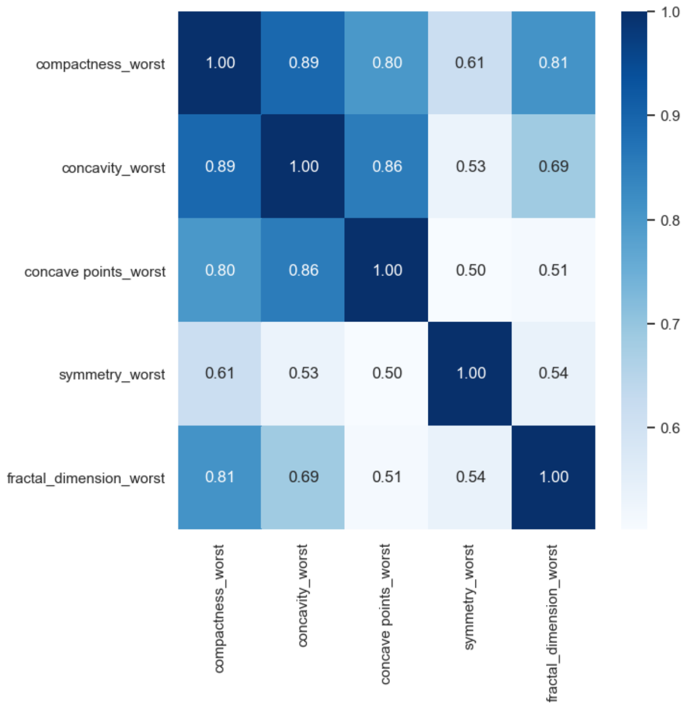


Figure 4.15: Correlation heatmap of the BCW database for 5 features

together at sensitivity and specificity to analyze each model's performance.

As the main goal is to make a precise classification for both types of tumors, the GP2 model achieved the highest sensitivity and specificity rates, indicating that the proportion of exact classification with the original labels for benign and malign tumors is higher. Although the GP1 model also has an accurate performance, its specificity rates are not promising, and the same situation occurs with the GP3 model.

Precision and Recall:

If the accuracy rate is closer to 1, each observation of a specific class was correctly classified. However, indexes as precision and recall disclose what is the proportion of, in this case, benign tumors being identified correctly concerning the predicted results and the original labels in the test database, respectively. Therefore, it is ideal to seek models with higher values for precision and recall but with a higher accuracy rate. It is the case of the GP2 and GP3 models applied to Test C, for which the proportion of benign tumors concerning the predicted results and the original labels of class also has higher accuracy rates.

Algorithm 1: Classification algorithm based on goal programming

```
input : $D \in \mathbb{R}^{n \times m}$ (set of $n$ data points with $m$ attributes to be classified)
        $\beta$ (half size of division zone)
        $\alpha$ (computational parameter)
        $MaxTime$ (maximum time for iteration)
output: $w = \{w_0, w_1, \ldots, w_n\}$ (set of weights)
1  Separate D in two parts: training and test
2  for $i = 1, \ldots, n$ do
3      for $j = 1, \ldots, m$ do
4          if $(x_{ij})$ belong to class 1 (C1) then
5              Build constraint $x_{ij}w_i + n_i^{C1} - p_i^{C1} = w_0 + \beta$
6          else
7              Build constraint $x_{ij}w_i + n_i^{C2} - p_i^{C2} = w_0 - \beta$
8          end
9      end
10 end
11 Optimize according to minimize $n_i^{C1} + p_i^{C2}$
12 return vector w
```

Previous experimental results by including dimensional reduction method

In the correlation heatmap presented before, we note that some features have higher correlations, indicating that the complete information in the thirty features could be express by a lower number of features. A dimension reduction method, the PCA method, was applied to select the principal components (lower number of features). Figure 4.19 illustrates how the variance decreases as the number of components increases.

Notably, when the number of components achieves 4, the projected data variance attains one of its lowest values and still has the database's principal information. The three GP1, GP2, and GP3 models were applied (also with Tests C and D) to this reduced database with 569 observations with four parameters. Therefore, the database is divided into two parts: 80% for training and validation by applying the K-folds cross-validation, and the 20% remaining for testing. Figure 4.20 illustrates the results of each model in the training phase, where the blue curve is the accuracy rate, and the orange curve is the error rate.

		Predicted class	
		Benign	Malign
Actual class	Benign	True positive (TP)	False negative (FN)
	Malign	False positive (FP)	True negative (TN)

Table 4.16: Confusion matrix for the BCW database

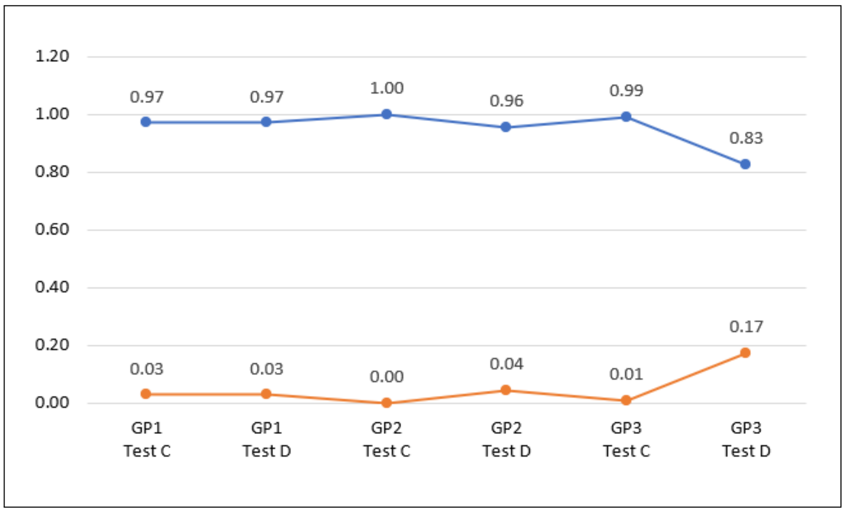


Figure 4.16: Average accuracy level and error rate for the training set

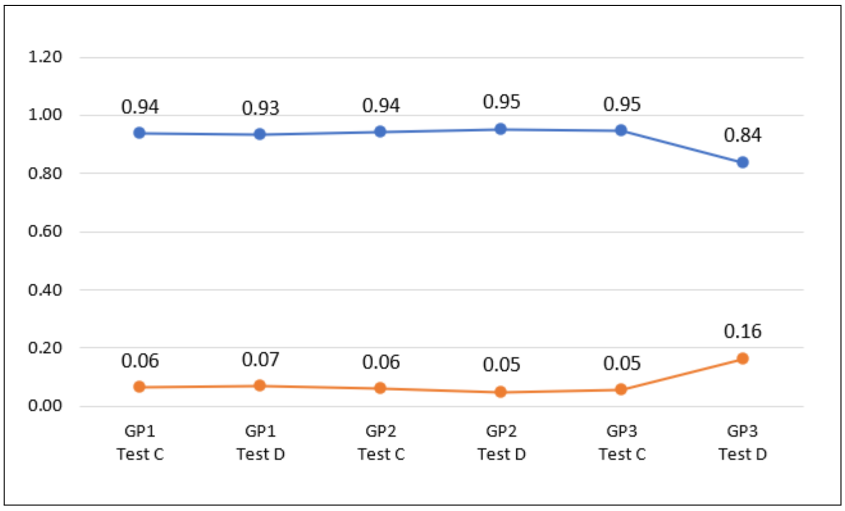


Figure 4.17: Average accuracy level and error rate for the validation set

Similar to the analysis seen previously, the accuracy rates achieve higher values since they are from the training phase, but it is clear that they are not as much as those obtained with the complete database. The GP1 and GP2 models still present the highest values, and the GP3 model the lowest that also relates to their error rate performance. However, it is more important to validate if the model does not have overfitting. Thus, the accuracy level and error rate in the *k* validation subsets are displayed in Figure 4.21.

