UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

PAULO RENATO CARVALHO DE AZEVEDO FILHO

# SLACGS: Simulator for Loss Analysis of Classifiers using Gaussian Samples

RIO DE JANEIRO
2024

PAULO RENATO CARVALHO DE AZEVEDO FILHO

# SLACGS: Simulator for Loss Analysis of Classifiers using Gaussian Samples

Undergraduate thesis presented to the Institute of Computing at the Federal University of Rio de Janeiro as part of the requirements for obtaining the degree of Bachelor of Science in Computer Science.

Orientador: Prof. Daniel Sadoc Menasche

RIO DE JANEIRO
2024

PAULO RENATO CARVALHO DE AZEVEDO FILHO

SLACGS: Simulator for Loss Analysis of Classifiers using Gaussian Samples

Undergraduate thesis presented to the Institute of Computing at the Federal University of Rio de Janeiro as part of the requirements for obtaining the degree of Bachelor of Science in Computer Science.

Aprovado em 19 de março de 2024

BANCA EXAMINADORA:

Prof. Daniel Sadoc Menasché
Orientador
Ph.D. (UFRJ)

Prof. João Ismael D. Pinheiro
Co-orientador
D.Sc. (UFRJ)

Prof. João Antonio Recio da Paixão
D.Sc. (UFRJ)

Prof. Pedro Henrique Cruz Caminha
D.Sc. (UFRJ)

## Resumo

In this work, we developed a simulator to explore the impact of different factors on the expected error rate in binary classification problems. The simulator allows us to analyze how variations in dataset cardinality and the number of features affect the effectiveness of a classifier. Using simulated data, generated from multivariate Gaussians, we investigated the behavior of the expected error rate as a function of dataset cardinality and the number of features.

The results obtained with the simulator reveal conditions under which an increase in dataset cardinality can help compensate for the absence of features. Our results suggest that the relationship between dataset cardinality and the number of features is complex and may not be linear, highlighting a possible trade-off between the two. The simulator developed in this study thus emerges as a crucial tool for unraveling this and other dynamics involved in binary data classification.

**keywords**: Classifiers; SVM; Bayes Error; Performance.

# LISTA DE ILUSTRAÇÕES

## SUMÁRIO

# 1 INTRODUCTION AND MOTIVATION

In this chapter, we present an introduction and overview of the sample classification problem, and motivate the need to develop a simulator to evaluate classifiers. We indicate some of the design choices considered in this work, including the focus on linear classifiers and Gaussian samples, and present our main contributions.

## 1.1 MOTIVATION: THE GENERAL CLASSIFICATION PROBLEM

Suppose that the elements of a population can be divided into groups, with each group corresponding to a label. Furthermore, some of the attributes of each element can be measured, and there is an (unknown) relationship between the attributes and the labels.

The problem of extbfclassification learning consists, in essence, of obtaining a rule that allows assigning a label to a given unlabeled observation based on its attributes. This rule is constructed from a sample of elements for which both the attributes and the labels are known. In this context, two key elements that influence the task of obtaining a good classification rule, that is, a rule that minimizes the classification error, are:

- extbfcardinality: the number of observations used to train the classification algorithm.

- extbfdimensionality: the number of attributes in each observation.

Although the two elements above have a fundamental impact on classifier performance, a quantitative assessment of their influence on classifier quality is still not fully understood.

One of our main objectives is to build a simulator to help understand the effects of different elements, such as data cardinality and dimensionality, on classification outcomes.

## 1.2 BRIEF REVIEW OF RELATED WORK

There is extensive literature on the impact of the number of observations and the number of attributes on classifier performance (ENTEZARI-MALEKI; REZAEI; MINAEI-BIDGOLI, 2009). However, most studies consider very large datasets. In this work, we focus on the case where the dataset is small, that is, with few samples (in the order of thousands of samples) and few attributes (in the order of one to three attributes), noting that the constructed simulator is generic and can be used to generate and analyze large datasets, provided there is computational power for it.

In this work, we aim to understand the joint role of the number of observations and the number of attributes in classification tasks. To this end, we built a simulator and considered one of the simplest possible scenarios, namely, focusing on the classification of observations derived from multivariate Gaussian models.

## 1.3 OBJECTIVES AND OVERVIEW

Our main objective is to evaluate the impact of different elements that influence the quality of a classifier that classifies elements between two classes.[1] To this end, we produced a versatile and extensible simulator.

The simulator allows us to analyze the influence of different parameters of interest on classification quality. The parameters of interest that we evaluate are the inputs to the simulator, namely:

- means of each class, $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, for classes 1 and 2, $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)$,
- number of observations in both considered classes, $n$, where each class has $n/2$ observations,
- number of attributes in each observation, $D$,
- covariance matrix, that is, standard deviation for each attribute, $\sigma_i$, and correlation between each pair of attributes $\rho_{i,j}$.

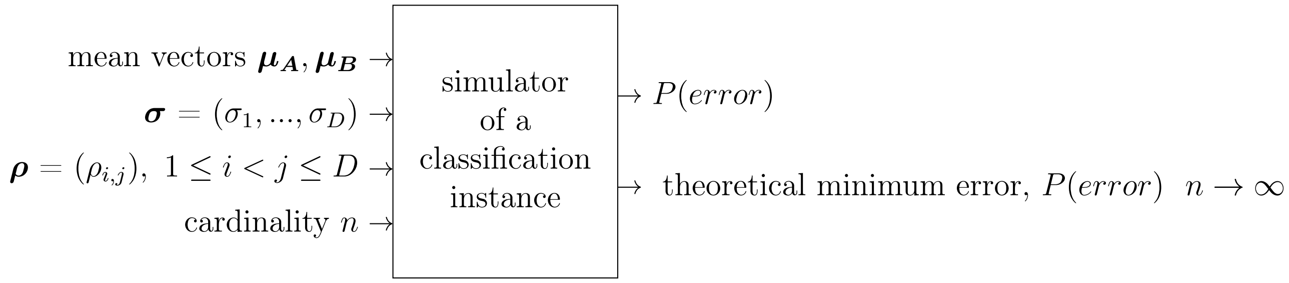

mean vectors $\boldsymbol{\mu_A}, \boldsymbol{\mu_B}$ →
$\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_D)$ →
$\boldsymbol{\rho} = (\rho_{i,j}),\ 1 \le i < j \le D$ →
cardinality $n$ →
simulator of a classification instance
→ $P(error)$
→ theoretical minimum error, $P(error)\ \ n \to \infty$

Figura 1 – General scheme of the simulator for estimating the error associated with the classification of two datasets collected from Gaussian samples. Each dataset is associated with one of two classes, and each class is associated with a mean from the mean vector. We assume that the covariance matrix, constructed based on $\boldsymbol{\sigma}$ and $\boldsymbol{\rho}$, is shared between the two classes, and we consider $n/2$ observations in each class.

Figure 1 illustrates the general scheme of the simulator for a classification instance, taking into account the elements mentioned above.

## 1.4 QUESTIONS OF INTEREST

To illustrate the results that can be obtained with the simulator, we focused on the following questions:

[1] Such classifiers are also called discriminators, as they consider only two classes.

- When is it possible to trade features for observations? In other words, when additional features cannot be collected, under what conditions can similar classification performance be achieved by collecting additional observations?
- When does an additional feature contribute to improving classification performance?
- What are the essential differences between classification with 1, 2, or 3 features? Specifically, when moving from 1 to 2 features, or from 2 to 3 features, what makes the additional feature contribute significantly in terms of the classifier's expected accuracy?

We implemented a simulator to generate Gaussian samples and used a Support Vector Machine (**SVM**) to classify the samples. We then provided some (partial) answers to the above questions. In particular, we identified scenarios where, with a small number of observations, it is advantageous to use only 2 features instead of 3, to avoid the problem of *overfitting*. Additionally, we identified that there is a threshold number of observations such that, above this threshold $n^\star$, it is always advantageous to use all 3 features.

## 1.5 CASES OF INTEREST AND SCENARIOS

Below, we present the cases of interest addressed by the simulator, as well as the scenarios considered in this thesis.

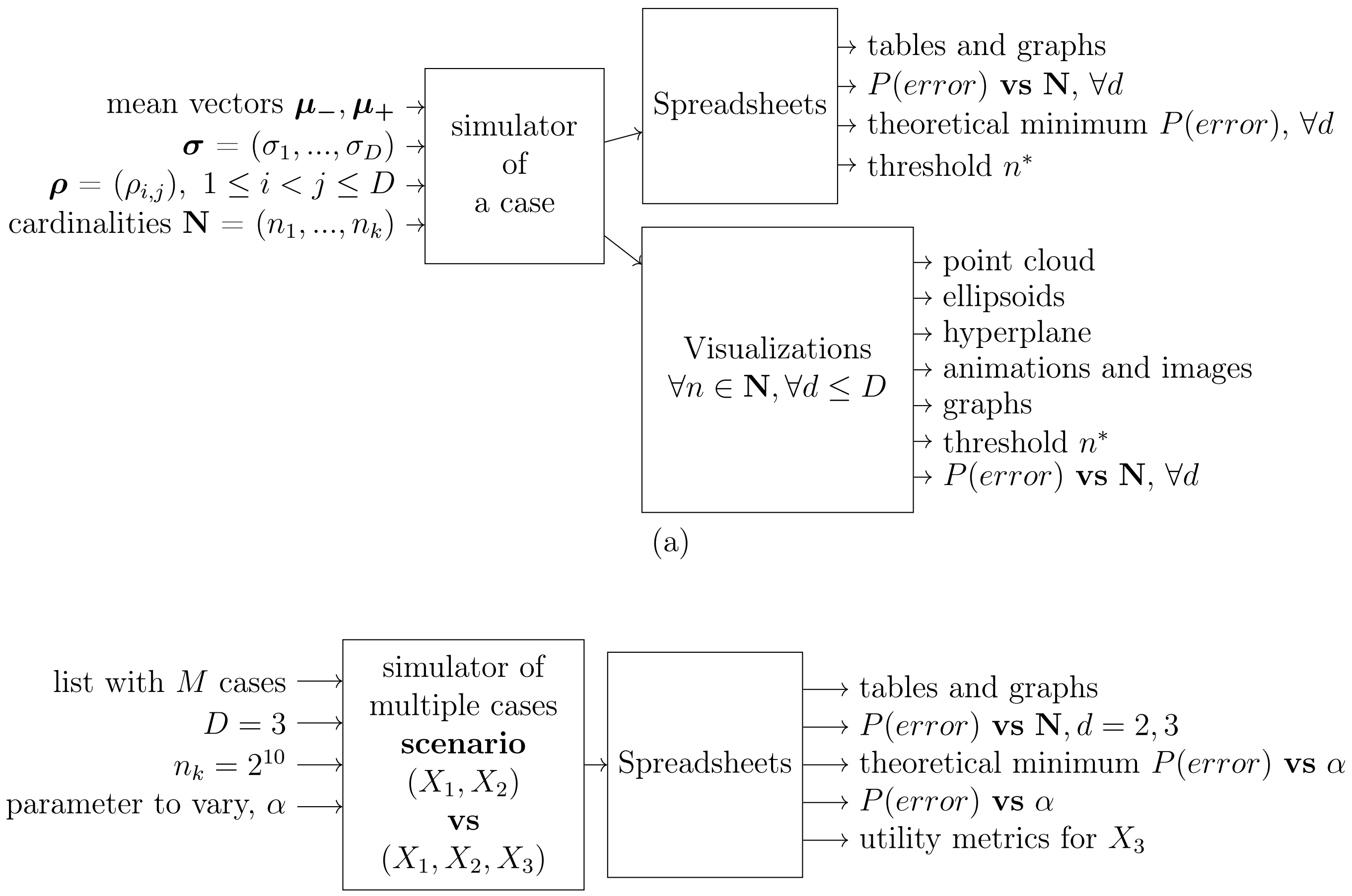
### 1.5.1 What is the impact of the number of observations and the number of features on classification error?

Next, we consider the impact of the number of observations and the number of features on classification error. We refer to this analysis as the analysis of a case of interest.

The general scheme of the simulator for analyzing a case of interest is illustrated in Figure 2(a). In Figure 2(a), we have a diagram illustrating how the simulator can be used to evaluate a case of interest, in which we vary the number of observations over the range of values given by the set $\mathbf{N}$ and the number of features between 1 and $D$, and verify the effect of both on classification error. As an example of output, we have curves capturing the classification error as a function of the number of observations. Each curve corresponds to a specific number of features.

The report obtained from simulating a case allows us, for example, to observe the influence of each feature added to a classifier.

In the top right corner of Figure 2(a), we indicate the outputs of the simulator when analyzing a case of interest. Such outputs are tables and graphs including, for example, the following three elements:


mean vectors $\boldsymbol{\mu}_-, \boldsymbol{\mu}_+$
$\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_D)$
$\boldsymbol{\rho} = (\rho_{i,j}),\ 1 \leq i < j \leq D$
cardinalities $\mathbf{N} = (n_1, ..., n_k)$
simulator of a case
Spreadsheets
tables and graphs
$P(error)$ **vs** $\mathbf{N}$, $\forall d$
theoretical minimum $P(error)$, $\forall d$
threshold $n^*$
Visualizations $\forall n \in \mathbf{N}, \forall d \leq D$
point cloud
ellipsoids
hyperplane
animations and images
graphs
threshold $n^*$
$P(error)$ **vs** $\mathbf{N}$, $\forall d$
list with $M$ cases
$D = 3$
$n_k = 2^{10}$
parameter to vary, $\alpha$
simulator of multiple cases
**scenario**
$(X_1, X_2)$
**vs**
$(X_1, X_2, X_3)$
Spreadsheets
tables and graphs
$P(error)$ **vs** $\mathbf{N}, d = 2, 3$
theoretical minimum $P(error)$ **vs** $\alpha$
$P(error)$ **vs** $\alpha$
utility metrics for $X_3$

(a)

(b)

Figura 2 – General scheme of the simulator: (a) case of interest, involving multiple classification instances, and (b) scenario involving multiple cases of interest, varying parameter $\alpha \in \{\sigma_3, \rho_{12}, \rho_{13}\}$, considering particularly 4 scenarios: scenario 1) varies $\sigma_3$, scenario 2) varies $\rho_{12}$, scenario 3) varies $\rho_{13}$ with $\rho_{13} = \rho_{23}$, $\rho_{12} = 0$, and scenario 4) same as scenario 3 but with $\rho_{12} \neq 0$.

- $P(error)$ as a function of the number of observations $\mathbf{N}$, for $d = 1, 2, \ldots, D$: we consider 3 approaches to calculate $P(error)$, as detailed in Appendix C, namely theoretical error, empirical error on the training set, and empirical error on the test set;

- Theoretical minimum $P(error)$, for $d = 1, 2, \ldots, D$: this error is also known as Bayes Risk, see Appendix C;

- Threshold $n^\star$: the threshold $n^\star$ corresponds to the number of observations beyond which the presence of feature $X_3$ becomes advantageous, given that the other two features $X_1$ and $X_2$ are already present. For more details, see Section 3.1. It is worth mentioning that this threshold depends on multiple factors, including the variance of the features and the correlation between them, as discussed below.

### 1.5.2 What is the impact of feature correlation and variance on classification error?

Next, we consider the impact of feature variance and correlation on classification error. We refer to this analysis as the analysis of a scenario, which consists of a set of cases of interest.

In Figure 2(b), we analyze a scenario composed of a set of cases that aim to compare various classifiers, taking into account the impact of feature variance and correlation on classification error. We can still compare a classifier with 2 features $(X_1, X_2)$ against a classifier with 3 features $(X_1, X_2, X_3)$. The report obtained from executing a scenario allows us to understand the influence of varying a particular parameter of the Gaussian distribution on the discriminative power of a classifier.

We observed changes in the discriminative power by adding feature $X_3$ in all scenarios. For this reason, we denote features $X_1$ and $X_2$ as present features, and feature $X_3$ as a candidate feature, as it is a candidate to be added to the set of present features.

The scenarios considered in this thesis are:

**Scenario 1:** We vary the dispersion parameter $\sigma_3$, that is, the variance, of feature $X_3$.

**Scenario 2:** We vary the correlation parameter $\rho_{12}$ between features $X_1$ and $X_2$.

**Scenario 3:** We vary the correlation parameters $\rho_{13}$ and $\rho_{23}$, which refer to the correlation between feature $X_3$ and feature $X_1$, and between feature $X_3$ and feature $X_2$. In this scenario, $\rho_{13} = \rho_{23}$ and $\rho_{12} = 0$.

**Scenario 4:** This scenario is equivalent to scenario 3, except that $\rho_{12} \neq 0$.

Each scenario corresponds to $L$ cases of interest. In each case in the list of $L$ cases of interest, a distinct value is assigned to parameter $\alpha$. The other parameters of the covariance matrix remain constant across all $L$ cases.[2]

It is worth mentioning that the parameter $\alpha$ to be varied corresponds to one of the following three parameters: $\sigma_3$, $\rho_{12}$, or $\rho_{13}$. In scenario 1, we vary $\sigma_3$, in scenario 2 we vary $\rho_{12}$, and in scenarios 3 and 4 we vary $\rho_{13}$, assuming $\rho_{13} = \rho_{23}$. In scenario 3, we assume $\rho_{12} = 0$, and in scenario 4, we assume $\rho_{12} \neq 0$.

In the bottom right corner of Figure 2(b), we indicate the outputs of the simulator for each scenario. Such outputs summarize the simulation of $L$ cases of interest in tables and graphs, including, for example, the following elements:

- $P(error)$ as a function of the number of observations $\mathbf{N}$, with each curve corresponding to a value of $d$, $d = 2, 3$, and for each value of the parameter $\alpha$ considered;

[2] In economics, such an analysis where one parameter is modified while others are kept constant is referred to as an analysis where one parameter is varied while the others are held ceteris paribus.

- Theoretical minimum $P(error)$ as a function of the values of the parameter $\alpha$ considered;
- $P(error)$ as a function of the values of the parameter $\alpha$ considered, with a curve for each value of $n$ considered, $n \in \mathbf{N}$.

The first class of curves above considers both cases with 2 and 3 features, while the latter two classes of curves consider the scenario with 3 features. In these latter two classes, to capture the relevance of the third feature, $X_3$, we also consider special metrics that indicate the utility of $X_3$, as detailed in Appendix **??**.

## 1.6 ILLUSTRATION OF RESULTS AND METHODOLOGY

To illustrate the relationship between sample size, number of features, and the asymptotic error probability of the trained classifier, we considered a simple dataset with either 1 or 2 features. We assumed that the conditional distribution of each sample given its class is a univariate or bivariate Gaussian distribution (as in, for example, (WILLETT; SWASZEK; BLUM, 2000)).

Tabela 1 – Error Probabilities for **SVM** (multiplied by 100), $\rho = 0$

| | | $L_1(n)$, | $L_2(n;\delta)$, 2 features | | | | | |
|---|---|---|---|---|---|---|---|---|
| $n$ | 1 feature | $\delta$=1 | $\delta$=2 | $\delta$=3 | $\delta$=4 | $\delta$=5 | $\delta$=6 | $\delta$=7 |
| 2 | 20.50 | 14.99 | 23.29 | 26.72 | 28.20 | 28.93 | 29.36 | 29.63 |
| 4 | 18.83 | 13.91 | 21.73 | 24.47 | 25.62 | 26.20 | 26.52 | 26.73 |
| 8 | 17.63 | 11.98 | 18.95 | 21.02 | 21.82 | 22.21 | 22.42 | 22.54 |
| 16 | 16.87 | 10.24 | 16.28 | 17.92 | 18.55 | 18.86 | 19.01 | 19.11 |
| 32 | 16.39 | 9.18 | 14.74 | 16.22 | 16.78 | 17.05 | 17.20 | 17.29 |
| 64 | 16.15 | 8.57 | 13.96 | 15.39 | 15.94 | 16.20 | 16.34 | 16.43 |
| 129 | 16.01 | 8.24 | 13.57 | 14.99 | 15.53 | 15.79 | 15.93 | 16.02 |
| 256 | 15.94 | 8.06 | 13.38 | 14.79 | 15.33 | 15.59 | 15.73 | 15.82 |
| 512 | 15.90 | 7.96 | 13.28 | 14.69 | 15.23 | 15.49 | 15.63 | 15.72 |
| 1024 | 15.88 | 7.91 | 13.23 | 14.64 | 15.18 | 15.44 | 15.58 | 15.67 |
| $n^\star$ | | | 12 | 28 | 46 | 70 | 98 | 130 |
| Linear Regression for Error Probability: $E(n) = \alpha' + \beta'/n$. | | | | | | | | |
| $\alpha'$ | 0.159 | 0.080 | 0.132 | 0.146 | 0.151 | 0.154 | 0.155 | 0.156 |
| $\beta'$ | 0.142 | 0.333 | 0.468 | 0.522 | 0.542 | 0.552 | 0.556 | 0.558 |
| $n^\star$ | | | 12 | 30 | 52 | 78 | 106 | 136 |

In the bivariate case, we have a correlation coefficient $\rho$; one of the features has variance 1 and the other has variance $\delta^2$. We observe that for $\rho = 0$, the discriminative power of the second feature decreases as $\delta$ increases. The covariance matrix, shared by both classes, is given by

$$\Sigma = \begin{bmatrix} 1 & \rho\delta \\ \rho\delta & \delta^2 \end{bmatrix}.$$

Although this workload is admittedly simple, data collected in practice can often be represented by a bivariate Gaussian distribution. Moreover, this simple model already serves our purposes, which are to show that *(a)* depending on the discriminative power of the second feature, it may be worthwhile to ignore it, and *(b)* additional samples may compensate for the lack of a feature, as illustrated in Table 1.

## 1.7 SIMULATION

### 1.7.1 Review of Objectives

To recap, among our objectives, we aim to:

1. Compare scenarios where 3 features, $X_1$, $X_2$, and $X_3$, are present against scenarios where only $X_1$ and $X_2$ are present.
2. Evaluate the expected classification error as the dataset cardinality $n$ increases.
3. Estimate how the threshold $n^*$ behaves for a given combination of covariance matrix parameters.

### 1.7.2 Classification Problem

We define a classification problem characterized by:

- A combination of covariance matrix parameters of the Gaussian distribution considered, namely
$$\Sigma = \begin{bmatrix} 1 & \rho\delta \\ \rho\delta & \delta^2 \end{bmatrix}$$
in the bivariate case, and more generally, still in the bivariate case,
$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}.$$
In the three-dimensional case, we have as parameters $\sigma_1, \sigma_2, \sigma_3, \rho_{12}, \rho_{13}, \rho_{23}$, which form the covariance matrix,
$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 & \rho_{13}\sigma_1\sigma_3 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 & \rho_{23}\sigma_2\sigma_3 \\ \rho_{13}\sigma_1\sigma_3 & \rho_{23}\sigma_2\sigma_3 & \sigma_3^2 \end{bmatrix}.$$
Unless stated otherwise, we focus on the bivariate and three-dimensional cases in this work.
- A set of cardinalities $\mathbf{N} = \{n_1, ..., n_k\}$.

For this classification problem, we generate numerous synthetic datasets and find, for each of them, the separating hyperplane using a linear classifier (**SVM**).

#### 1.7.2.1 What is the Impact of the Number of Observations and Features on Classification Error?

To answer objectives 1 and 2 from the list presented in Section 1.7.1, we plot, for each case of interest, two error curves. The first curve uses three features, $X_1$, $X_2$, and $X_3$, while the other uses only $X_1$ and $X_2$. Each curve expresses $P(Error)$ as a function of the number of observations, $n$. The threshold $n^{\star}$, present in Table 1, is the abscissa of the point where the two curves intersect.

#### 1.7.2.2 What is the Impact of Feature Correlation and Variance on Classification Error?

To answer objective 3 from the list of objectives presented in Section 1.7.1, we proceed with the following steps. For each of the four scenarios of interest indicated in Section **??**:

1. Specify a list containing $L$ combinations of parameters characterizing cases to be simulated.
2. Execute the simulation for all $L$ parameter combinations in this list.
3. Interpret the results and connect the interpretations with the geometry of the problem.

## 1.8 ADDITIONAL DETAILS ABOUT THE CLASSIFICATION PROBLEM CONSIDERED

We have a classification problem with two classes, where:

- The random vector $(X, Y)$ follows a joint probability distribution $P$.
- $X$ is a $d$-dimensional feature vector.
- The response $Y$ can be -1 or $+1$.
- We only consider classifiers from a specific dictionary $H$ (e.g., the set of linear classifiers).

Our objective is to use a dataset $\mathcal{D}$ with $n$ pairs $(x_i, y_i)$ to choose a classifier $h \in H$. How do we make this choice?

The smaller the expected error rate, the more efficient a classifier will be considered. Since we are working with synthetic simulated data, the best way to evaluate a classifier's expected performance is by calculating its theoretical loss.

Once we select the best classifier for each case of interest, we investigate a potential trade-off between dimensionality and cardinality in the context of a binary classification problem. The idea is to compare two situations, $S_1$ and $S_2$, where:

- In situation $S_1$, we have dimensionality $d'$ and cardinality $n'$.
- In situation $S_2$, we have dimensionality $d''$ and cardinality $n''$.
- All features present in $S_1$ are also present in $S_2$.

If $d' < d''$, can the lower dimensionality of situation $S_1$ be compensated for by an increase in the dataset cardinality $n'$, i.e., by making $n' > n''$?

Of course, answering this question is not just about comparing dimensionalities and cardinalities. Suppose we consider adding new features to the dataset $A$ to increase its discrimination capacity. Indeed, there are features that, when added, significantly improve the model's discriminative power, while others would be practically useless for this purpose. Thus, it is not just a matter of how many new features are added to the dataset. Depending on the effective contribution of each new feature, in terms of increasing the model's discriminative power between the two populations involved in the problem, adding it will be more or less advantageous.

In Table 2, we present a summary of the notation adopted in the remainder of this work.

Tabela 2 – Notation Table

| Variable | Meaning |
|---|---|
| $n$ | Cardinality being evaluated (each class has $n/2$ observations) |
| $n_k$ | Maximum cardinality considered in the case of interest |
| $\mathbf{N}$ | Set of cardinalities considered in the case of interest |
| $d$ | Dimensionality being evaluated |
| $D$ | Maximum dimensionality considered in the case of interest |
| $(X, Y)$ | Random vector of features and response |
| $Y$ | Response, which can be -1 or $+1$ |
| $X$ | $d$-dimensional feature vector |
| $\mathcal{D}$ | Dataset (synthetically sampled from Gaussians) |
| $\boldsymbol{\mu}_+$ | Mean vector of the Gaussian distribution for class $+1$, by default<br>($+1$,$+1$) in the bivariate case and ($+1$,$+1$,$+1$) in the three-dimensional case |
| $\boldsymbol{\mu}_-$ | Mean vector of the Gaussian distribution for class -1, by default<br>(-1,-1) in the bivariate case and (-1,-1,-1) in the three-dimensional case |
| $\boldsymbol{\mu}$ | Mean vector, $\boldsymbol{\mu} = (\boldsymbol{\mu}_+, \boldsymbol{\mu}_-)$ |
| ($\boldsymbol{\sigma}$, $\boldsymbol{\rho}$) | Parameters of the covariance matrix shared between the two classes |
| $\Sigma$ | Covariance matrix, shared between the two classes |
| ($\Sigma$, $\boldsymbol{\mu}$) | Parameters of the Gaussian model |
| $\delta^2$ | Variance of feature $X_2$, in a special case of the bivariate model |
| $P(error)$ | Error probability of the classifier |
| $h(x)$ | Classification function, or classifier |
| $L(h(x))$ | Loss function (Error) for classifier $h(x)$ |
| $n^\star$ | Cardinality threshold beyond which it is advantageous to use feature $X_3$ given features $X_1$ and $X_2$ |
| $\Phi(x)$ | CDF of the univariate Gaussian with mean 0 and variance 1, i.e.,<br>Cumulative distribution function of the standard normal distribution |

## 1.9 TRAINING EMPIRICAL ERROR AND THEORETICAL ERROR

Below, we introduce the concepts of **Training Empirical Error** and **Theoretical Error**. These concepts will be explored in more detail in Section 2.4 (equations (2.1) and (2.3), respectively).

**Training Empirical Error** refers to the fraction of misclassified samples, as observed exclusively using the training data. For calculation purposes, we assume the following are known: (a) the training data; (b) the classifier function, empirically obtained from the training data. As seen in equation 2.1

**Theoretical Error** refers to the fraction of misclassified samples, as predicted by the Gaussian probabilistic model defined by:

- standard deviations $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_D)$
- correlations $\boldsymbol{\rho} = (\rho_{i,j}),\ 1 \leq i < j \leq D$
- means $\mu_+ = [+\mu_1, \ldots, +\mu_D]$ and $\mu_- = [-\mu_1, \ldots, -\mu_D]$

To calculate the **Theoretical Error**, we assume the following are known: (1) the parameters of the Gaussian model ($\boldsymbol{\sigma}$, $\boldsymbol{\rho}$, $\boldsymbol{\mu}$); (2) the equation of the class separation surface, learned empirically from the training data. As seen in equation 2.3

### 1.9.1 Univariate Example

Next, we consider the example illustrated in Figures 3 and 4. In the examples of this section, we consider two univariate normal distributions, both with a standard deviation of $\sigma = 1$ and means $\mu_{red} = -1$ and $\mu_{blue} = +1$. In the terminology of Section 1.8, we refer to the two data classes as corresponding to responses $Y = -1$ and $Y = +1$. In the following example, for visual purposes, red corresponds to -1, and blue corresponds to $+1$.