The variation in respect to the results seen before differs in the accuracy levels and error rates for each model. The accuracy levels are higher and error rates are lower when it is only considered four parameters. This observation could indicate that no overfitting was found, and the classification improves from GP1 to GP2, but it decreases at a fewer level on GP3. However, to make a final conclusion is better to apply three classification models in a new set of data, i.e., the 20%

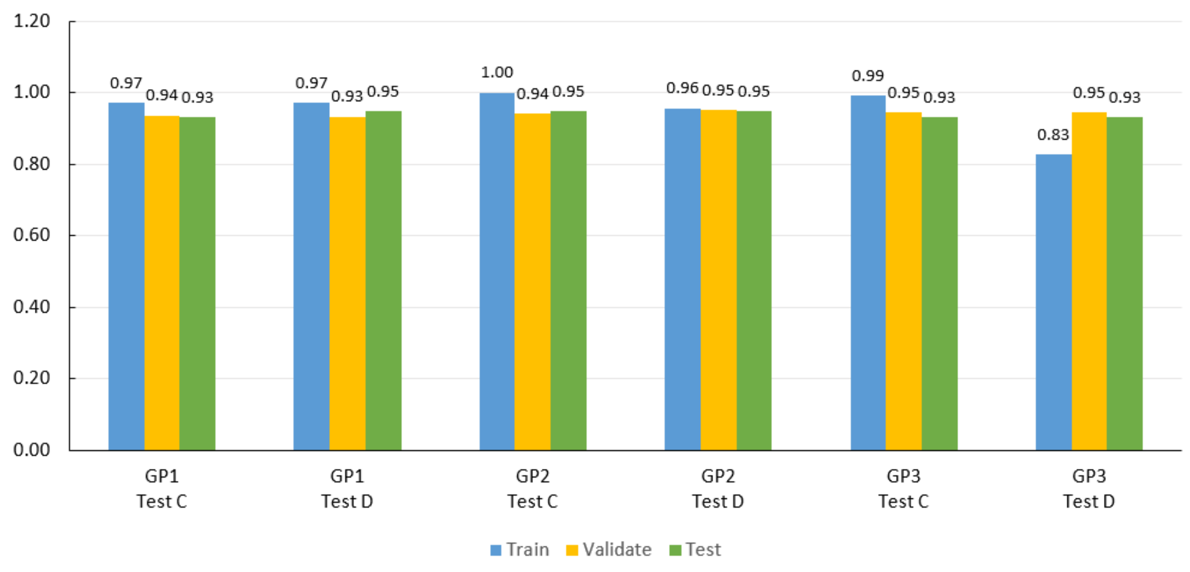


Figure 4.18: Comparative graphic of accuracy level among the training, validation and test sets

	GP 1		GP 2		GP3	
	Test C	Test D	Test C	Test D	Test C	Test D
Accuracy	0.93	0.95	0.95	0.95	0.93	0.85
Error Rate	0.07	0.05	0.05	0.05	0.07	0.15
Sensitivity	0.95	0.97	0.97	0.93	0.96	1.00
Specificity	0.90	0.91	0.91	0.98	0.89	0.63
Precision	0.95	0.94	0.94	0.98	0.93	0.80
Recall	0.95	0.97	0.97	0.93	0.96	1.00

Table 4.17: Indicator performance per model

remaining of the original database.

A comparison chart of accuracy level between the training, validation, and test subsets is displayed in Figure 4.22. It is clear that during the K-folds cross-validation, the accuracy level performance between the training and validation was improved during the testing phase. However, the accuracy level for each model decreased. This situation happened when the training and validation sets were not representative enough to make the model learns from the data and finds a hyperplane that separates both classes. Table 4.17 presents six indicators to analyze the performance of GP1, GP2, and GP3 for Tests C and D. Next, we summarize the obtained results for each indicator of measure.

Accuracy Level:

The highest accuracy levels came from the GP2 model applied to Test D and GP3 model to both Tests. Although the GP1 and GP2 models' results are not as high as the previous analysis

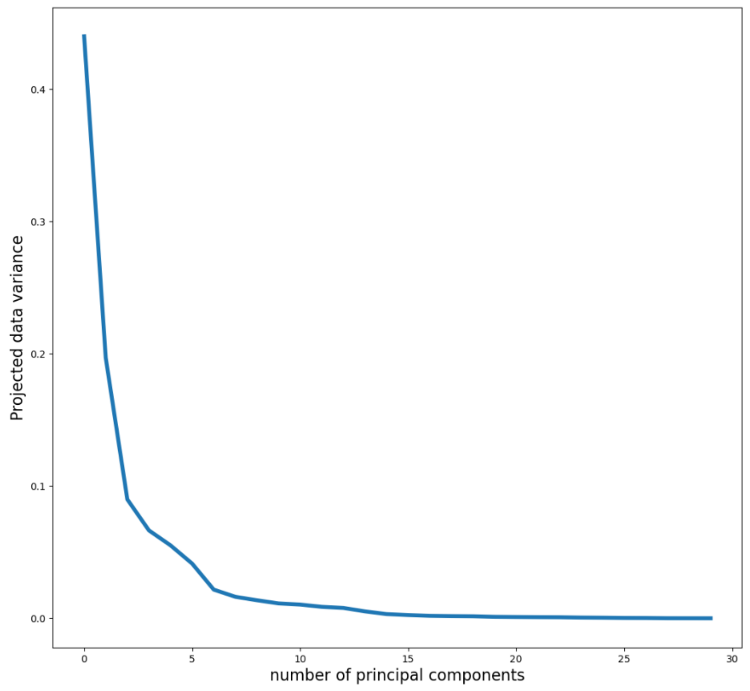


Figure 4.19: Projected data variance versus number of principal components for the PCA method

	GP 1		GP 2		GP3	
	Test C	Test D	Test C	Test D	Test C	Test D
Accuracy	0.84	0.84	0.84	0.87	0.85	0.85
Error Rate	0.16	0.16	0.16	0.13	0.15	0.15
Sensitivity	0.84	0.84	0.84	0.84	0.90	0.90
Specificity	0.85	0.85	0.85	0.91	0.78	0.78
Precision	0.89	0.89	0.89	0.93	0.86	0.86
Recall	0.84	0.84	0.84	0.84	0.90	0.90

Table 4.18: Indicator performance per model obtained from principal components

considering all data, all models achieved an accurate classification but including some unclassified observations.

Error Rate:

In contrast to the accuracy level results presented above, the error rate for the GP2 model in Test D is lower than the GP3 model. Notably, the values are higher than in Table 4.17, since less information is available after the dimension reduction, some observations are not classified correctly. However, it is not honest to speculate with this indicator how many observations from a class of tumors are unclassified than another since originally exists a meaningful difference between benign and malign tumor cases.

Sensitivity and Specificity:

Sensitivity in GP3 achieves the highest values confirming correct classification for a proportion

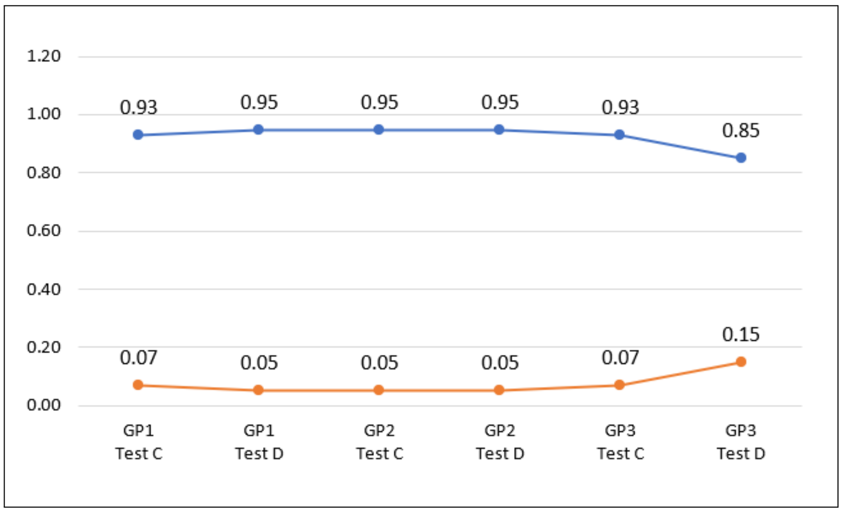


Figure 4.20: Average accuracy level and error rate for the training set obtained from principal components

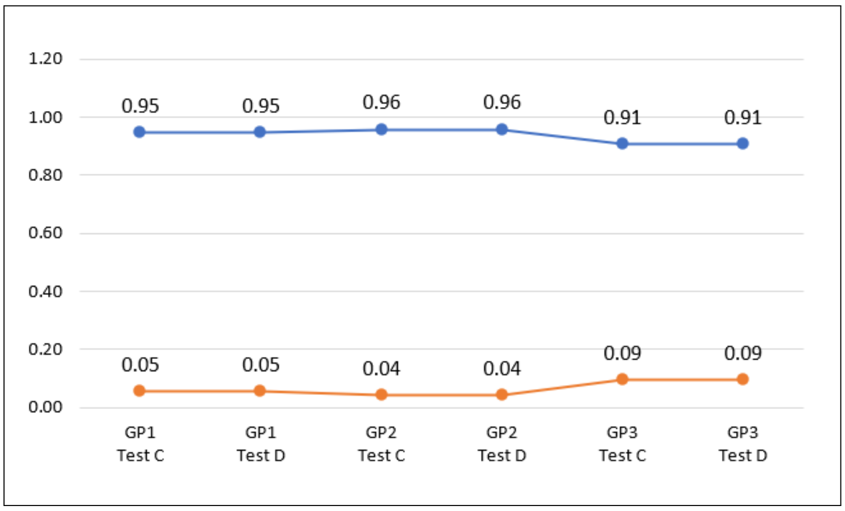


Figure 4.21: Average accuracy level and error rate for the validation set obtained from principal components

of 0.90 of benign tumors. Nevertheless, the specificity level in GP3 was the lowest as it only accurately classifies the 0.78 of malign tumors whose classification was accurate. The results may relate to the weighting method since it only made a ranking of importance but not defines how relevant is one penalization from another.

Precision and Recall:

Precision and recall values also decrease concerning the previous indicators. The high precision value was from the GP2 model for Test C, saying that the rate of benign cases accurately classifies to all the predicted observations is 0.93. However, comparing it with recall, only the 0.84 of benign instances were correctly classified, making the model less accurate due to a particular class of observations that are considered malignant when they are not. This same analysis leads us to

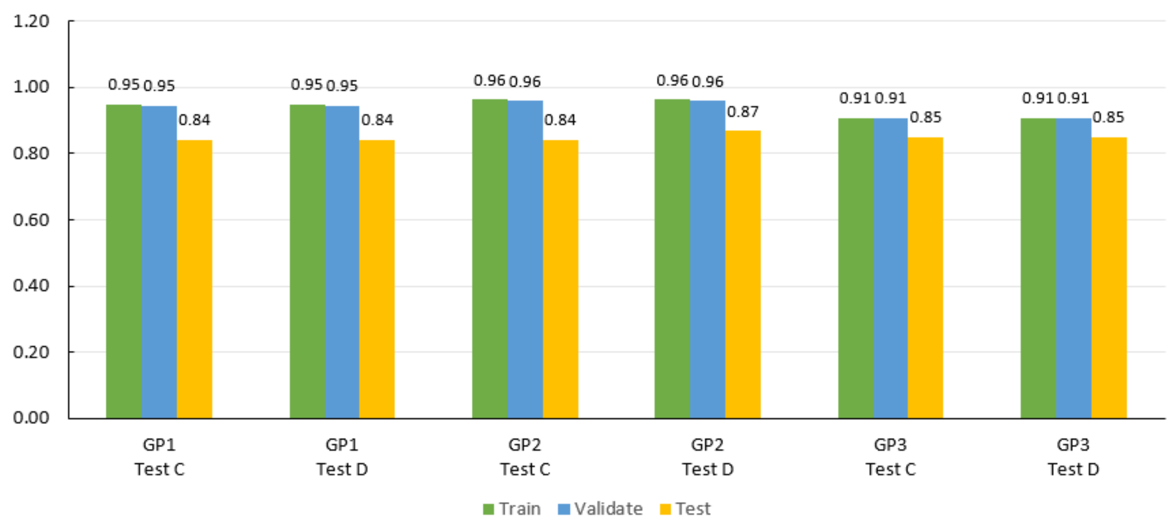


Figure 4.22: Comparative graphic of accuracy level for the training, validation, and test sets obtained from principal components

prefer a model whose precision and recall have acceptable values since it could be dangerous to classify a benign tumor as malign. The best performance was from the GP3 model as it returns a precision of 0.86 on classifying the predicted benign tumors and correctly labeling 0.90 of them.

4.2 Numerical tests for clustering

This section analyzes different tests for the distance measures for the clustering model based on integer mathematical programming. Section 3.2 describes this model. We use two databases of bovine animals for a farming company. The first one has 102 observations, and the second has 526 observations for which we consider a sparse reduction of data.

4.2.1 Setting the parameters

The integer clustering model (3.22)–(3.26) of Section 3.2 is related to the k-means clustering problem, which uses object assignation with its respective cluster center through minimization of distances between objects. This subsection presents preliminary numerical tests for setting the type of distance that better adjust to bovine animals' data in a farming company. Since the model presents an objective function attempting to minimize the sum of the distance between each object i and its assigned cluster center j, it is essential to analyze which type of distance gets better consistency in each cluster. Euclidean ($\rho = 2$), Manhattan ($\rho = 1$), and Chebyshev ($\rho = \infty$)

distances are the chosen metrics to evaluate the studied clustering model.

The parameter analysis works with 102 observations from bovine animals with four characteristics: Breed type, ECC, Frame, and Weight. The first has two types of breed cattle called: CA and NEL; the second is a body condition score (BCS or ECC as its initials in Portuguese) that describes the nutritional value through visual validation and has a scale from 1 to 9; the third is a numerical skeletal description called "Frame" which has a range from 1 to 9; and the fourth measures the total weight of each beef in Kg. Data points are plot in Figure 4.23, considering just the last three features. And given that only one feature is qualitative, the following Table 4.19 has the statistical description of the last three features that includes: mean, standard deviation (std), minimum value (min), percentile values (25%, 50%, and 75%), and maximum value (max).

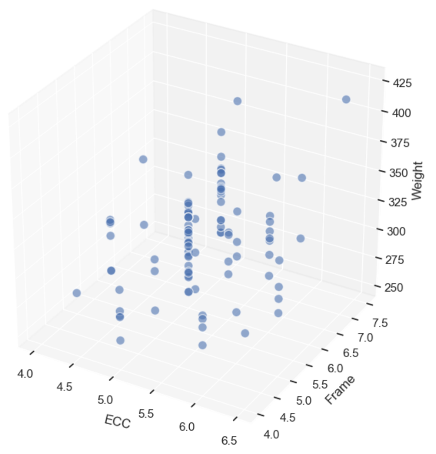


Figure 4.23: Three-dimensional plot of the bovine animal database

	ECC	Frame	Weight
mean	5.21	5.92	307.96
std	0.61	0.91	28.93
min	4	4	253
25%	5	6	290
50%	5	6	307.5
75%	5.5	6.75	324.75
max	6.5	7.5	423

Table 4.19: Descriptive statistic of the bovine animal database with 102 observations

From the standard deviation score, data points are more disperse with respect to the Weight characteristic than the other features; and the first, second and third quartile (25%, 50%, and 75%) does not have a huge difference for ECC and Frame features, unlike the Weight score. To find possible relations between features, Figure 4.24 shown a pairwise plot for both breed categories. Here, it indicates that neither feature has any linear relationship with the others so that no correlation may exist.

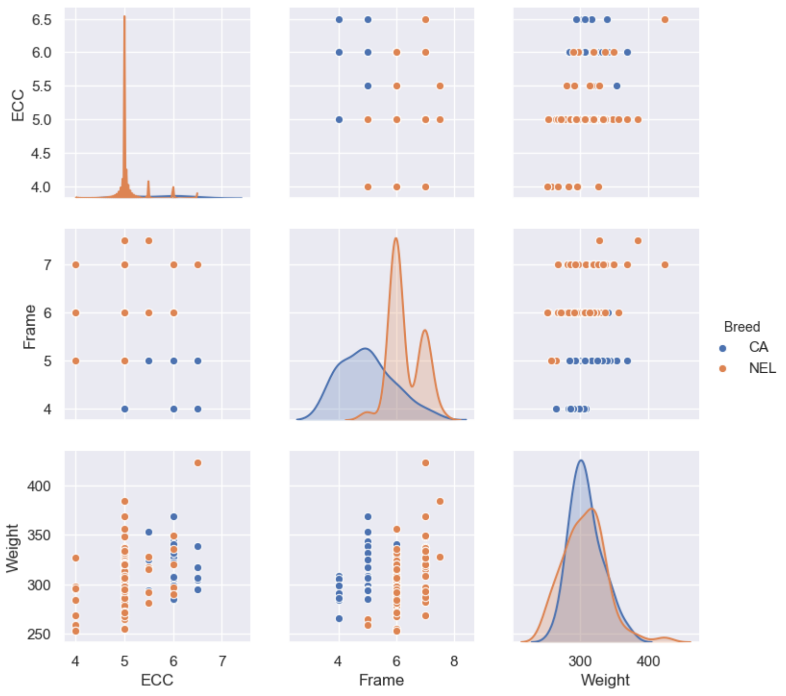


Figure 4.24: Pairwise plot of bovine animal database

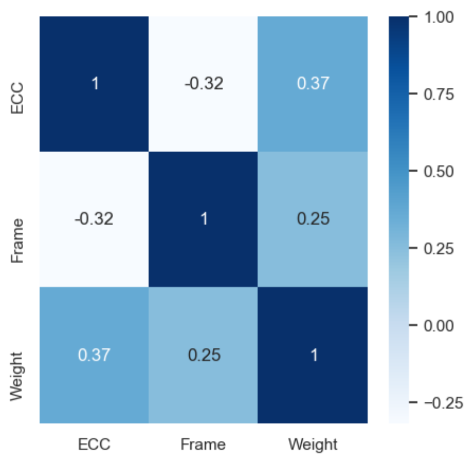


Figure 4.25: Correlation heatmap of the bovine animal database with 102 observations

A correlation heatmap in Figure 4.25 shows that while the color gets more obscure, a higher the correlation magnitude exists between two features. Moreover, since the maximum value of correlation is 0.37 between ECC and Weight, no correlation exists between the variables.

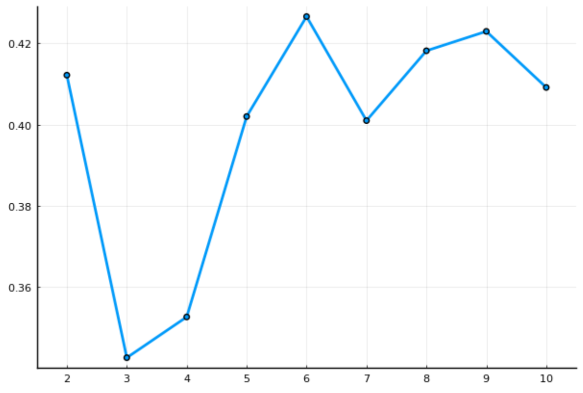
The previous statistical analysis allows understanding if some variables are highly correlated that may influence badly in models' development. However, as is seen in Figures 4.24 and 4.25, each feature gives independent information, thus the model presented in (3.22)–(3.26) can be used correctly changing the number of clusters k. In this subsection, the impact of changing k from 2 to 10 is analyzed in three different types of distances. The applied algorithm follows in the pseudocode described in Algorithm 2.