In Figure 3, we consider a sample containing two observations ($n = 2$), one from each class, where the red and blue observations correspond to the values 0 and $+1$, respectively (marked on the x-axis in Figure 3). In this case, the best classifier learned from the data consists of a classifier that labels the points to the left of $c = 0.5$ as red points, and those to the right as blue. There is nothing more sophisticated that can be learned from such limited data. The **Training Empirical Error** is 0, but the **Theoretical Error** is equal to

$$L_1(0.5) = \frac{1}{2}(1 - \Phi(0.5)) + \frac{1}{2}(1 - \Phi(1.5)) = 0.188$$

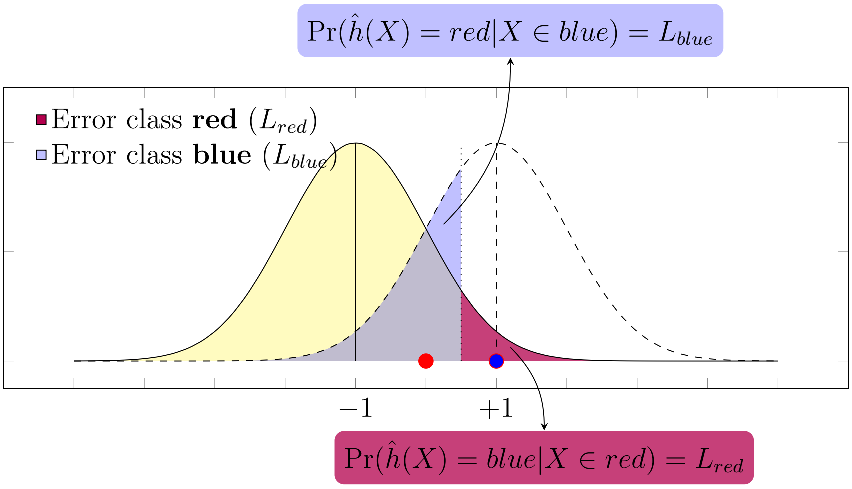


Figura 3 – Binary classification problem, univariate case: the empirically found cut-off point is $c = 0.5$

In Figure 4, we consider a sample containing four observations ($n = 4$), two from each class, with the red and blue observations corresponding to the values $\{-1, +2\}$ and $\{-2, +1\}$, respectively. In this case, the best classifier learned from the data consists of a classifier that labels points to the left of the origin ($c = 0$) as red, and points to the right as blue. The **Training Empirical Error** is 0.5, but the theoretical error, **Theoretical Error**, is equal to

$$L_1(0) = 1 - \Phi(1) = 0.158$$

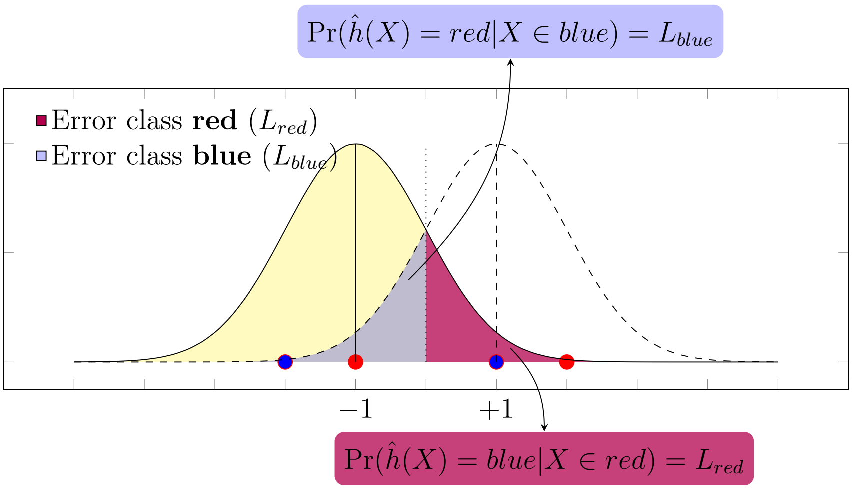
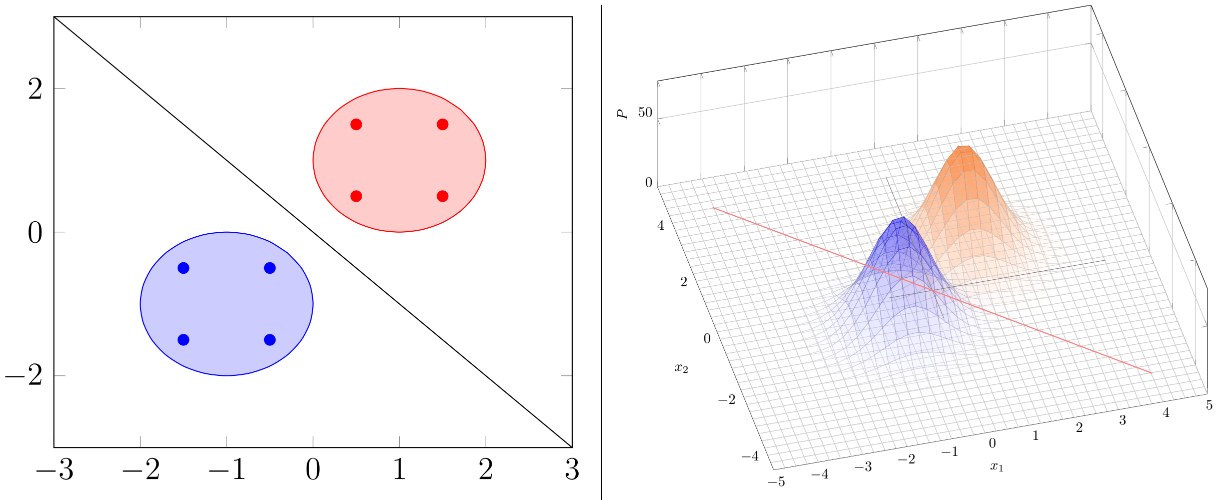


Figura 4 – Binary classification problem, univariate case: the empirically found optimal cut-off point is $c = 0$

Comparing the two figures above, we notice that in the first one, the empirical error is lower, but the theoretical error is higher.

### 1.9.2 Bivariate Example

As a second example, to illustrate the distinction between empirical error and theoretical error, Figure 5 shows two bivariate Gaussians. Each Gaussian has a covariance matrix equal to the identity, and the means of the Gaussians are (-1,-1) and (+1,+1), for the blue and red Gaussians, respectively. In Figure 5(a), we see 4 observations from each Gaussian, generating a classification empirical error of zero. The best separation line is the line $y = -x$. In Figure 5(b), we observe the underlying Gaussian model, corresponding to a theoretical error greater than zero, calculated in Appendix D.2. One of the purposes of the simulator developed in this work is to evaluate different errors, theoretical and empirical, under different scenarios and identify how the different parameters of the Gaussians affect the errors.



(a) (b)

Figura 5 – Bivariate Gaussian: (a) empirical error equal to zero, considering only generated data; (b) theoretical error greater than zero, considering underlying model.

## 1.10 CONTRIBUTIONS

Our main contribution is a simulator for evaluating the performance of the **SVM** classification algorithm under a set of Gaussian data. The generated source code is available on Github at the following link:

- <https://github.com/paulorenatoaz/slacgs>

The development of the simulator is our main contribution, as detailed in Chapter 2. With the simulator, we gathered several interesting observations for bivariate and trivariate Gaussians, and such illustrative results are presented in Chapter **??**.

## 1.11 OUTLINE

The rest of this work is organized as follows. In Chapter 2, we present the implemented simulator. In Chapter **??**, we illustrate some results that can be obtained from the simulator. Chapter 4 discusses related work, and Chapter 5 concludes.

We then present a series of additional results in appendices. Appendices A and B provide a contextualization of the problem considered in this work, as well as an introduction to Gaussian distributions, respectively. Then, Appendices **??** and **??** discuss types of errors and the calculation of such errors in special cases when the number of samples is large. Still considering a large number of samples, Appendices E and F present additional results for the 2D and 3D cases, respectively. In contrast, Appendix G formally discusses Bayes error for a small sample, with two observations, one in each class. Appendix H presents a heuristic model for understanding the behavior of Bayes error. In particular, in Appendix H.4, we indicate how the proposed simulator generates results to capture the relevance of the third attribute, $X_3$, considering special metrics that indicate the usefulness of $X_3$, assuming $X_1$ and $X_2$ are given. In other words, we evaluate the gain of considering the trivariate scenario as opposed to the bivariate scenario, using the simulator along with heuristics.

# 2 SLACGS: SIMULATOR FOR LOSS ANALYSIS OF CLASSIFIERS USING GAUSSIAN SAMPLES

In this chapter, we present the modules of the implemented simulator. The simulator, titled SLACGS, aims to analyze classifier errors under Gaussian load. In particular, one of the applications considered, mentioned in the introduction and to be illustrated in the next chapter, is to evaluate the trade-off between collecting additional samples (observations) or attributes.

## 2.1 OVERVIEW

Figure 6 illustrates the general scheme of the simulator.[1]

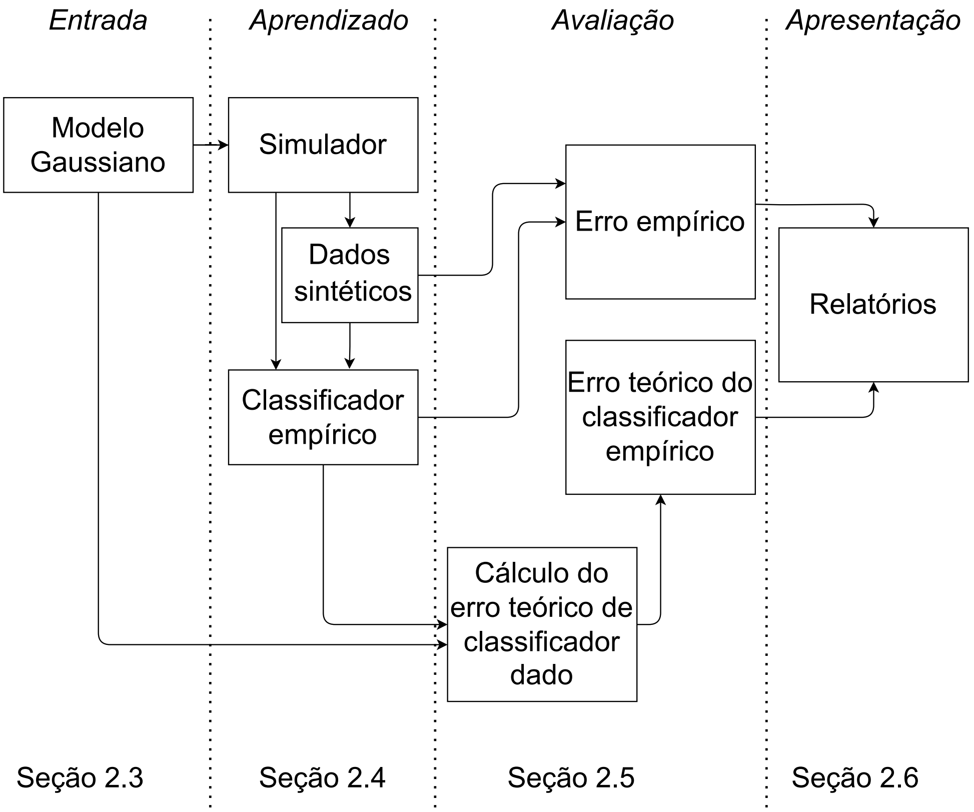


Figura 6 – Simulator scheme

Next, we briefly describe each of the elements in the figure above, from left to right. In the remainder of this chapter, we detail each of the modules considered.

### 2.1.1 Input: Gaussian Model

The input of the simulator consists of the description of the Gaussian model to be used to generate observations that will be classified. We assume there are two classes of

[1] Additional documentation can be found at: <https://slacgs.netlify.app/>

observations. Each class is characterized by a multivariate Gaussian distribution. The user must specify the number of observations to be generated, as well as the parameters of the corresponding multivariate Gaussian for each class.

### 2.1.2 Learning

During learning, synthetic data is generated, and an **SVM** classifier is used to separate the data into two classes. The synthetic data is generated incrementally, meaning that the dataset with $n_{i+1}$ observations is built from the dataset with $n_i$ observations by adding $n_i$ observations ($n_{i+1} = 2n_i$). For each dataset of interest, an **SVM** classifier is trained, and a separating surface between the classes is generated as output.

### 2.1.3 Evaluation: Error Functions

To evaluate the quality of the classifier obtained, we consider the following errors:

- Theoretical error: estimated using probability theory and the generated classifier.
- Empirical error: estimated directly from the data.
  - Empirical error with training data: estimated using an empirical approach only with training data.
  - Empirical error with test data: estimated using an empirical approach with distinct training and test data.

### 2.1.4 Presentation: Reports

Next, we briefly discuss report generation.

- Reports with results are stored in Google Sheets.[2]
- The spreadsheets are stored in a folder on Google Drive. By default, they are stored in the following folder:

  ```
  slacgs.demo.<user_email>
  ```

  belonging to the slacgs Google service account and shared with the user's Google Drive account.
- Additionally, data visualization images are exported to a local folder within the user's local folder (<user>/slacgs/images/ or /content/slacgs/images (for G-colab)).

[2] Each of the following elements generates a report: Experiment Scenario, Customized Experiment Scenario, and Customized Simulations. The meaning of each element is detailed in the simulator documentation.

#### 2.1.4.1 Exported Reports

We export the following reports:

- Error Report: Primarily contains results focused on the evaluation of the error functions (Section 2.1.3) for each model dimensionality.
- Comparison Report: Primarily contains results focused on comparing the performance of the model for a pair of dimensionalities (e.g., 2D versus 3D).
- Scenario Report: Contains results from all simulations in a scenario and links to other reports (available only for comparison between 2D and 3D).

#### 2.1.4.2 Exported Images

We export the following images:

- Scenario Data Plots (*.gif*): Animation containing all data plots, for $n = 1024$, generated for all models in an experimental scenario.
- Simulation Data Plot with Error Curves (*.png* and *.gif*): Images containing data plots of a sample for each cardinality $n_i \in \mathbf{N}$, along with graphs containing Error Curves (theoretical, empirical training, empirical testing). An animation containing all these images is also exported.

Figure 7 illustrates the classes involved in the simulator. The Model class implements a Gaussian model (Section **??**). The Simulator class is responsible for implementing the main loop of the simulator (Section 2.3), and generating the error estimates (Section 2.4). Finally, the Report class generates the results (Section 2.5).

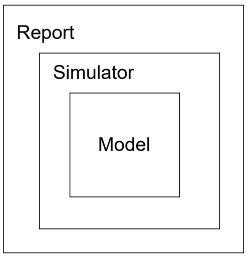


Figura 7 – Simulator class diagram.

In the next section, we detail the demo module.[3] With this module, we can also create and simulate new scenarios.

[3] The examples selected to be included in the demo module were chosen to replicate experiments already conducted by Prof. João I. Pinheiro in his Ph.D. thesis (PINHEIRO, 2021), for validation purposes of the simulator.

## 2.2 DEMO

Next, we detail the simulator's demo. Our purpose is to illustrate step by step the different functionalities of the simulator, pointing concretely to how they are implemented.

1. Download and Install

2. Configure/Start Report Service

3. Experiment Scenarios

4. Demonstration Functions

    a) **Run an Experiment Simulation:** Run a simulation for one of the experimental scenarios.

    b) **Add a Simulation to an Experiment Scenario:** Add simulation results to one of the experiment scenario spreadsheets.

    c) **Run a Customized Scenario:** Run a customized scenario and write the results to a Google Sheet shared with the user.

    d) **Add a Simulation to a Customized Scenario:** Add a simulation to a customized scenario spreadsheet.

    e) **Run a Customized Simulation:** Run a customized simulation for any dimensionality and cardinality.

    f) **Run All Experiment Simulations:** Run all simulations across all experimental scenarios.

### 2.2.1 Download and Install

To download and install the simulator, just run the following command in a prompt:

```
pip install slacgs
```

### 2.2.2 Configure Report Service

The simulator can generate cloud reports. To configure the report service, we have two options: 1) Use a Google cloud service account provided by the user[4] or 2) Use the server provided by slacgs, maintained by the slacgs developers, if you have the access password.