Algorithm 2: Clustering algorithm based on mathematical programming

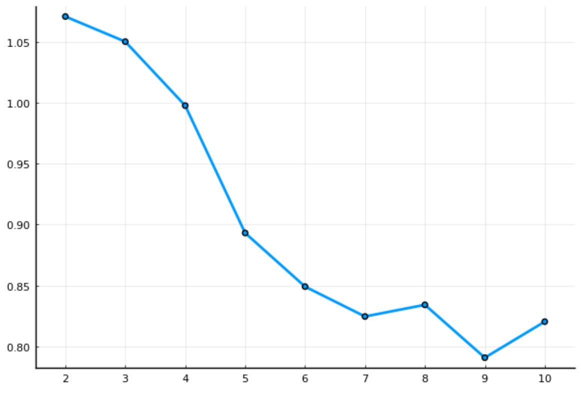
```
input : $D = \{x_1, \dots, x_n\}$ (set of data points to be clustered)
        $k$ (number of clusters)
        $MaxTime$ (maximum time for iteration)
output: $C = \{c_1, \dots, c_k\}$ (set of cluster centers)
        Label of points assign to one cluster $c_k$
1  Standardize D with mean = 0 and standard deviation = 1
2  Compute chosen distance between each point of D called $\delta_{ij}$
3  while $MaxTime$ do
4      for $i = 1, \dots, n$ do
5          for $j = 1, \dots, n$ do
6              calculate $\sum_{i=1}^{n} \sum_{j=1}^{n} \delta_{ij} z_{ij}$
7              if $\sum_{i=1}^{n} z_{ij} = 1$ and $\sum_{j=1}^{n} z_{ij} \leq n y_i$ and $\sum_{i=1}^{n} y_i = k$ then
8                  Find optimal minimum
9              end
10         end
11     end
12 end
13 return vector $C = \{c_1, \dots, c_k\}$
```

We run the clustering model for each Euclidean, Manhattan, and Chebyshev distance metrics and varying the number of clusters in the solution from parameter k. Furthermore, three evaluation scores, Silhouettes (SH), Davies Bouldin (DB), and Calinski-Harabasz (CH), are used together to evaluate if exists an optimal value for the parameter k. The SH score evaluates the clustering model's quality since it calculates how well each observation belongs to its respective cluster, comparing it to the other clusters. The DB score seeks the model with lower separation within the cluster and a higher distance between other clusters. The CH score, or the variance ratio criteria, measures how large the distances are within the cluster and the proximity of intra-cluster distances. Figure 4.26 displays the three graphics of Silhouette, Davies Bouldin, and Calinski-Harabasz score for the clustering model using Euclidean distance.

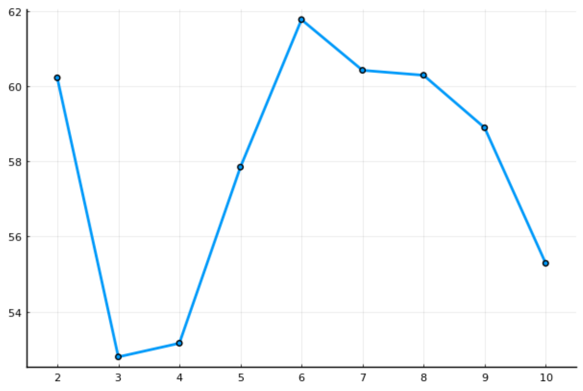
The SH score in Figure 4.26(a) represents the mean value between all the SH measures for each observation i. This metric has a range between -1 and 1. If the score is closer to -1, it suggests that the observations do not rely well upon its cluster. Otherwise, if the score is closer to 1, it indicates the cluster model can group the observations with higher similarity. Using the Euclidean distance, three points reached the highest values. However, it can only be considered the second higher SH score, i.e., when $k = 6$.



(a) Silhouette score



(b) Davies Bouldin score



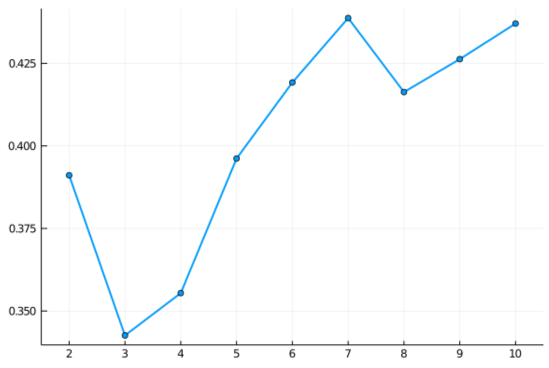
(c) Calinski-Harabasz score

Figure 4.26: Evaluation scores for the Euclidean distance metric

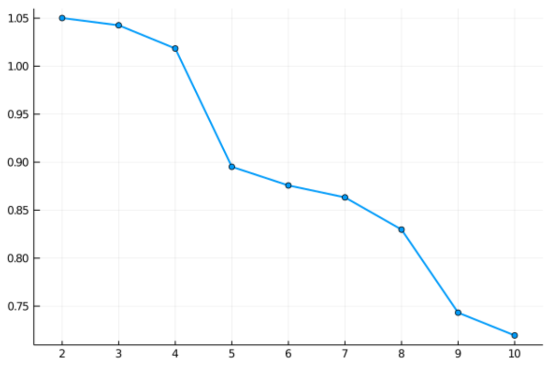
The DB score represents the variance of points in its respective cluster and how far are the clusters relative to each other. Although this measure does not have a range of values, the lower the score is, the better is the clusters. This measure complements the SH score to choose the number of k clusters. Figure 4.26(b) has two inflection points. The first when k is between 6 and 7 and the second when k is equal to 9.

Furthermore, the CH score represents the dispersion between the k clusters and the inter-cluster dispersion for all k clusters. When the clusters are more dense and well separated, then the score is higher. Figure 4.26(c) shows the results of the CH score for k different clusters, and the largest value is equal to $k = 6$. Therefore, the three scores coincide in the optimal number of clusters using the Euclidean distance, i.e., for $k = 6$.

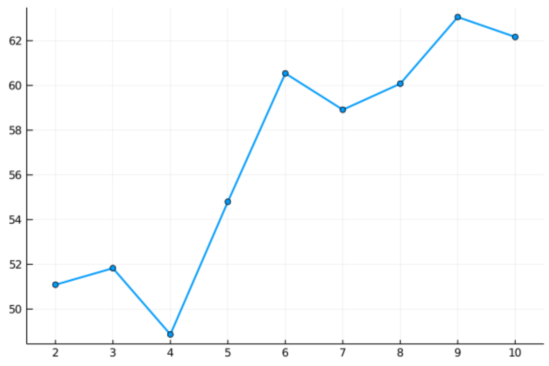
Similarly to the previous analysis, the results obtained from the clustering model in (3.22)–(3.26) using the Manhattan distance are presented for k different clusters in Figure 4.27. The SH score in Figure 4.27(a) suggests the clusters achieve higher similarity when its score is at $k = 7$. Otherwise, the DB score in Figure 4.27(b) has two inflection points: when k is between 5 and