[4] <https://cloud.google.com/iam/docs/keys-create-delete>

```python
from slacgs.demo import *

## opt-1: configure report service with your own Google cloud service account key file
path_to_google_cloud_service_account_api_key = 'path/to/key.json'
set_report_service_conf(path_to_google_cloud_service_account_api_key)

# opt-2: configure report service to use slacgs server if you have the access password
set_report_service_conf()
```

### 2.2.3 Experiment Scenarios

To print the experiment scenarios to be analyzed, use the following command:

```
from slacgs.demo import print_experiment_scenarios

print_experiment_scenarios()
```

The function **print_experiment_scenarios()** prints the following table containing input parameters for all experiment scenarios in the format $[\sigma_1, \sigma_2, \sigma_3, \rho_{12}, \rho_{13}, \rho_{23}]$

| | Scenario 1 | Scenario 2 | Scenario 3 | Scenario 4 |
|---|---|---|---|---|
| 0 | [1, 1, 1.3, 0, 0, 0] | [1, 1, 2, -0.8, 0, 0] | [1, 1, 2, 0, -0.7, -0.7] | [1, 1, 1, -0.1, -0.6, -0.6] |
| 1 | [1, 1, 1.4, 0, 0, 0] | [1, 1, 2, -0.7, 0, 0] | [1, 1, 2, 0, -0.6, -0.6] | [1, 1, 1, -0.1, -0.5, -0.5] |
| 2 | [1, 1, 1.5, 0, 0, 0] | [1, 1, 2, -0.6, 0, 0] | [1, 1, 2, 0, -0.5, -0.5] | [1, 1, 1, -0.1, -0.4, -0.4] |
| 3 | [1, 1, 1.6, 0, 0, 0] | [1, 1, 2, -0.5, 0, 0] | [1, 1, 2, 0, -0.4, -0.4] | [1, 1, 1, -0.1, -0.3, -0.3] |
| 4 | [1, 1, 1.7, 0, 0, 0] | [1, 1, 2, -0.4, 0, 0] | [1, 1, 2, 0, -0.3, -0.3] | [1, 1, 1, -0.1, -0.2, -0.2] |
| 5 | [1, 1, 1.8, 0, 0, 0] | [1, 1, 2, -0.3, 0, 0] | [1, 1, 2, 0, -0.2, -0.2] | [1, 1, 1, -0.1, -0.1, -0.1] |
| 6 | [1, 1, 1.9, 0, 0, 0] | [1, 1, 2, -0.2, 0, 0] | [1, 1, 2, 0, -0.1, -0.1] | [1, 1, 1, -0.1, 0.0, 0.0] |
| 7 | [1, 1, 2.0, 0, 0, 0] | [1, 1, 2, -0.1, 0, 0] | [1, 1, 2, 0, 0.0, 0.0] | [1, 1, 1, -0.1, 0.1, 0.1] |
| 8 | [1, 1, 2.5, 0, 0, 0] | [1, 1, 2, 0.0, 0, 0] | [1, 1, 2, 0, 0.1, 0.1] | [1, 1, 1, -0.1, 0.2, 0.2] |
| 9 | [1, 1, 3.0, 0, 0, 0] | [1, 1, 2, 0.1, 0, 0] | [1, 1, 2, 0, 0.2, 0.2] | [1, 1, 1, -0.1, 0.3, 0.3] |
| 10 | [1, 1, 3.5, 0, 0, 0] | [1, 1, 2, 0.2, 0, 0] | [1, 1, 2, 0, 0.3, 0.3] | [1, 1, 1, -0.1, 0.4, 0.4] |
| 11 | [1, 1, 4.0, 0, 0, 0] | [1, 1, 2, 0.3, 0, 0] | [1, 1, 2, 0, 0.4, 0.4] | [1, 1, 1, -0.1, 0.5, 0.5] |
| 12 | [1, 1, 4.5, 0, 0, 0] | [1, 1, 2, 0.4, 0, 0] | [1, 1, 2, 0, 0.5, 0.5] | [1, 1, 1, -0.1, 0.6, 0.6] |
| 13 | [1, 1, 5.0, 0, 0, 0] | [1, 1, 2, 0.5, 0, 0] | [1, 1, 2, 0, 0.6, 0.6] | |
| 14 | [1, 1, 6, 0, 0, 0] | [1, 1, 2, 0.6, 0, 0] | [1, 1, 2, 0, 0.7, 0.7] | |
| 15 | [1, 1, 7, 0, 0, 0] | [1, 1, 2, 0.7, 0, 0] | | |
| 16 | [1, 1, 8, 0, 0, 0] | [1, 1, 2, 0.8, 0, 0] | | |
| 17 | [1, 1, 9, 0, 0, 0] | | | |
| 18 | [1, 1, 10, 0, 0, 0] | | | |
| 19 | [1, 1, 11, 0, 0, 0] | | | |
| 20 | [1, 1, 12, 0, 0, 0] | | | |
| 21 | [1, 1, 13, 0, 0, 0] | | | |

Tabela 3 – Experiment scenarios considered in the demo: $[\sigma_1, \sigma_2, \sigma_3, \rho_{12}, \rho_{13}, \rho_{23}]$

## 2.3 CLASS SIMULATOR

The `Simulator` class represents a simulator for analyzing classification error in Gaussian samples.

### 2.3.1 Elements

The simulator for an SLACGS model contains:

- $m$: an SLACGS Model object, as described in Section **??**.
- $\mathbf{d}$: a vector of dimensionalities $d_i$ to be simulated, where $d_i \leq D$, with $D$ being the maximum dimensionality of model $m$.
- $\mathbf{L}$: a list of loss functions $L_i$ to be estimated for each covariance matrix $\Sigma$, cardinality $n \in \mathbf{N}$, and dimensionality $d \in \mathbf{d}$. Note that

  $$\mathbf{L} = [L_{\text{THEORETICAL}}, L_{\text{EMPIRICAL\_TRAIN}}, L_{\text{EMPIRICAL\_TEST}}].$$

  All loss functions depend on the following elements:

  - covariance matrix $\Sigma$ of model $m$,
  - dimensionality $d < D$, where $D$ is the maximum dimensionality of model $m$,
  - cardinality $n \in \mathbf{N}$, where $\mathbf{N}$ is the list of cardinalities of model $m$.

  Thus, we can describe a loss function as $L_i = L_i(\Sigma, n, d)$. For more details, see Appendix **??**.

### 2.3.2 Method run()

The main loop of the simulator is presented below in Algorithm 1. For each cardinality of interest, $n_j$, we repeat the following steps $R(n_j)$ times: 1) generate random dataset (line 4); 2) train SVM (line 6); 3) update the error estimate (line 8). Note that the last two steps are executed for each of the dimensions of interest (loop between lines 5 and 10).

**Algoritmo 1** Simplified version of the `run` method of the `Simulator` class

```
 1: para each cardinality $n_j \in \mathbf{N}$ faça
 2:     Let $R(n_j)$ be the number of repetitions to be performed by the simulator
          for cardinality $n_j$
 3:     para each repetition $r$ from 1 to $R(n_j)$ faça
 4:         Generate dataset $\mathcal{D}(\Sigma, r) = \mathcal{D}_{train} \cup \mathcal{D}_{test}$
 5:         para each dimensionality $d_i$ of interest faça
 6:             Train an SVM classifier with the training set $\mathcal{D}_{train}$
                  in a $d_i$-dimensional space
 7:             para each error type $L_k$ faça
 8:                 Compute error $L_k(\Sigma, n_j, d_i; r, \mathcal{D})$ and update $\mathbb{E}[L_k(\Sigma, n_j, d_i)]$
 9:             fim para
10:         fim para
11:     fim para
12:     Report $\mathbb{E}[L_k(\Sigma, n_j, d_i)]$, $\forall i, \forall k$
13: fim para
```

In the algorithm above, we ignore some of the challenges related to the stopping criteria. In fact, it is possible that before running $R(n_j)$ iterations, we already have an accurate enough error estimate. To account for this, we need additional conditions, as illustrated in the more detailed version of the algorithm presented below (Algorithm 2).

#### 2.3.2.1 Detailed Algorithm

Algorithm 2 presents a more detailed view of the main loop of the algorithm. In line 11 of Algorithm 2, we check if the error calculated in two consecutive steps is smaller than a given threshold $\epsilon$, set by the user. If this occurs for all error estimates, we assume the estimate is accurate, and proceed to analyze the next cardinality. In particular, we verify the following three conditions, where $\Delta L_k(n, d) < \epsilon$ refers to the difference in error calculated between two consecutive steps of the algorithm:

- $\Delta L_{\text{EMPIRICAL\_TRAIN}}(n, d) < \epsilon$: the difference between $\mathbb{E}[L_{\text{EMPIRICAL\_TRAIN}}]$ calculated in two consecutive *steps* must be smaller than $\epsilon$ for a cardinality of $n$ samples and a dimensionality of $d$ attributes.

- $\Delta L_{\text{EMPIRICAL\_TEST}}(n, d) < \epsilon$: the difference between $\mathbb{E}[L_{\text{EMPIRICAL\_TEST}}]$ calculated in two consecutive *steps* must be smaller than $\epsilon$ for a cardinality of $n$ samples and a dimensionality of $d$ attributes.

- $\Delta L_{\text{THEORETICAL}}(n, d) < \epsilon$: the difference between $\mathbb{E}[L_{\text{THEORETICAL}}]$ calculated in two consecutive *steps* must be smaller than $\epsilon$ for a cardinality of $n$ samples and a dimensionality of $d$ attributes.

**Algoritmo 2** Extended version of the `run` method of the `Simulator` class

```
 1: **para** each untreated cardinality $n_j \in \mathbf{N}$ **faça**
 2:     Let $R(n_j)$ be the number of repetitions to be performed by the simulator
          for cardinality $n_j$
 3:     **para** each repetition $r$ from 1 to $R(n_j)$ **faça**
 4:         Generate dataset $\mathcal{D}(n_j, \Sigma, r) = \mathcal{D}_{train}(n_j, \Sigma, r) \cup \mathcal{D}_{test}(n_j, \Sigma, r)$
              adding elements to $\mathcal{D}(n_{j-1}, \Sigma, r) = \mathcal{D}_{train}(n_{j-1}, \Sigma, r) \cup \mathcal{D}_{test}(n_{j-1}, \Sigma, r)$
 5:         **para** each dimensionality $d_i$ of interest **faça**
 6:             Train an **SVM** classifier with the training set $\mathcal{D}_{train}$
                  in a $d_i$-dimensional space
 7:             **para** each error type $L_k$ **faça**
 8:                 Compute error $L_k(\Sigma, n_j, d_i; r, \mathcal{D})$ and update $\mathbb{E}[L_k(\Sigma, n_j, d_i)]$
 9:             **fim para**
10:         **fim para**
11:         **se** all last updates are "insignificant" **então**
12:                                                                              ▷
    The idea here is to save time by avoiding iterations with little gain –
      more details about the stopping criterion are in the text and documentation
13:             STOP analysis of the current cardinality, $n_j$, to address $n_{j+1}$
14:         **fim se**
15:     **fim para**
16:     **se** there are untreated cardinalities in $\mathbf{N}$ **então**
17:         CONTINUE
18:     **senão**
19:         **se** no crossing between error curves found **então**
20:                                                                              ▷
    if no crossing between error curves for dimensions $D$ and $D-1$ has been found yet,
      additional cardinalities are considered
21:             $n_{j+1} \leftarrow 2\max(\mathbf{N})$
22:             $\mathbf{N} \leftarrow \mathbf{N} \cup n_{j+1}$
23:             CONTINUE the simulation for new cardinality $n_{j+1}$
24:         **senão**
25:             Report $\mathbb{E}[L_k(\Sigma, n_j, d_i)]$, $\forall i, \forall j, \forall k$
26:         **fim se**
27:     **fim se**
28: **fim para**
```

After completing the simulation for all cardinalities $n \in \mathbf{N}$, we perform checks to evaluate if it is necessary to consider additional cardinalities. In line 16, we verify if there are any cardinalities that have not yet been addressed, and if they exist, they are processed.

Recall that one of the purposes of the simulator is to find the threshold $n^\star$, introduced at the end of Section **??**, which corresponds to the intersection of error curves. In line 19, we verify if the intersection between the error curves, corresponding to dimensions $D$ and $D-1$, has already been found. As long as the intersection has not yet been found, we proceed with the simulation using additional cardinalities. Clearly, for a sufficiently large amount of data (*big data*), meaning for a sufficiently large cardinality, the error when considering $D$ attributes is necessarily smaller than the error when considering $D-1$ attributes. On the other hand, for a sufficiently small cardinality, working with $D-1$ dimensions is advantageous compared to $D$ dimensions, as it avoids the problem of *overfitting*. Therefore, in all experiments carried out, we observed that the error curves intersect, and one of our objectives with the simulator is to find the crossing point $n^\star$. Thus, while such a point is not found, the simulation continues with additional cardinalities. Finally, in line **??**, we conclude the simulation, reporting all error estimates for all cardinalities, dimensions, and types of errors considered.

#### 2.3.2.2 Stopping Criterion

In summary, the simulation only ends when the following two conditions are met:

- Test if $n^\star$ has been found: Stop the simulation only if $L_k(\Sigma, n, d_i) < L_k(\Sigma, n, d_{i-1})$, for some $d_{i-1} \leq d_i$. If the condition has not been met for some $k$, it is still necessary to find the point $n^\star$ of intersection between error curves of type $k$.

- Convergence Test of $L$: Stop the simulation only if $L(\Sigma, n_i, d) - L(\Sigma, n_{i-1}, d) < \sqrt{\epsilon}$. Otherwise, execute additional repetitions of the simulation.

The first test above is implemented between lines 16 and **??** of Algorithm 2. The second test is not presented in the pseudocode for simplicity.

## 2.4 FUNCTIONS IN THE SIMULATOR.PY MODULE: CALCULATING THEORETICAL AND EMPIRICAL ERRORS

### 2.4.1 Empirical Error Function

The `loss_empirical` function calculates the empirical classification error based on the following elements:

- The empirical classifier $\hat{h}$. This classifier is part of the hypothesis space $H$ and, in this work, is obtained by an SVM trained using the dataset $\mathcal{D}_{train}$. This set contains $n$ observations with $d$ attributes, distributed in two classes, positive $(+)$ and negative $(-)$.

- A test dataset $\mathcal{D}_{test} = [\mathbf{X}_{test}, \mathbf{Y}_{test}]$.

Let $\hat{h}^{(\mathcal{D})}$ be the best empirical classifier in the dictionary $H$ for the dataset $\mathcal{D}$ with $d$ attributes and $n$ samples. Under these conditions, we can define two error rates.

The **Training Empirical Error Rate** of an empirical classifier $\hat{h}^{(\mathcal{D})}$ represents the proportion of observations classified incorrectly in the training dataset. In this case, $\mathcal{D}_{test} = \mathcal{D}_{train} = [\mathbf{X}_{train}, \mathbf{Y}_{train}]$:

$$\hat{L}(\hat{h}^{(\mathcal{D})}) = \frac{1}{n}\sum_{i=1}^{n} 1(y_i \neq \hat{h}(x_i)) \tag{2.1}$$

where $(x_i, y_i) \in [\mathbf{X}_{train}, \mathbf{Y}_{train}]$. Thus, $\hat{L}(\hat{h}^{(\mathcal{D})})$ is the **Training Empirical Error Rate** of the classifier $\hat{h}^{(\mathcal{D})}$, where $\mathcal{D}_{test} = \mathcal{D}_{train}$.

The **Testing Empirical Error Rate** of an empirical classifier $\hat{h}^{(\mathcal{D})}$ represents the proportion of observations classified incorrectly in a test dataset, in this case, $\mathcal{D}_{train} \neq \mathcal{D}_{test} = [\mathbf{X}_{test}, \mathbf{Y}_{test}]$:

$$\hat{L}(\hat{h}^{(\mathcal{D})}, \mathcal{D}') = \frac{1}{n}\sum_{i=1}^{n} 1(y_i \neq \hat{h}(x_i)) \tag{2.2}$$

where $(x_i, y_i) \in [\mathbf{X}_{test}, \mathbf{Y}_{test}]$, and the test set is given by $\mathcal{D}'$. Thus, $\hat{L}(\hat{h}^{(\mathcal{D})}, \mathcal{D}')$ is the **Testing Empirical Error Rate** of the classifier, $\hat{h}^{(\mathcal{D})}$, $\mathcal{D}_{\text{test}} \neq \mathcal{D}_{\text{train}}$.

The following Algorithm 3 summarizes the points above.

**Algoritmo 3** Calculation of Empirical Error

```
função LOSS_EMPIRICAL( $\hat{h}^{(\mathcal{D}_{train})}, \mathcal{D}_{test}$ )
    Calculate $\hat{L}(\hat{h}^{(\mathcal{D}_{train})}, \mathcal{D}_{test})$:
        $\hat{L} \leftarrow \frac{1}{n} \sum_{\forall (x_i, y_i)} 1(y_i \neq \hat{h}^{(\mathcal{D}_{train})}(x_i)), \quad \text{where } (x_i, y_i) \in \mathcal{D}_{test}$
    retorne $\hat{L}$
fim função
```

### 2.4.2 Theoretical Error Function

The `loss_theoretical` function uses probability theory to calculate the **Theoretical Error** based on the following elements:

- The empirical classifier $\hat{h}$. This classifier is part of the hypothesis space $H$ and, in this work, is obtained by an SVM trained using the dataset $\mathcal{D}_{train}$. This set contains $n$ observations with $d$ attributes, distributed in two classes, positive $(+)$ and negative $(-)$.

- The means of the classes $\mu_{(+)} = [+1, \ldots, +1]$ and $\mu_{(-)} = [-1, \ldots, -1]$.

- The covariance matrix $\Sigma$ used to generate observations in both classes of the dataset $\mathcal{D}$.

Note that the first element above, i.e., the classifier, is an input for both the empirical error function, introduced in the previous section, and the theoretical error function, introduced here. However, while the empirical error function uses data to estimate the error, the theoretical error function uses the parameters of the Gaussian model for this purpose.

We consider the general form of the equation of the separating hyperplane of the empirical classifier $\hat{h}^{(\mathcal{D})}$ in a space $\mathbb{R}^d$ given by the following expression:

$$a_1 x_1 + \ldots + a_d x_d = \kappa,$$

where $\kappa$ is a constant. We can rewrite the above equation as:

$$x_d = \frac{\kappa - (a_1 x_1 + \ldots + a_{d-1} x_{d-1})}{a_d}.$$

Alternatively,

$$x_d = \mathbf{w}' \cdot \mathbf{v}$$

where $\tilde{b} = -\kappa / a_d$,

$$\mathbf{w} = \begin{bmatrix} -a_1/a_d \\ \vdots \\ -a_{d-1}/a_d \\ -1 \end{bmatrix}$$

and

$$\mathbf{v} = \begin{bmatrix} x_1 \\ \vdots \\ x_{d-1} \\ -\frac{\kappa}{a_d} \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_{d-1} \\ \tilde{b} \end{bmatrix}.$$

The following Algorithm 4 summarizes the discussion above.

**Algoritmo 4** Calculation of **Theoretical Error**

```
função LOSS_THEORETICAL($\hat{h}, \Sigma$)
    Define $\mathbf{a} \leftarrow$ coefficients of the normal vector to the separating hyperplane of $\hat{h}$
    Define $\kappa \leftarrow$ intercept term of the separating hyperplane of $\hat{h}$
    Define $\tilde{b}$:
        $$\tilde{b} \leftarrow -\frac{\kappa}{a_d}$$
    Define $\mathbf{w}$:
        $$\mathbf{w} \leftarrow \begin{bmatrix} -a_1/a_d \\ \vdots \\ -a_{d-1}/a_d \\ 1 \end{bmatrix}$$
    retorne h_error_rate($\tilde{b}, \mathbf{w}, \Sigma$)
fim função
```

Note that in the above algorithm, when calling the function h_error_rate$(\tilde{b}, \mathbf{w}, \Sigma)$, we do not explicitly pass the vector of means $\boldsymbol{\mu}$ as an argument. This is because, in this work, we assume that the means of the positive and negative classes are given by $\mu_{(+)} = [+1, \ldots, +1]$ and $\mu_{(-)} = [-1, \ldots, -1]$.

According to the function above, after extracting the coefficients $\mathbf{w}$ and $\tilde{b}$, we call the function h_error_rate$(\tilde{b}, \mathbf{w}, \Sigma)$, described below.

Let $\hat{h}^{(\mathcal{D})}$ be an empirical classifier trained on the dataset $\mathcal{D}$, generated synthetically from multivariate normal distributions, divided equally into two classes, assuming the same covariance matrix $\Sigma$ and means symmetrically opposite with respect to the origin. We can estimate its classification error rate as follows:

$$L(\hat{h}^{(\mathcal{D})}) = \text{the rate of } \textbf{Theoretical Error} \text{ of the classifier } \hat{h}^{(\mathcal{D})}$$

where

$$L(\hat{h}^{(\mathcal{D})}) = \frac{1}{2}(1 - \Phi(\text{Distance}(+))) + \frac{1}{2}(1 - \Phi(\text{Distance}(-))), \tag{2.3}$$

and

- $\Phi(\cdot)$ is the cumulative distribution function of a Gaussian with mean zero and unit variance.
- Distance(+) and Distance(-) are the distances from the separating hyperplane to the mean of the positive and negative classes, respectively.

Knowing the equation of the separating hyperplane of the empirical classifier $\hat{h}^{(D)}$ and the covariance matrix $\Sigma$, we can calculate the distances using the following equations:

$$\text{Distance}(+) = \frac{\left|w_1 + \ldots + w_{d-1} - 1 - \tilde{b}\right|}{\sqrt{\Delta}}$$

$$\text{Distance}(-) = \frac{\left|w_1 + \ldots + w_{d-1} - 1 + \tilde{b}\right|}{\sqrt{\Delta}}$$

Thus, we can express the **Theoretical Error** as a function of the weight vector $\mathbf{w}$ and the bias $b$:

$$L_d(\mathbf{w}, b) = \frac{1}{2}\left(1 - \Phi\left(\frac{|\sum_{i=1}^{i=d} w_i - \tilde{b}|}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{|\sum_{i=1}^{i=d} w_i + \tilde{b}|}{\sqrt{\Delta}}\right)\right) \tag{2.4}$$

In the above expression, we have the following:

- $\mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_{d-1} \\ -1 \end{bmatrix}$ is the weight vector of the classifier.
- $\tilde{b}$ is the bias of the classifier.
- $\Delta$ can be obtained from $\Sigma$ and $\mathbf{w}$, as presented in Algorithm 5

Analyzing Algorithm 5, we observe that

$$h\_error\_rate(\tilde{b}, \mathbf{w}, \Sigma) = h\_error\_rate(-\tilde{b}, \mathbf{w}, \Sigma).$$

In the source code of the simulator, the variable *bias* of the function *loss_theoretical*, which implements Algorithm 4, corresponds to $-\tilde{b}$. If we had defined *bias* as $\tilde{b}$, the result of Algorithm 4, which calls Algorithm 5, would still be correct.

## 2.5 CLASS REPORT

The `Report` class is responsible for generating a report of the executed simulations.

### 2.5.1 Elements

The constructor of the `Report` class accepts a `Simulator` object as an argument.

The `Report` class contains the following elements. Some of these are received as input from the simulator, while others are generated as output from the simulator:

- `sim` (Simulator): Simulator object.
- `iter_N` (dict): Number of iterations effectively adopted for simulating each dimensionality and type of error.
- `max_iter_N` (list): Maximum number of iterations for each dimensionality.
- `loss_N` (dict): Error for each dimensionality and type of error.

**Algoritmo 5** Calculating the Error Probability for a Linear Classifier

```
função H_ERROR_RATE($\tilde{b}, \mathbf{w}, \Sigma$)
    Define $\boldsymbol{\lambda} \leftarrow$ lower triangular factor from the Cholesky decomposition of matrix $\Sigma$
    Define $\Delta$:
        $\tilde{\boldsymbol{\delta}} \leftarrow \boldsymbol{\lambda}^T \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_{d-1} \\ -1 \end{bmatrix} = \begin{bmatrix} \tilde{\delta}_1 \\ \vdots \\ \tilde{\delta}_d \end{bmatrix} \qquad (2.5)$
        $\Delta \leftarrow \sum_{i=1}^{d} \tilde{\delta}_i^2 \qquad (2.6)$
    Define Distance(+) and Distance(-):
        $\text{Distance}(+) \leftarrow \frac{\left| w_1 + \ldots + w_{d-1} - 1 - \tilde{b} \right|}{\sqrt{\Delta}}$
        $\text{Distance}(-) \leftarrow \frac{\left| w_1 + \ldots + w_{d-1} - 1 + \tilde{b} \right|}{\sqrt{\Delta}}$
    Define $P$(Error(+)) and $P$(Error(-)) using the cumulative distribution function of the standard normal (CDF):
        $P(\text{Error}(+)) \leftarrow 1 - \Phi(\text{Distance}(+))$
        $P(\text{Error}(-)) \leftarrow 1 - \Phi(\text{Distance}(-))$
    Calculate the final error probability:
        $P(\text{Error}) \leftarrow \frac{P(\text{Error}(+)) + P(\text{Error}(-))}{2}$
    retorne $P$(Error)
fim função
```

- `loss_bayes` (dict): Bayes error for each dimensionality.
- `d` (dict): Distance from the origin to the intersection point between the normalized ellipsoid and the principal diagonal for each dimensionality.
- `duration` (float): Duration of the simulation.
- `time_spent` (dict): Time spent for each dimensionality and type of error.
- `sim_tag` (dict): Attributes of the Simulator object.
- `model_tag` (dict): Attributes of the Model object.
- `loss_plot` (matplotlib.figure.Figure): Figure showing the error behavior.

The `Report` class encapsulates the other classes considered in the simulator. A diagram can be found in Figure 7.

### 2.5.2 Results

The results obtained by the Report class will be presented in the next chapter.

## 2.6 CHALLENGES

Below, we list some of the challenges encountered in implementing the simulator, along with the corresponding design decisions to address these challenges.

### 2.6.1 Obtaining Smooth Error Curves via Cumulative Data Generation

One of the initial challenges encountered was that the error curves were displaying erratic behavior, e.g., sometimes non-monotonic, as the number of observations increased. To overcome this challenge, we began constructing the datasets in a cumulative manner.

To obtain a smooth error curve, even with a limited number of observations, we constructed the dataset with $n$ observations from the dataset with $n'$ observations, $n' \leq n$, simply by adding more observations to the dataset cumulatively. Starting from the dataset with $n'$ observations, we added observations to it to obtain the dataset with $n$ observations.

For each replication $r$ of the simulator for cardinality $n$, we generated a dataset $\mathcal{D}(n, \Sigma, r)$, where $\Sigma$ is the covariance matrix of the Gaussian model.

- For the various replications involving $n = 2$ observations, we generated the datasets $\mathcal{D}(2, \Sigma, 1)$, $\mathcal{D}(2, \Sigma, 2)$, $\mathcal{D}(2, \Sigma, 3)$, $\mathcal{D}(2, \Sigma, 4)$, $\ldots$

- For the next cardinality $n = 4$, we generated the datasets $\mathcal{D}(4, \Sigma, 1)$, $\mathcal{D}(4, \Sigma, 2)$, $\mathcal{D}(4, \Sigma, 3)$, $\mathcal{D}(4, \Sigma, 4)$, $\ldots$ adding data to the respective previous sets

In this way, we ensured that the data generation was done incrementally:

$$\mathcal{D}(n_k, \Sigma, r) \subset \mathcal{D}(n_{k+1}, \Sigma, r).$$

We can see this property being guaranteed in line 4 of Algorithm 2.

Note that the simulator results are reproducible. By using a known pseudo-random number seed, we traverse an established sequence of pseudo-random states, ensuring that we will have the same results when reproducing the simulation with the same input.

### 2.6.2 Generalization of Theoretical Error Calculation

One of the challenges consisted of calculating the theoretical error associated with a given classifier. In the work of Prof. João I. Pinheiro (PINHEIRO, 2021), there is a simulation algorithm for each dimensionality $d$, and the calculation of $L(\hat{h}^{(\mathcal{D})})$ (**Theoretical Error** of the empirical classifier $\hat{h}^{(\mathcal{D})}$) is performed using analytical formulas available only for $d \leq 3$.

In this work, we created a generic **Theoretical Error** function for any dimensionality $d$, using linear algebra concepts, as shown in Algorithm 5. The proof of correctness of this algorithm is found in Appendix D.4.

### 2.6.3 Establishment of Stopping Criteria

We now discuss the stopping criteria.

#### 2.6.3.1 Stopping Criterion for a Given Cardinality $n$

One of the challenges was to generate results quickly and efficiently, avoiding unnecessary iterations of the simulator. To achieve this, we observed the evolution of the estimator $\mathbb{E}(L)$. If, after a sequence of $s$ replications, the mean of the error $L$ shows a variation smaller than a constant $\epsilon$, we stop the replications for cardinality $n_i$ and proceed to $n_{i+1}$. To simplify the presentation of Algorithm 2, we did not include this detail in its presentation, although it is covered in the implementation.

#### 2.6.3.2 Stopping Criterion for Simulation Termination

We recall the stopping criteria discussed in Section 2.3.2.2.

1. If the cardinality threshold $n^*$, corresponding to the crossing between the error curves for $D$ and $D-1$, has not been found, the simulation continues. To achieve this, additional cardinalities are added to the vector of cardinalities **N**. In this way, we extend the simulation until the threshold $n^*$ is found.

2. To test the convergence of the error curve $L$, for each dimensionality $d$, we observe the difference between $L(n_k)$ and $L(n_{k-1})$. If this difference is greater than a threshold established by the user, we assume that the curve has not yet converged. We then add additional cardinalities to the vector of cardinalities **N**. In this way, we extend the simulation until an asymptotic value for the error is found. Especially for cases where we do not have an analytical formula for calculating the Bayes risk (theoretical error), it is essential to have stable curves to estimate the error with satisfactory accuracy.

The stopping criteria can be observed in lines 16 to 19 of Algorithm 2.

### 2.6.4 Estimating the Bayes Error When No Analytical Formula is Available

When there is no analytical formula to calculate the Bayes error ($n > 3$), we estimate the error empirically from the average of the **L** values for the largest cardinality considered. The result is presented to the user in line 25 of Algorithm 2. This result is contrasted with the error obtained from the function that calculates the **Theoretical Error**, described in Algorithm 5.

### 2.6.5 Spreadsheet Formatting

When creating spreadsheets, it is essential to adapt to the dimensions of the cardinality vector $\mathbf{N}$ and the maximum dimension $D$ of the Gaussian model. To achieve this, we developed generic functions to generate reports. The reports can be produced from a simulation for any maximum dimensionality $D$ and for any dimension of the cardinality vector $\mathbf{N}$.

### 2.6.6 3D Data Visualization

A challenge consists of visualizing the data separator (in the form of a hyperplane) as well as visualizing the data itself (in the form of ellipsoids). To visualize the hyperplane and the ellipsoids, we use the variances and eigenvalues of the covariance matrix, respectively.