(a) Silhouette score



(b) Davies Bouldin score



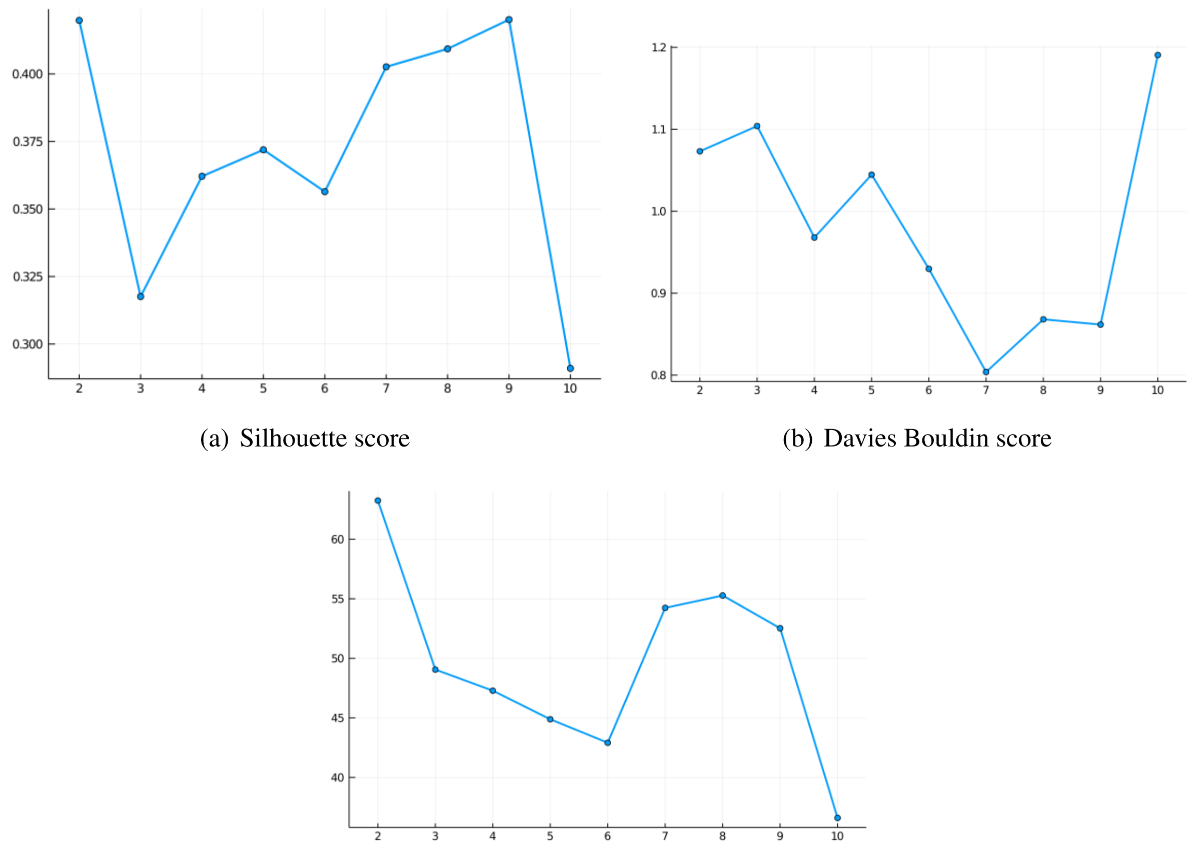
(c) Calinski-Harabasz score

Figure 4.27: Evaluation scores for the Manhattan distance metric

6, and when $k = 9$. Finally, the CH score in Figure 4.27(c) displays two solutions for when the clusters are more dense and well separated. The first at $k = 6$, and the second at $k = 9$. Since no coincidence exists between the three scores under the optimal number of clusters, the Manhattan distance does not is an adequate distance metric for the bovine animal database.

The last distance metric called Chebyshev displays its results for k clusters in Figure 4.28. The SH score indicates that when $k = 7$, the clustering model arranges the observations with higher similarity in 7 clusters. Furthermore, the DB score also coincides with the SH score since the lowest value presented in Figure 4.28(b) is achieved when $k = 7$. Therefore, the model also assembles clusters that better separate between each other. The last CH score achieves its higher values when k is between 7 and 9. Thus, in this range, the model can make clusters with higher density and well separated.

To conclude, the model (3.22)–(3.26) used three types of distance, Euclidean, Manhattan, and Chebyshev for different values of k. Moreover, the Silhouette, Davies Bouldin, and Calinski-Harabasz scores measure the model performance to find the ideal value of k, i.e., the number of clusters. Thus, given the bovine animal clustering data structure, the euclidean distance gives the



(a) Silhouette score (b) Davies Bouldin score

(c) Calinski-Harabasz score

Figure 4.28: Evaluation scores for the Chebyshev distance metric

same value of k in the three evaluation scores SH, DB, and CH. Since the clustering model based on integer mathematical programming is object-oriented, the number of variables and constraints increases according to the database's growing. This relation causes the model to spend more time to obtain an optimal solution. The next subsection analyzes another bovine animal database containing 526 observations, for which we before applied a sparse dimension reduction of data.

4.2.2 Case of study: Bovine animals database

This subsection studies a new database of bovine animals containing 526 observations for a farming company whose objective is to obtain groups of their animals with the most homogeneous characteristics per cluster. Thus, we use the previous distance parameter setting in the model (3.27)–(3.31) presented in Section 3.2 with a sparse dimension reduction. the database three characteristics: ECC, Frame, and Weight. The first is a body condition score (BCS or ECC as its initials in Portuguese) that describes the nutritional value through visual validation and is a scale from 1 to 9. The third is a numerical skeletal description called Frame, which has a range from 1 to 9. And

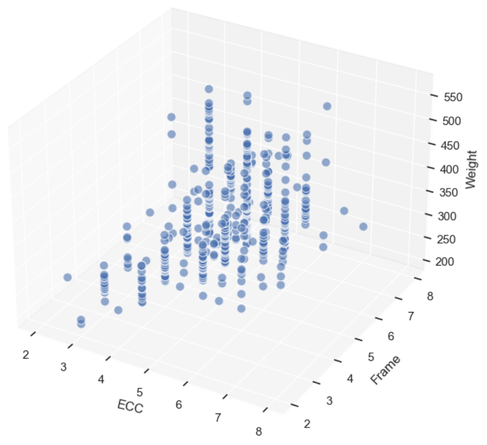
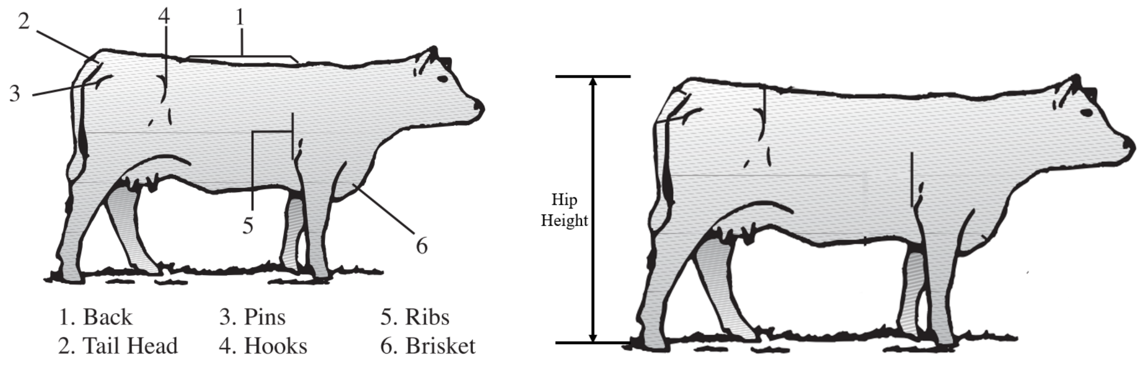


Figure 4.29: Three-dimensional plot of the bovine animal database

the fourth measure the total weight of each beef in Kg.

Figure 4.29 shows a three dimensional representation of the database characteristics. ECC suggests the relative fatness of the cow's body composition, for which the value 1 associates with a thin body and the value 9 with an extreme fatness body. This score is an excellent indicator of the nutritional status in beef cows (Eversole et al. 2005), which will lead to establishing a new nutrition plan for each cow or group of cows to achieved optimal values. The ECC score is based on six key areas for evaluation: backbone, tail head, pins, hooks, ribs, and brisket (Figure 4.30(a))

Similarly, Frame score describes the skeletal size of the cattle (McDonald 1982) based on the hip height (Figure 4.30(b)), were 1 is the smallest, and 9 represents the largest cow size. As our problem is to group the cattle by similarity, this score will indicate which one has an optimal weight



(a) Visual areas used to determine ECC in beef cows (Eversole et al. 2005)
(b) Proper position for correctly hip height measure (Eversole et al. 2005)

Figure 4.30: Specific references of a cattle to localize the ECC and Frame scores

to be a slaughter. Prediction of the animal maturity also can be made by seeking an optimal weight that matches the Frame value considering that this score usually does not change in time.

	ECC	Frame	Weight
mean	4.74	5.46	327.41
std	0.89	1.13	66.61
min	2	2	205
25%	4	5	285
50%	5	6	311.5
75%	5	6	344
max	8	8	566

Table 4.20: Descriptive statistics of bovine animal database with 526 observations

Table 4.20 presents the statistical description of the three features which includes: mean, standard deviation (std), minimum value (min), percentile values (25%, 50%, and 75%), and maximum value (max). From the standard deviation score, data points are more disperse with respect to the Weight characteristic than the other features; and the first, second and third quartile (25%, 50%, and 75%) does not have a huge difference for ECC and frame features, unlike the Weight score

A correlation heatmap in Figure 4.31 illustrates that while the color gets more obscure, a higher correlation magnitude exists between two features. Moreover, since the maximum value of correlation is equal to 0.4 between ECC and frame, no correlation exists between the variables.