Ordering the variances and eigenvalues in descending order is important for visualizing hyperplanes and ellipsoids because it allows prioritizing and representing the main axes, which correspond to directions of maximum variance in the data or the largest eigenvalue. Below, we describe some code excerpts from the *Model.plot_ data_ 3d_ 3by3* function:

- Visualization of the Hyperplane
  - The code plots a 3D hyperplane, which is essentially a flat surface that separates the data into two classes.
  - To determine the orientation and position of the hyperplane, the code calculates the coefficients of the hyperplane equation based on the **SVM** (Support Vector Machine) classifier parameters.
  - The ordered variance values are used to identify the axes with the greatest variability.
  - By ordering the variances, it is ensured that the hyperplane's contour dimensions correspond to the axes with the highest variance, making the hyperplane visualization more representative.

```
# obtain sorted indices of standard deviations in descending order for the 3
↪  features
sorted_indices = np.argsort(sigmas)[::-1]

# get the largest standard deviation and multiply by 5 to set the hyperplane
↪  bound
hyperplane_bound = max(sigmas) * 5

axes = [None, None, None]
```

```
# Create a mesh for the axis with the largest and second largest standard
↪  deviation
axes[sorted_indices[0]], axes[sorted_indices[1]] = np.meshgrid(
    np.linspace(-hyperplane_bound, hyperplane_bound, 50),
    np.linspace(-hyperplane_bound, hyperplane_bound, 50))

# Calculate the corresponding values for the axis with the smallest standard
↪  deviation
axes[sorted_indices[2]] = (-clf.intercept_[0] -
↪  clf.coef_[0][sorted_indices[0]] * axes[sorted_indices[0]] -
                           clf.coef_[0][sorted_indices[1]] *
                           ↪  axes[sorted_indices[1]]) / clf.coef_[0][
                              sorted_indices[2]]

# Plot the hyperplane as a surface
ax.plot_surface(axes[0], axes[1], axes[2], color='gray', alpha=0.15,
↪  zorder=0)
```

- Visualization of Ellipsoids
  - The code also plots 3D ellipsoids for both classes. These ellipsoids are used to represent the shape and orientation of the data distribution for each class.
  - The ellipsoids are defined by their principal axes, which correspond to the eigenvectors of the covariance matrix, and their radii, which are related to the eigenvalues.
  - By sorting the eigenvalues in descending order, it is ensured that the longest axis (principal axis) of the ellipsoid is aligned with the direction of maximum variance, making the ellipsoid capture the main shape of the data distribution.

```
# Plot 3D ellipsoids for both classes
for samples, color in zip([samples_blue, samples_orange], [blue_color,
↪  orange_color]):
    mean_vector = np.mean(samples, axis=0)[list(combo)]
    sub_cov_matrix = cov_matrix[np.ix_(combo, combo)]

    eigvals, eigvecs = np.linalg.eigh(sub_cov_matrix)

    # Sort the eigenvalues in descending order and obtain the indices
    sorted_indices = np.argsort(eigvals)[::-1]

    # Reorder the eigenvalues and eigenvectors
    eigvals = eigvals[sorted_indices]
    eigvecs = eigvecs[:, sorted_indices]
```

```
for scaling_factor, alpha in zip([1, 4], [1, 0.3]):
    # Get the radii (widths) of the ellipsoid axes
    radii = scaling_factor * np.sqrt(eigvals)

    # Generate the ellipsoid mesh
    u = np.linspace(0, 2 * np.pi, 100)
    v = np.linspace(0, np.pi, 100)
    x = radii[0] * np.outer(np.cos(u), np.sin(v))
    y = radii[1] * np.outer(np.sin(u), np.sin(v))
    z = radii[2] * np.outer(np.ones_like(u), np.cos(v))

    ellipsoid = np.array([x.flatten(), y.flatten(), z.flatten()]).T  # reshape
    ↪  to (10000, 3)

    # Apply rotation and translation to the ellipsoid mesh
    transformed_ellipsoid = np.dot(eigvecs, ellipsoid.T).T
    transformed_ellipsoid += mean_vector

    # Reshape the transformed ellipsoid mesh to (100, 100, 3)
    transformed_ellipsoid = transformed_ellipsoid.reshape((100, 100, 3))

    ax.plot_surface(transformed_ellipsoid[:, :, 0], transformed_ellipsoid[:, :,
    ↪  1],
            transformed_ellipsoid[:, :, 2],
            rstride=4, cstride=4, color=color, alpha=ellipsoid_alpha,
            ↪  edgecolor='none')
```

In summary, ordering the variances and eigenvalues in descending order is essential to correctly align the hyperplane and ellipsoids concerning their most significant directions. This ensures that the visualizations adequately represent the separation between the classes and the data distribution.

It is worth mentioning that the axis ordering heuristic proposed in this section was derived empirically, by trial and error. Visually, we observe that it works, as illustrated in the next chapter, but we still do not have an adequate formal explanation to justify the proposed heuristic.

# 3 RESULTS

In this section, we present numerical results obtained with the constructed simulator.

## 3.1 WHEN DOES $X_3$ CONTRIBUTE TO INCREASING THE DISCRIMINATIVE POWER OF $(X_1, X_2)$? VISUALIZING CLASSIFICATION INSTANCES

Given a classification problem with the characteristics we are analyzing, i.e., two tri-normal distributions centered respectively at $\mu_{(+)} = (+1, +1, +1)$ and $\mu_{(-)} = (-1, -1, -1)$, both with covariance matrices equal to:

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 & \rho_{13}\sigma_1\sigma_3 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 & \rho_{23}\sigma_2\sigma_3 \\ \rho_{13}\sigma_1\sigma_3 & \rho_{23}\sigma_2\sigma_3 & \sigma_3^2 \end{bmatrix} \tag{3.1}$$

We know that, depending on the values of the six parameters (variances and correlations) that define this matrix $\Sigma$, there is a threshold $n^*$ for the number of observations, above which the presence of attribute $X_3$ becomes advantageous, given that the other two attributes $X_1$ and $X_2$ are already present. For a small number of observations, using all three attributes may lead to *overfitting*, and it is advantageous to use only two attributes. As the number of observations increases, it becomes worthwhile to use the third attribute, as the risk of *overfitting* ceases to exist.

**Question:** Given a matrix $\Sigma$, characterized by a particular combination of the six parameters $\sigma_1$, $\sigma_2$, $\sigma_3$, $\rho_{12}$, $\rho_{13}$, $\rho_{23}$ that define it, how can we evaluate the potential additional contribution of $X_3$ to the discrimination between the two groups, once $X_1$ and $X_2$ are already present?

To analyze this question, four scenarios were created, in each of which only three of the six parameters can be chosen freely. In each of these four scenarios, given the matrix $\Sigma$, the quotient $Q$ can be calculated,

$$Q = \frac{R_2}{R_3}, \tag{3.2}$$

where $R_3$ and $R_2$ are the Bayes risks (Bayes error) related, respectively, to the situation in which all three attributes are present and the situation in which only $X_1$ and $X_2$ are present. Supposedly, the larger the quotient $R_2/R_3$, the smaller $n^*$ should be.

### 3.1.1 Visualizing the Ellipsoids

Visualizing the ellipsoids corresponding to the two tri-normal distributions we want to separate is very useful for gaining intuition about the problem. Our simulator allows us to visualize such ellipsoids.

Figures 8 and 9 illustrate the ellipsoids in two different situations:

- In Figure 8, we have $\rho_{12} < 0$. As can be seen in the figure, this facilitates the classification task. After all, since the centroids are $(+1, +1, +1)$ and $(-1, -1, -1)$, the main axis of the ellipsoids aligns closely with the direction of the line passing through the two centroids;

- In Figure 9, we have $\rho_{12} > 0$. As can be seen in the figure, this makes the classification task more difficult. In the extreme case where $\rho_{12} = 1$, the main axis of the ellipsoids becomes perpendicular to the direction of the line passing through the two centroids.

With the visualization in Figures 8 and 9, it becomes easy to see the impact of $\rho_{12}$ on the classification difficulty, illustrating the usefulness of the simulator (see also Appendix H.1).

### 3.1.2 Summary

In summary, we know that the level surfaces corresponding to a given tri-normal distribution are ellipsoids centered at their respective centroid (mean vector). Therefore, if the two centroids are $(+1, +1, +1)$ and $(-1, -1, -1)$, the closer the direction of the main axis of these ellipsoids is to the direction of the line passing through these two centroids, the greater the intersection between the point clouds relative to the two groups, i.e., the greater the expected probability of classification error if all three attributes are present.

The proposed simulator helps to visualize these observations in concrete scenarios. Note that to draw the hyperplanes and ellipsoids in Figures 8 and 9, we used the ideas presented in Section 2.6.6. In particular, to draw the hyperplanes, we first constructed a mesh with the axes of the largest and second-largest standard deviations. This allows a clear visualization of the hyperplane separating the orange and blue ellipsoids, both in Figure 8 and Figure 9. Projections in the planes $X_1 \times X_2$, $X_1 \times X_3$, and $X_2 \times X_3$ are also presented in 2D in the figures.

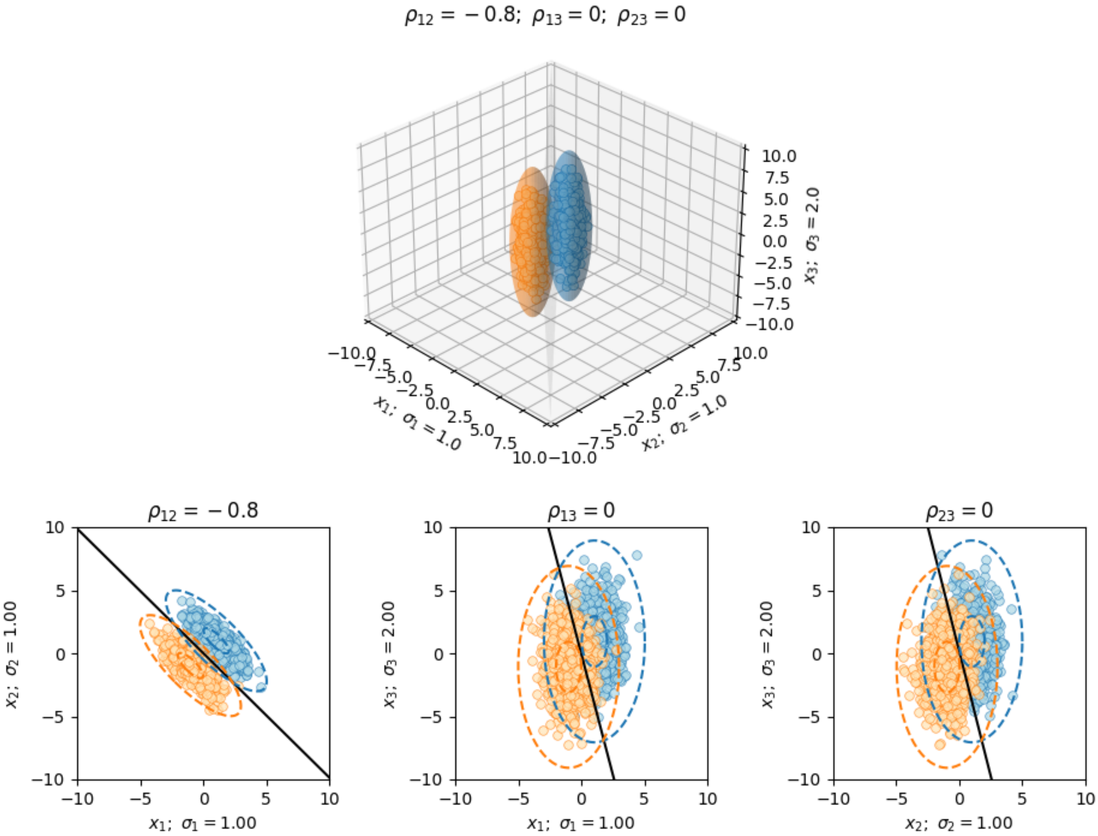


Figura 8 – Visualizing Classification Instance: $\boldsymbol{\sigma} = [1, 1, 2], \boldsymbol{\rho} = [-0.8, 0, 0]$; $n = 1024$

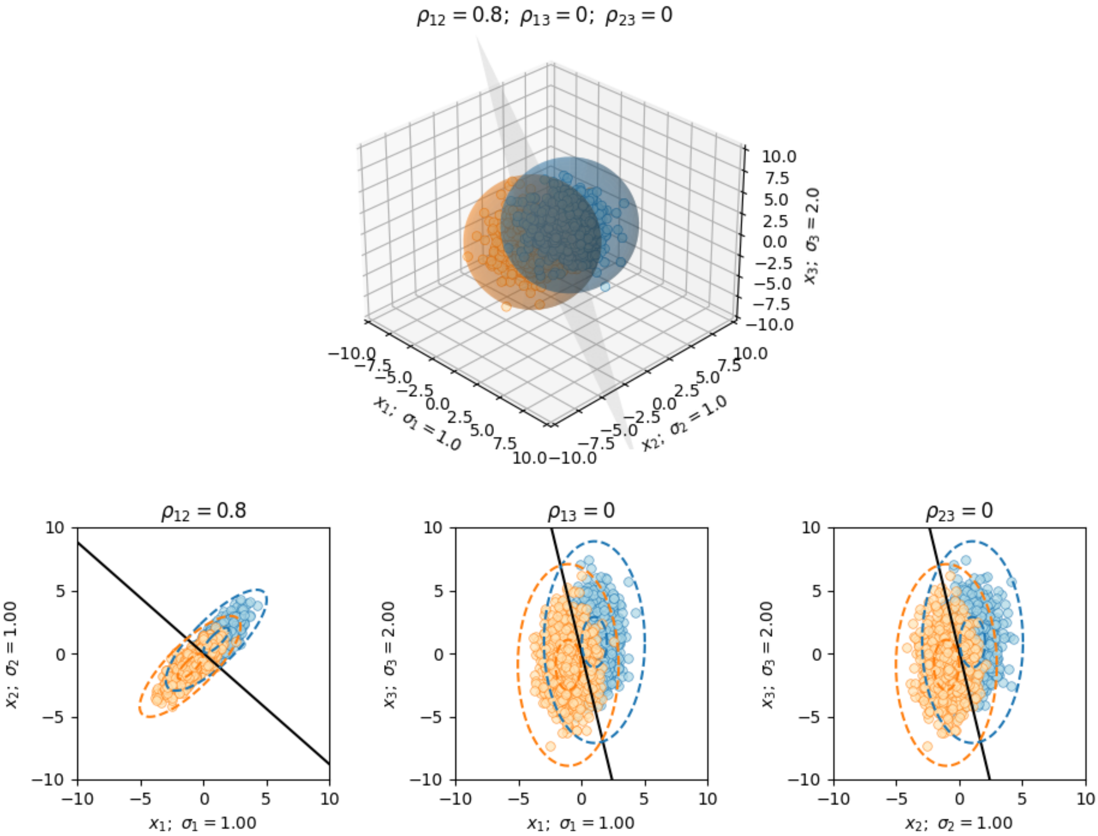


Figura 9 – Visualizing Classification Instance: $\boldsymbol{\sigma} = [1, 1, 2]$, $\boldsymbol{\rho} = [0.8, 0, 0]$; $n = 1024$

## 3.2 WHAT IS THE IMPACT OF THE NUMBER OF OBSERVATIONS? REPORT PRODUCED BY A SIMULATION CASE

Below, we present examples of results for a simulation case where:

- $\mu_{(+)} = (+1; +1; +1)$ and $\mu_{(-)} = (-1; -1; -1)$
- $\boldsymbol{\sigma} = (1, 1, 2)$
- $\boldsymbol{\rho} = (-0.4; 0; 0)$
- $\mathbf{N} = (2, 4, 8, 16, 32, 64, 128, 256, 512, 1024)$

To recap, the first item refers to the centroids of the classes, the second and third items refer to the parameters of the shared covariance matrix between the classes, and the last item refers to the number of observations in each classification instance. Also recall that a simulation case contains several classification instances, and the goal of a simulation case is to evaluate the impact of the number of observations on the error.

### 3.2.1 Graphs

We can visualize the error curves grouped by type of error (Figures 10, 11, and 12) or by the number of features (Figures 13, 14, and 15).

In the graphs where the curves are grouped by type of error, we can observe that:

1. In Figure 10, the theoretical error curves for 2 and 3 features intersect at a single point. This point corresponds to the value of $n^*$ discussed in the previous section. When $n \geq n^*$, it becomes advantageous to use the third feature.
2. In Figure 11, the same observations made above continue to hold when analyzing the empirical test error.
3. In Figure 12, we see that the empirical training error behaves differently from those discussed above. In particular, the empirical training error starts at 0 when we have only one observation on each side of the separating hyperplane. That is, for a dataset with cardinality $n = 2$ (one observation on each side of the separating hyperplane), the empirical training error rate is zero. The error grows as the cardinality $n$ increases, and asymptotically tends to the Bayes error. Note that there is no intersection between the empirical training error curves presented in Figure 12.