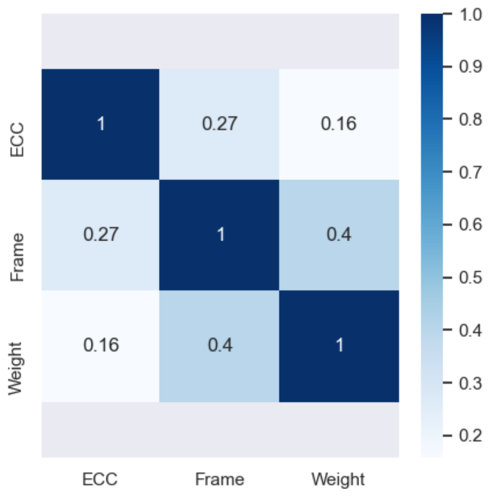


Figure 4.31: Correlation heatmap of the bovine animal database with 526 observations

The previous statistical analysis looked for a presumed linear relationship between the characteristics that could impact the algorithm development. Although each feature is independent,

	SH	DB	CH	time(s)
Test 1	0.308	0.997	65.844	69.86
Test 2	0.293	1.045	106.346	194.67
Test 3	0.305	1.059	134.845	300.90

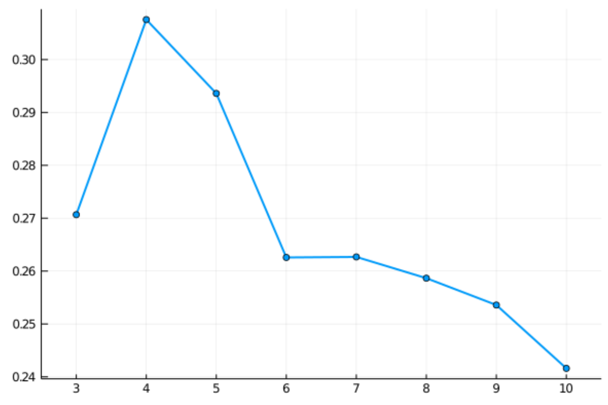
Table 4.21: Evaluation scores and running times

the size of the database does not allow to apply the model directly since it is computational more expensive. Gnagi & Baumann (2017) presented a scaling approach to reduce the database size by identifying a set of representatives instead of the original objects. First, the 526 observations are standardized with a mean equal to zero and a standard deviation equal to one. Then, the range of each p feature (ECC, Frame, and Weight) is subdivided into g intervals of equal length. Thus the p-dimensional space has a partition of g^p blocks. Each observation is assigned to a single block based on its p feature values, and the number of non-empty blocks is denoted as q. Finally, it selects a single representative observation for each q block, calculating the gravity center of corresponding observations within each block.

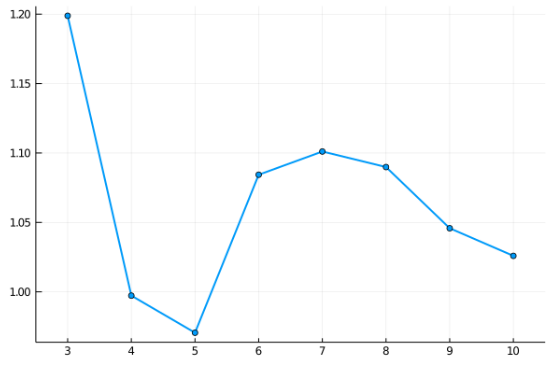
The model (3.27)–(3.31) is set up with the set of representatives to find the cluster centers, and later the same clustering membership is passed to the original observations. The cluster assignation now depends on the number of g divisions since if the number of representatives increases, each cluster has more similar observations. However, it also enlarges the time spend to solve the model. Three scenarios for $g = 10, 20, 30$ called Test 1, Test 2, and Test 3, respectively, evaluate the clustering model performance with different values of k. Figure 4.32 illustrates the graphic results for Test 1 in the Silhouette (SH), David Bounie (DB), and Calinski Harabasz (CH) scores.

Figure 4.32(a) shows that at $k = 4$, the model assembles clusters with more similar observations. Similarly, the DB score has its lowest values for k between 4 and 5; hence the observations per cluster have low variance. The CH score also reaches its highest value at $k = 4$ (Figure 4.32(c)) when clusters are more dense and well separated. The results analysis for Tests 2 and 3 also coincides with $k = 4$ as the optimal number of clusters. However, the three scenarios differ in the number of the representative points and the time spent to solve the clustering model. Table 4.21 summarizes the three evaluation scores and the time (in seconds) to solve the model with $k = 4$ for Test 1, 2, and 3.

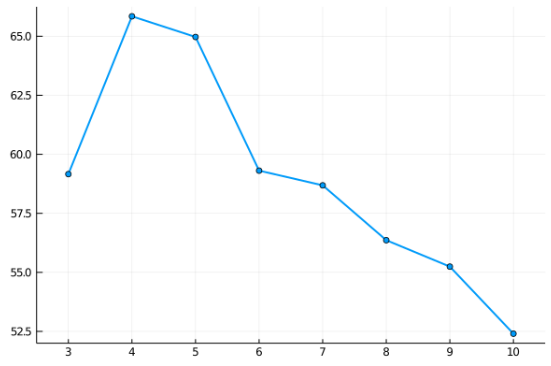
The time spent on solving the clustering model increase from Test 1 to Test 2 because the number of subdivisions g also increasing from 10 to 30. The SH score reflects how similar are the



(a) Silhouette score



(b) Davies Bouldin score



(c) Calinski-Harabasz score

Figure 4.32: Evaluation score for the Euclidean distance metric in Test 1

points inside every k cluster as it gets closer to 1. From Test 1 to 3, the SH score improves since the number of representatives increases, and more observations are assigned to its respective cluster center. On the other hand, the DB score does not reduce its value, but the difference between Test 1, 2, or 3 scores is not as large as to be concerned. Lastly, when $k = 4$, the CH score expresses that each cluster is denser while being well separated from the others.

	Test 1			Test 2			Test 3		
	ECC	Frame	Weight	ECC	Frame	Weight	ECC	Frame	Weight
C1	4.05	4.08	282.42	4.02	3.96	277.54	3.97	3.85	271.69
C2	4.65	6.35	307.41	4.64	6.15	306.35	4.64	6.13	309.16
C3	4.83	6.05	441.38	4.83	6.12	442.25	4.77	6.04	449.34
C4	5.72	5.17	311.80	5.84	5.16	317.47	5.84	5.13	317.77

Table 4.22: Average ECC, Frame and Weight in each cluster

Table 4.22 summarizes the average values of each characteristic (ECC, Frame, and Weight) in all 4 clusters for the three tests. The clusters (C1, C2, C3, and C4) have a similar value per feature in Test 1, 2, and 3. It explains that the number of subdivisions g does not have a strong influence on the clusters' structure, and it only differs on the amount of time the model spends finding an optimal solution.

This application aims to improve the farming company's sales by using a suitable number of clusters to split the animals into similar groups. Therefore, it is easier to identify which group holds the best characteristics to be sold first at a better price. According to Eversole et al. (2005), the optimal ECC value is when it ranges between 5 and 7 because the cattle cow has an excellent overall appearance. When ECC is lower than 5, the cow loses more muscle tone since less fat is available to supply energy to sustain vital bodily functions (Gadberry 2012).

From the results in Table 4.22, clusters C1, C2, and C3 have lower values than 5, being C1 the lowest. In C1, ECC ranges between 3.97 and 4.05, implying some cows inside the cluster need a change in their nutrition plan to reach the ideal amount of fat at the moment of slaughter. Otherwise, clusters C2 and C3 do not attain the optimal measure, then is suggested to wait for the cow to gain more fat to improve its ECC. McDonald (1982) indicates that to be considered as a potential market sell, the range of weight cattle should have is related to its Frame value (Table 4.23). However, unlike the ECC score, a larger Frame score does not represent a better cow, but it links to how long the cow will achieve maturity.

Frame	Weight (Kg)
1-2	150 - 180
3-5	200 - 350
6-9	350 - 450

Table 4.23: Optimal cattle weight according to the Frame score

Analyzing the optimal cattle weight and frame to the clusters results in Table 4.22, we identify that for cluster C2 its average frame value between 6.13 and 6.35 does not achieve the minimum weight of 350 Kg. Therefore, the cattle in this cluster will take more time to gain weight to get their corresponding maturity. However, this can be accelerated by changing the nutrition plan, or the cattle's purpose can be altered, for example, to use it for reproduction.

CHAPTER 5

Final remarks and some clues for future researches

5.1 Conclusions

This research examined mathematical optimization models that are useful to an enormous number of computer science applications, and jointly with machine learning are two growing fields of artificial intelligence. Each studied model has a machine learning background, and they are techniques well-known in this area of knowledge. Moreover, the recent mathematical programs and optimization techniques for machine learning have started considerable growth, such that both approaches are performed to improve each other.

The classification model based on goal programming seeks to find a hyperplane that separates two classes of data. It uses the margin concept, introduced for the Support Vector Machine, as the smallest distance between the separating hyperplane and any observation. The clustering model based on integer mathematical programming is a different overview of the k-means clustering problem. It groups the most similar observations into a fixed number of clusters by assigning an inner specific object as a cluster center for a particular set of objects. The objective is to minimize the distance between the cluster center and its designated objects.

Both the classification and the clustering models depend on a suitable parameter setting to achieve the correct optimal solution. Preliminary numerical experiments for both models helped understand their behavior according to the data and setting the parameters. For example, in classification models, we conducted tests for the GP1, GP2, and GP3 models for setting the parameters β and α related to the data variance. Thus, we evaluated the influence of the distance metrics used in the clustering model while minimizing the ratio between the cluster center and its assigned objects.

Moreover, if the database's information increases, both models need more time to achieve an optimal solution. Thus, it is essential to implement dimension reduction methods to reduce the number of features or observations. For the classification model, the PCA dimension reduction method found the principal components that decrease the original features while keeping as most information as possible. Similarly, for the clustering model, a technique based on sparse-reduced computation identified the representative points to decrease the number of original observations. Thus the centers of the clusters are the same for the reduced and original data.

In this dissertation, we addressed a case study for classifying cancer breast tumors between benign and malign that uses three classification models based on goal programming on a database containing 569 observations with thirty features. Additionally, we performed a case study on bovine animals the obtains the optimal number of clusters using a database containing 526 objects with three characteristics. The following subsection summarizes the results obtained from both case studies.

5.2 Summary of the obtained results

The case studies lead to obtaining accurate results that were validated with different metric indicators. The first case involved classifying the database Breast Cancer Wisconsin (569 observations and 30 features) with two breast cancer labels, where we used non-standard preferences for penalizing incorrect classification. The second case involved clustering a bovine database (526 observations and four features), where we used different distance metrics to obtain similar groups of animals.

The classification models used in the breast cancer tumor problem achieved higher accuracy rates and lower error rates considering both the original database and the reduced database from the principal component analysis method. The success in this case study confirms that the structure of data follows the numerical results from setting the parameters α and β made in Section 4.1. However, by adding the preferences and internal weights in the GP3 model does not improve the classification. Therefore, the weighting method may not be very accurate since we only rank the criteria but do not give a scale of importance.

The results of a clustering model for the bovine database allow seeking the best assignation of objects into k clusters. Since the objective function is a based-distance metric that minimizes the distance between each observation and its cluster center, the numerical experiments showed that Euclidean distance fits the used database. Nevertheless, it is computationally more expensive, for the 526 observations, to solve the model since it is object-oriented. Thus, we applied a sparse reduction method that selects the representative objects within the original database to solve the clustering model using only these representative objects. Note that the representative objects save the information of cluster memberships related to the 526 original observations.

5.3 Future researches

This dissertation aimed to use optimization models to achieve solutions in classification and clustering approaches. We observed that these studied models together with recent optimizations methods could successfully assist to improve machine learning techniques for classification and clustering.

We note that there are still points of improvement to be discussed in future researches. For example, in the classification model named GP3, the internal penalty weights did not improve the accuracy level when classifying breast cancer tumors. This weakness may be due to the ranking weights method used, which only considers the order of importance among them but not how important each weight is compared to each other. For future research, we plan to seek new assignment criteria methodologies to make the classification more accurate.

Although the practical applications with both studied models achieve outstanding results, they only apply to linear problems. Therefore, it would be interesting to analyze and compare other machine learning techniques with mathematical programming approaches to solve nonlinear problems such as neural networks.

Bibliography

Alpaydin, E. (2010), *Introduction to Machine Learning*, 2nd edn, The MIT Press.

Aragon-Gómez, R. & Clempner, J. B. (2020), 'Traffic-signal control reinforcement learning approach for continuous-time markov games', *Engineering Applications of Artificial Intelligence* **89**, 103415.

Baesens, B., Setiono, R., Mues, C. & Vanthienen, J. (2003), 'Using neural network rule extraction and decision tables for credit-risk evaluation', *Manage. Sci.* **49**(3), 312–329.

Berkhin, P. (2006), A survey of clustering data mining techniques, *in* 'Grouping multidimensional data', Springer, pp. 25–71.

Bishop, C. M. (2006), *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer-Verlag.

Bottou, L., Curtis, F. E. & Nocedal, J. (2016), 'Optimization methods for large-scale machine learning (2016)'.

Carrizosa, E. & Romero Morales, D. (2013), 'Supervised classification and mathematical optimization', *Comput. Oper. Res.* **40**(1), 150–165.

Chang, C. T. (2008), 'Revised multi-choice goal programming', *Applied mathematical modelling* **32**(12), 2587–2595.

Charnes, A. & Cooper, W. W. (1957), ‘Management models and industrial applications of linear programming’, *Management Science* **4**(1), 38–91.

Charnes, A., Cooper, W. W. & Ferguson, R. O. (1955), ‘Optimal estimation of executive compensation by linear programming’, *Management Science* **1**(2), 138–151.

Cintolo-Gonzalez, J. A., Braun, D., Blackford, A. L., Mazzola, E., Acar, A., Plichta, J. K., Griffin, M. & Hughes, K. S. (2017), ‘Breast cancer risk models: a comprehensive overview of existing models, validation, and clinical applications’, *Breast cancer research and treatment* **164**(2), 263–284.

Cortes, C. & Vapnik, V. (1995), ‘Support-vector networks’, *Machine learning* **20**(3), 273–297.

Danielson, M. & Ekenberg, L. (2014), Rank ordering methods for multi-criteria decisions, *in* ‘Joint International Conference on Group Decision and Negotiation’, Springer, pp. 128–135.

Danielson, M. & Ekenberg, L. (2017), ‘A robustness study of state-of-the-art surrogate weights for MCDM’, *Group Decision and Negotiation* **26**(4), 677–691.

De Andres, R., García-Lapresta, J. L. & González-Pachón, J. (2010), ‘Performance appraisal based on distance function methods’, *European Journal of Operational Research* **207**(3), 1599–1607.

Dhanachandra, N. & Chanu, Y. J. (2017), ‘A survey on image segmentation methods using clustering techniques’, *European Journal of Engineering Research and Science* **2**(1), 15–20.

Dinov, I. D. (2018), *Data science and predictive analytics: Biomedical and health applications using R*, Springer.

Eversole, D. E., Browne, M. F., Hall, J. B. & Dietz, R. E. (2005), ‘Body condition scoring beef cows’.

Flavell, R. et al. (1976), ‘A new goal programming formulation’, *Omega* **4**(6), 731–732.

Fox, W. P. & Burks, R. (2019), *Mathematical Modeling, Management Science, and Operations Research for Military Decision-Making*, Springer International Publishing, pp. 1–15.

Freed, N. & Glover, F. (1981), ‘Simple but powerful goal programming models for discriminant problems’, *European Journal of Operational Research* **7**(1), 44–60.

Freed, N. & Glover, F. (1986), ‘Notes and communications resolving certain difficulties and improving the classification power of lp discriminant analysis formulations’, *Decision Sciences* **17**(4), 589–595.

Frey, P. W. & Slate, D. J. (1991), ‘Letter recognition using holland-style adaptive classifiers’, *Machine learning* **6**(2), 161–182.

Gadberry, S. (2012), Feeding beef cows based on body condition scores, *in* ‘MP (University of Arkansas (System). Cooperative Extension Service)’, University of Arkansas Libraries, Fayetteville.

Gambella, C., Ghaddar, B. & Naoum-Sawaya, J. (2020), ‘Optimization problems for machine learning: a survey’, *European Journal of Operational Research* .

García, F., Guijarro, F. & Moya, I. (2010), ‘A goal programming approach to estimating performance weights for ranking firms’, *Computers & Operations Research* **37**(9), 1597–1609.

Gavrishchaka, V. & Banerjee, S. (2006), ‘Support vector machine as an efficient framework for stock market volatility forecasting’, *Computational Management Science* **3**, 147–160.

Ghufran, S., Khowaja, S. & Ahsan, M. (2015), ‘Optimum multivariate stratified double sampling design: Chebyshev’s goal programming approach’, *Journal of Applied Statistics* **42**(5), 1032–1042.

Giger, M. L. (2018), ‘Machine learning in medical imaging’, *Journal of the American College of Radiology* **15**(3), 512–520.

Giménez, J. C., Bertomeu, M., Diaz-Balteiro, L. & Romero, C. (2013), ‘Optimal harvest scheduling in eucalyptus plantations under a sustainability perspective’, *Forest Ecology and Management* **291**, 367–376.

Gnagi, M. & Baumann, P. (2017), ‘Large-scale clustering using mathematical programming’, pp. 789–793.

Han, J., Pei, J. & Kamber, M. (2011), *Data mining: concepts and techniques*, Elsevier.

Ignizio, J. P. (1976), *Goal programming and extensions*, Lexington Books, Lexington, MA.

Ignizio, J. P. (1982), *Linear Programming in Single and Multiple Objective Systems*, Prentice Hall, Upper Saddle River, NJ.

Ignizio, J. P. (1985), 'An algorithm for solving the linear goal programming problem by solving its dual', *Journal of the Operational Research Society* **36**(6), 507–515.

Ignizio, J. P. (2004), 'Optimal maintenance headcount allocation: an application of chebyshev goal programming', *International journal of production research* **42**(1), 201–210.

James, G., Witten, D., Hastie, T. & Tibshirani, R. (2014), *An Introduction to Statistical Learning: With Applications in R*, Springer Publishing Company, Incorporated.

Jayaraman, R., Colapinto, C., La Torre, D. & Malik, T. (2017), 'A weighted goal programming model for planning sustainable development applied to gulf cooperation council countries', *Applied Energy* **185**, 1931–1939.