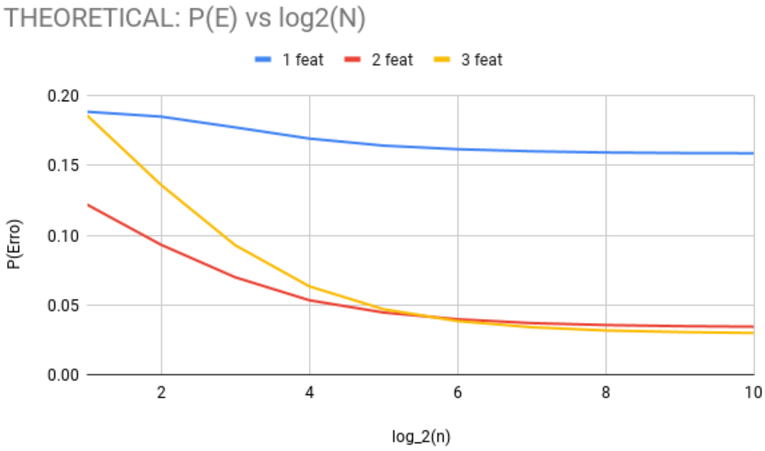


Figura 10 – Theoretical error as a function of the number of observations

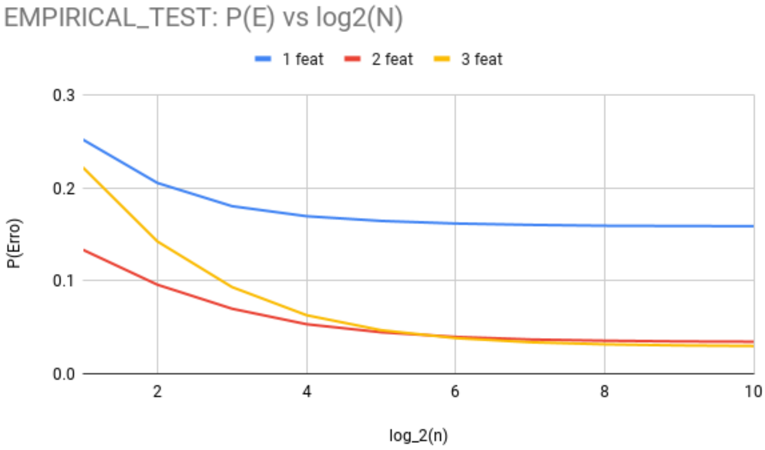


Figura 11 – Empirical test error as a function of the number of observations

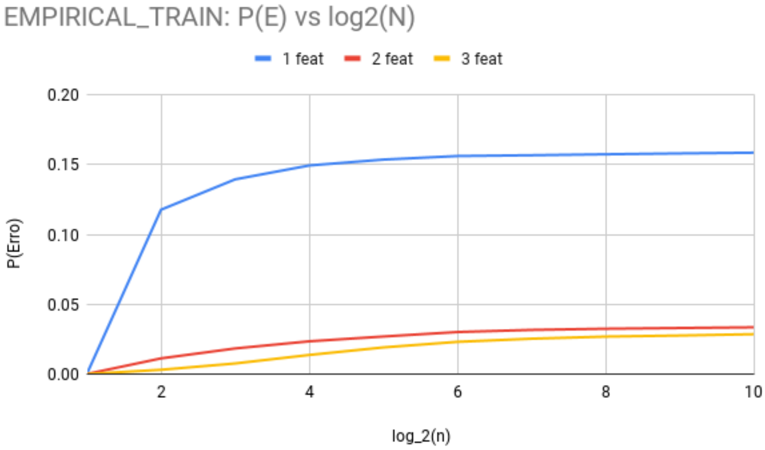


Figura 12 – Empirical training error as a function of the number of observations

Next, we discuss Figures 13, 14, and 15. In these graphs, the curves are grouped by the number of features. We can observe that in all cases, the errors converge to the theoretical Bayes error. Additionally, we note that the empirical test error is always greater than the theoretical error, which in turn is greater than the empirical training error.

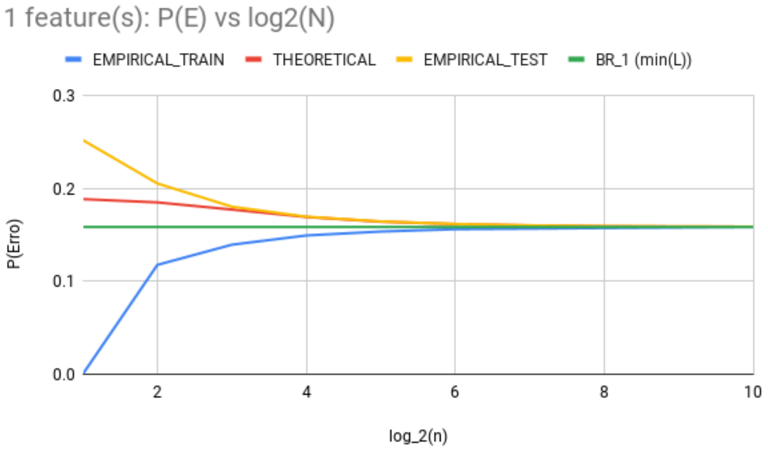


Figura 13 – Error as a function of the number of observations, with 1 feature present

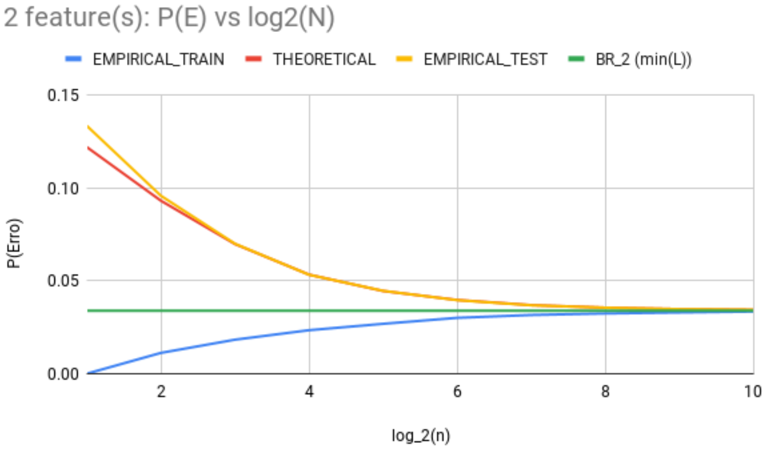


Figura 14 – Error as a function of the number of observations, with 2 features present

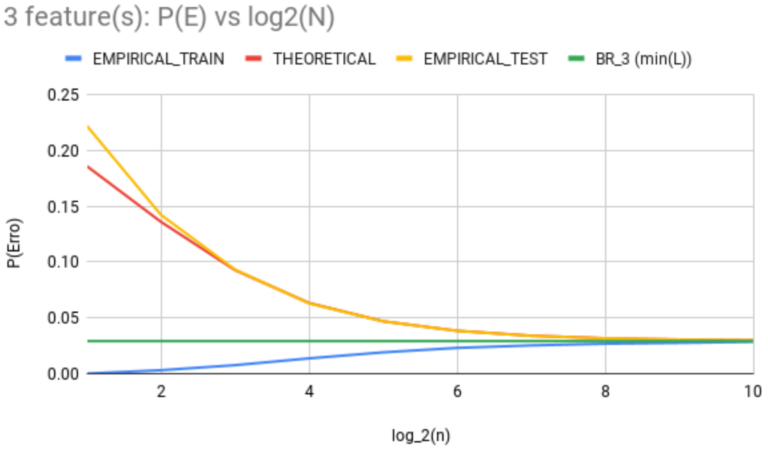


Figura 15 – Error as a function of the number of observations, with 3 features present

### 3.2.2 Visualizations

At the end of a simulation case, we obtain images (*.png*) containing visualizations of the samples considered. In particular, we generate images for all the dimensions and cardinalities simulated, together with graphs containing the error rate curves. Additionally, we also have an animation (*.gif*) containing all the produced images. Below, we see examples of visualizations for the cardinalities $n = 2$, $n = 4$, $n = 64$, and $n = 1024$, in Figures 16, 17, 18, and 19, respectively.

Some observations about the obtained visualizations:

1. At the top of the visualizations, we have the ellipsoids and separating hyperplane, produced as described in Section 2.6.6.

2. Next, we see the 2D projections, illustrating the separation between classes through separating lines.

3. In the row with the graphs titled "Feature 1,Feature 2,"and "Feature 3,"we see the PDFs of the three features considered. The separating line is marked as a black line. As the cardinality $n$ increases, the independent term in the equation of the separating hyperplane (also referred to as bias $b$) approaches zero, and the separating plane approaches the origin, as proposed by the analytical model described in Appendix **??**.

4. Finally, in the last row of the figures, we have the error as a function of the number of observations. For $n = 2$, the graphs are empty. For $n = 4$, the graphs include $n = 2$ and $n = 4$. For $n = 64$, the graphs include $n = 2$, $n = 4$, and $n = 64$, and mark the intersection points between the curves with 2 and 3 features. Finally, for $n = 1024$, we see the complete error curves. This evolution can also be visualized through an animation (animated gif), where the curve is gradually filled in as we consider larger values for $n$.

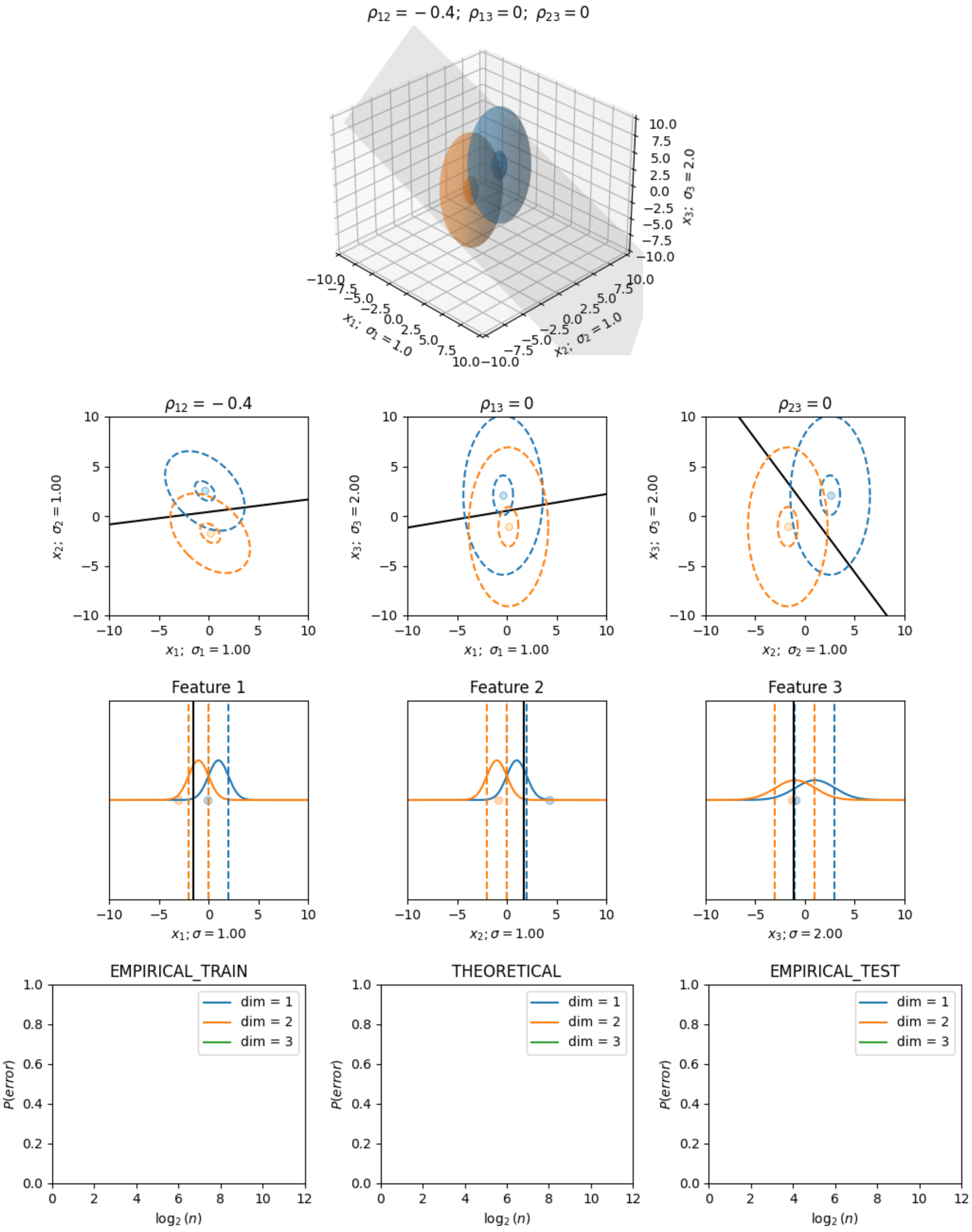


Figura 16 – Visualization of the ellipsoids and separating hyperplane, for $n = 2$

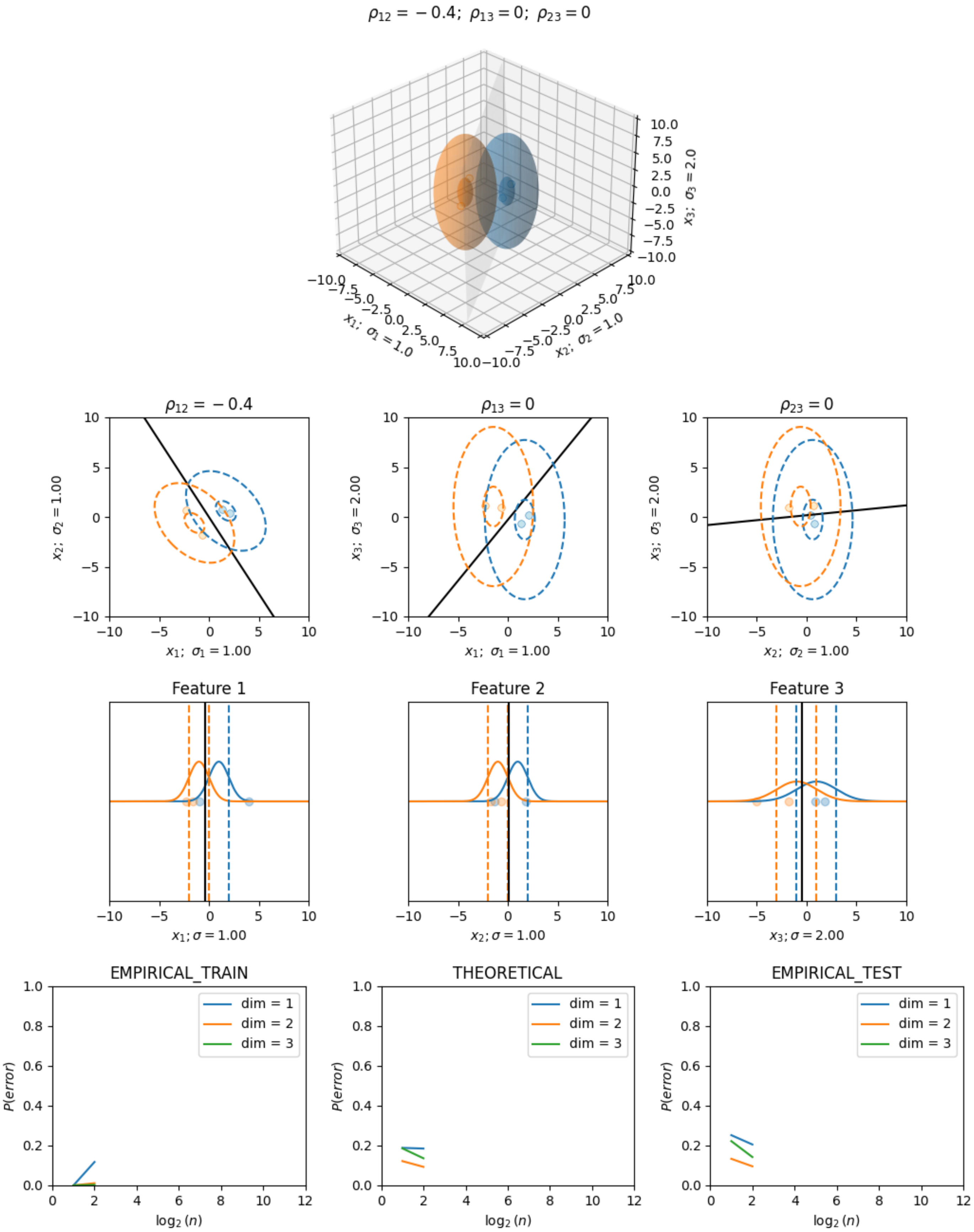


Figura 17 – Visualization of the ellipsoids and separating hyperplane, for $n = 4$