Jones, D., Collins, A. & Hand, C. (2007), 'A classification model based on goal programming with non-standard preference functions with application to the prediction of cinema-going behaviour', *European Journal of Operational Research* **177**, 515–524.

Jones, D. & Tamiz, M. (1995), 'Improving the flexibility of goal programming via preference modelling techniques.', pp. 391–399.

Jones, D. & Tamiz, M. (2010), *Practical goal programming*, Vol. 141, Springer.

K. Brian Haley B.Sc., P. a. (1967), *Mathematical Programming for Business and Industry*, Macmillan Education UK.

Kakushadze, Z. & Yu, W. (2017), 'K-means and cluster models for cancer signatures', *Biomolecular detection and quantification* **13**, 7–31.

Koolagudi, S. G., Murthy, Y. S. & Bhaskar, S. P. (2018), 'Choice of a classifier, based on properties of a dataset: case study-speech emotion recognition', *International Journal of Speech Technology* **21**(1), 167–183.

Kose, E. & Karabay, S. (2016), 'Mathematical programming model proposal to solve a real-life public sector facility location problem', *International Journal of Operational Research* **26**, 1.

Kraus, M., Feuerriegel, S. & Oztekin, A. (2020), ‘Deep learning in business analytics and operations research: Models, applications and managerial implications’, *European Journal of Operational Research* **281**(3), 628–641.

Kuhn, M. & Johnson, K. (2013), *Applied predictive modeling*, Vol. 26, Springer.

Larivière, B. & Van den Poel, D. (2005), ‘Predicting customer retention and profitability by using random forests and regression forests techniques’, *Expert Systems with Applications* **29**(2), 472–484.

Lee, S. M. (1972), *Goal programming for decision analysis*, Auerback Publishers Philadelphia.

Liu, Y. & Ge, Z. (2018), ‘Weighted random forests for fault classification in industrial processes with hierarchical clustering model selection’, *Journal of Process Control* **64**, 62–70.

Lloyd, S. (1982), ‘Least squares quantization in PCM’, *IEEE transactions on information theory* **28**(2), 129–137.

Luenberger, D. G. & Ye, Y. (2015), *Linear and Nonlinear Programming*, Springer Publishing Company, Incorporated.

Mangasarian, O., Street, N. & Wolberg, W. (1970), ‘Breast cancer diagnosis and prognosis via linear programming’, *Operations Research* **43**.

Martínez-Tenor, A., Fernández-Madrigal, J. A., Cruz-Martín, A. & González-Jiménez, J. (2018), ‘Towards a common implementation of reinforcement learning for multiple robotic tasks’, *Expert Systems with Applications* **100**, 246–259.

Mavaddat, N., Pharoah, P. D., Michailidou, K., Tyrer, J., Brook, M. N., Bolla, M. K., Wang, Q., Dennis, J., Dunning, A. M., Shah, M. et al. (2015), ‘Prediction of breast cancer risk based on profiling with common genetic variants’, *JNCI: Journal of the National Cancer Institute* **107**(5).

McDonald, A. (1982), ‘Frame scoring of beef cattle’, *Agnote-Melbourne. Department of Agriculture* .

McGregor, M. & Dent, J. (1993), ‘An application of lexicographic goal programming to resolve the allocation of water from the rakaia river (new zealand)’, *Agricultural systems* **41**(3), 349–367.

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

Nakayama, H. & Kagaku, N. (1998), 'Pattern classification by linear goal programming and its extensions', *Journal of Global Optimization* **12**(2), 111–126.

Nha, V. T., Shin, S. & Jeong, S. H. (2013), 'Lexicographical dynamic goal programming approach to a robust design optimization within the pharmaceutical environment', *European Journal of Operational Research* **229**(2), 505–517.

Ozan, T. M. (1986), *Applied Mathematical Programming for Production and Engineering Management*, Reston Publishing Co.

Pandey, T. N., Jagadev, A. K., Mohapatra, S. K. & Dehuri, S. (2017), Credit risk analysis using machine learning classifiers, *in* '2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)', IEEE, pp. 1850–1854.

Pedro Duarte Silva, A. (2017), 'Optimization approaches to supervised classification', *European Journal of Operational Research* **261**(2), 772–788.

Qureshi, M. N. & Ahamad, M. V. (2018), 'An improved method for image segmentation using k-means clustering with neutrosophic logic', *Procedia computer science* **132**, 534–540.

Rao, S. S. (2009), *Engineering Optimization: Theory and Practice: Fourth Edition*, John Wiley and Sons.

Romero, C. (1991), *Handbook of critical issues in goal programming*, Pergamon Press, Oxford.

Romero, C. (2001), 'Extended lexicographic goal programming: a unifying approach', *Omega* **29**(1), 63–71.

Rosenblatt, F. (1961), Principles of neurodynamics. perceptrons and the theory of brain mechanisms, Technical report, Cornell Aeronautical Lab Inc Buffalo NY.

Sah, S., Gaur, A. & Singh, M. P. (2018), Evaluating pattern classification techniques of neural network using k-means clustering algorithm, *in* 'Next-Generation Networks', Springer, pp. 563–588.

Sethy, H., Patel, A. & Padmanabhan, V. (2015), 'Real time strategy games: a reinforcement learning approach', *Procedia Computer Science* **54**, 257–264.

Shafigh, A. & Sourati, N. (2016), 'Proposed efficient algorithm to filter spam using machine learning techniques', *Pacific Science Review A: Natural Science and Engineering* .

Shailaja, K. & Anuradha, B. (2016), Effective face recognition using deep learning based linear discriminant classification, *in* '2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)', IEEE, pp. 1–6.

Shawky, M. S., Martin, H., Hugo, H. J., Lloyd, T., Britt, K. L., Redfern, A. & Thompson, E. W. (2017), 'Mammographic density: a potential monitoring biomarker for adjuvant and preventative breast cancer endocrine therapies', *Oncotarget* **8**(3), 5578.

Shobha, G. & Rangaswamy, S. (2018), *Computational Analysis and Understanding of Natural*, Vol. 38 of *Handbook of Statistics*, Elsevier.

Silva, A. F. & Marins, F. A. S. (2015), 'Revisão da literatura sobre modelos de programação por metas determinística e sob incerteza', *Production* **25**(1), 92–112.

Simeone, O. (2017), 'A brief introduction to machine learning for engineers', *Foundations and Trends in Signal Processing* **12**.

Song, H., Triguero, I. & Özcan, E. (2019), 'A review on the self and dual interactions between machine learning and optimisation', *Progress in Artificial Intelligence* **8**(2), 143–165.

Stillwell, W. G., Seaver, D. A. & Edwards, W. (1981), 'A comparison of weight approximation techniques in multiattribute utility decision making', *Organizational behavior and human performance* **28**(1), 62–77.

Tamiz, M. & Jones, D. F. (1997), An example of good modelling practice in goal programming: Means for overcoming incommensurability, *in* 'Advances in Multiple Objective and Goal Programming', Springer Berlin Heidelberg, pp. 29–37.

Tamiz, M., Jones, D. F. & El-Darzi, E. (1995), 'A review of goal programming and its applications', *Annals of operations Research* **58**(1), 39–53.

Theodoridis, S. (2015), *Machine Learning:A Bayesian and Optimization Perspective*, Academic Press.

Tiwari, R., Dharmar, S. & Rao, J. (1987), ‘Fuzzy goal programming – an additive model’, *Fuzzy sets and systems* **24**(1), 27–34.

Uría, V. R., Caballero, R., Ruiz, F. & Romero, C. (2002), ‘Meta-goal programming’, *European Journal of Operational Research* **136**(2), 422–429.

Watt, J., Borhani, R. & Katsaggelos, A. K. (2016), *Machine Learning Refined: Foundations, Algorithms, and Applications*, Cambridge University Press.

Wei, W., Liang, J., Guo, X., Song, P. & Sun, Y. (2019), ‘Hierarchical division clustering framework for categorical data’, *Neurocomputing* **341**, 118–134.

Wild, C., Weiderpass, E. & Stewart, B. (2020), *World Cancer Report: Cancer Research for Cancer Prevention*, International Agency for Research on Cancer.

Wolberg, W. & Mangasarian, O. (1989), ‘Breast cancer wisconsin (original) data set’, *UCI Machine Learning Repository, University of California* .

Zaki, M. J. & Meira Jr., W. (2014), *Data mining and analysis: fundamental concepts and algorithms*, Cambridge University Press.

Zhang, X., Rice, M., Tworoger, S. S., Rosner, B. A., Eliassen, A. H., Tamimi, R. M., Joshi, A. D., Lindstrom, S., Qian, J., Colditz, G. A. et al. (2018), ‘Addition of a polygenic risk score, mammographic density, and endogenous hormones to existing breast cancer risk prediction models: A nested case–control study’, *PLoS medicine* **15**(9).

Zografidou, E., Petridis, K., Arabatzis, G. & Dey, P. K. (2016), ‘Optimal design of the renewable energy map of greece using weighted goal-programming and data envelopment analysis’, *Computers & Operations Research* **66**, 313–326.