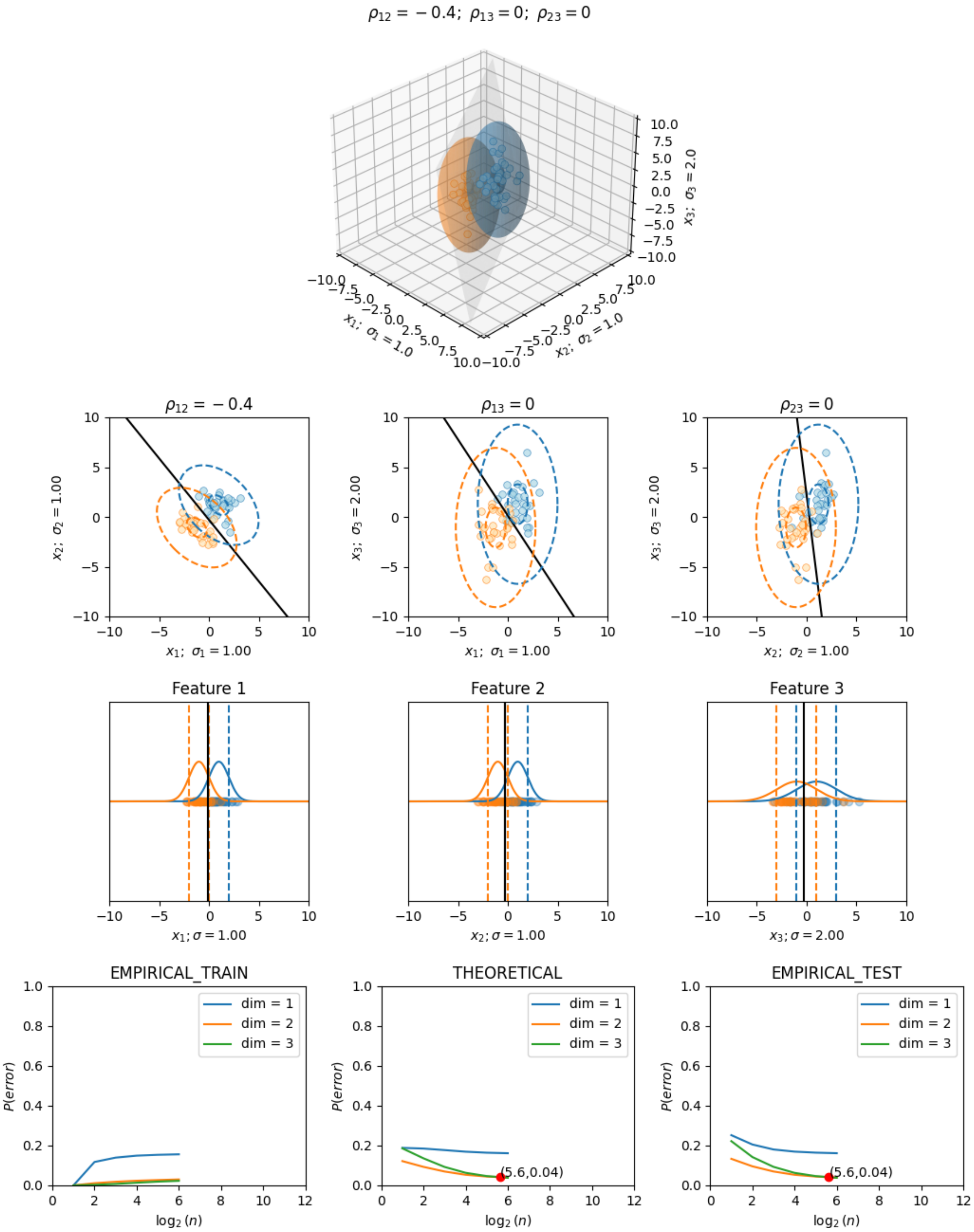


Figura 18 – Visualization of the ellipsoids and separating hyperplane, for $n = 64$

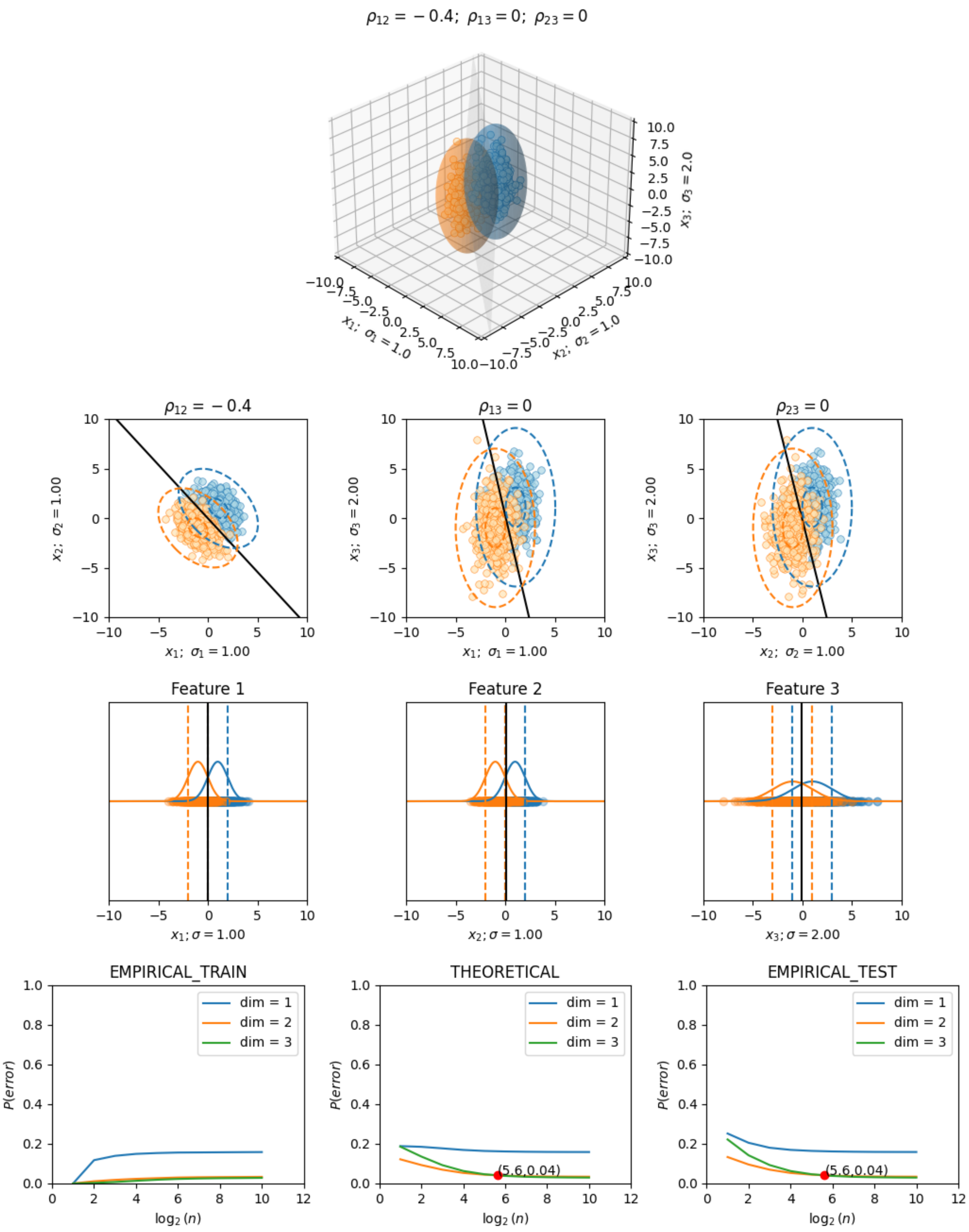


Figura 19 – Visualization of the ellipsoids and separating hyperplane, for $n = 1024$

## 3.3 WHAT IS THE IMPACT OF THE COVARIANCE MATRIX PARAMETERS? REPORT PRODUCED BY A SCENARIO CONTAINING MULTIPLE SIMULATION CASES

Next, we evaluate the impact of $\rho_{12}$ on the classification problem. This evaluation is within the scope of **Scenario 2**.

In particular, we vary $\rho_{12}$ between -0.8 and 0.2, in steps of 0.1, i.e., $\rho_{12} \in \{-0.8, -0.7, \ldots, 0.2\}$.

Here is the link to view, in the form of an animation, the impact of varying $\rho_{12}$ on the classification model and the separating plane: Scenario 2[1]

### 3.3.1 What is the Impact of $\rho_{12}$ on the Usefulness of $X_3$?

We consider the usefulness of feature $X_3$ for classification. For this, we use the quotient $Q$ introduced in (3.2). Empirically, it is found that the larger $Q$, the more useful feature $X_3$ is. However, calculating $Q$ requires simulation rounds. We then ask the following question: is it possible to find an approximation for $Q$ that is easier to obtain and that allows us to determine the usefulness of $X_3$? In Appendix H, we present a proposal, denoted as $d_2/d_3$. Appendix H.4 provides additional discussion on the topic.

Figure 20 illustrates the relationship between $d_2/d_3$ and $Q$, indicating a linear relationship between them.

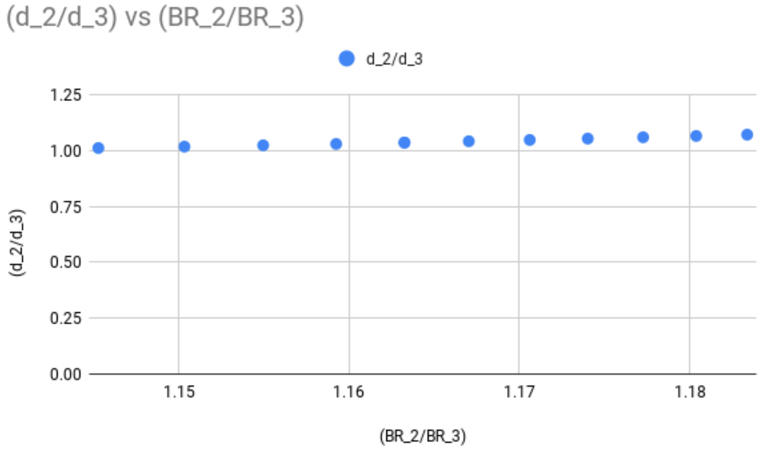


Figura 20 – Relationship between indicators of the usefulness of $X_3$. The indicators are $d_2/d_3$ and $Q$, corresponding to the ratio of Bayes risks with 2 and 3 features. The figure indicates a linear relationship between the two indicators, illustrating another application of the constructed simulator.

[1] <https://drive.google.com/file/d/18xUciYIv-3JguIMDvMbt9XVfJrRpZz0v/view?usp=drive_link>

Figure 21 supports the above argument. It shows the threshold $n^*$ as a function of $\rho_{12}$ in Scenario 2. The higher $\rho_{12}$, the lower $n^*$, and the greater the usefulness of $X_3$.

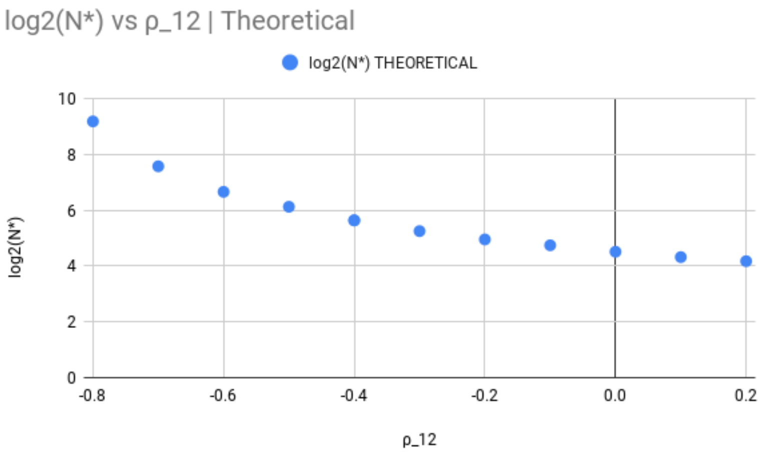


Figura 21 – Threshold $n^*$ as a function of $\rho_{12}$ in Scenario 2: the higher $\rho_{12}$, the lower $n^*$, and the greater the usefulness of $X_3$

### 3.3.2 Additional Animations

Click the links below to access additional animations, illustrating the other scenarios listed in Appendix F.2:

Scenario 1

Scenario 2

Scenario 3

Scenario 4

# 4 RELATED WORK

The analytical models presented in this work were based on the doctoral thesis of Prof. João I. Pinheiro (PINHEIRO, 2021). Our main contribution consists of building a simulator to experimentally investigate how different parameters impact linear classifiers on Gaussian samples. While (PINHEIRO, 2021) focused on the 1D and 2D cases, our work allows extrapolating the results to the 3D case.

## 4.1 LEARNING WITH FEW OBSERVATIONS

Training classifiers with few samples is a topic that has recently gained attention from the remote sensing community (ZHANG et al., 2021; HE et al., 2021), Large Language Models (LLMs) (BROWN et al., 2020), among others (HANCZAR; DOUGHERTY, 2013; MILLER; MATSAKIS; VIOLA, 2000; WANG et al., 2020).

**Detecting Change Points.** Consider a system that implements an algorithm for detecting change points aimed at tracking abrupt changes in the environment. Once a change point occurs, we may have a few samples from each of the classes representing our target of interest. Training a classifier immediately after such a change point is then restricted to small samples (few observations) and a possibly limited number of features (due to sensor network constraints) (NOH; RAJAGOPAL; KIREMIDJIAN, 2013). Although significant effort has already been made to investigate how the number of samples and the number of features affect classifier accuracy (HANCZAR; DOUGHERTY, 2013; NOH; RAJAGOPAL; KIREMIDJIAN, 2013), the relationship between these two elements still presents open analytical and practical problems.

**Bias vs Variance.** Another recent line of research refers to the study of the so-called "double descent"effect (DENG; KAMMOUN; THRAMPOULIDIS, 2021; NAKKIRAN et al., 2020; LOOG; VIERING; MEY, 2019; D'ASCOLI; SAGUN; BIROLI, 2020; BELKIN et al., 2019). The classical literature on the relationship between bias and variance suggests that the error probability decreases as the number of parameters increases and then rises again due to "overfitting."The "double descent"literature extends this idea, pointing out that if the number of parameters continues to increase, the error reaches peaks around the point where the model capacity (e.g., measured by the number of features) approximately equals the number of samples (number of observations), and then it falls again. In other words, when systems become complex enough to only fit the training examples, the error increases (due to "overfitting"), but increasing the number of parameters beyond this limit reduces the error again. The isotropic case with $\rho = 0$ and $\delta = 1$ is discussed in (DENG; KAMMOUN; THRAMPOULIDIS, 2021).

In this work, we consider the non-isotropic case and show that reducing the number of features has a regularization effect, naturally leading to better generalization ability, which is critical when we have a small number of observations.

## 4.2 SIMULATION OF CLASSIFIERS

Similar to what we do here, in (HUA et al., 2005) the authors also deal with binary classification through a simulation-based approach.

Our simulations address specific examples, and we consider what happens to the error probability as the cardinality increases, considering from 1 to 3 features, and Gaussian samples.

On the other hand, in (HUA et al., 2005), the authors consider a much broader combination of probabilistic models and statistical procedures. They show that, under suitable conditions, for each dataset cardinality $n$, there is an optimal number of features to be used in the classifier to minimize its error probability.

We believe that the simulator proposed here could be used to reproduce some of the results presented in (HUA et al., 2005) and extend them to the specific case where the samples are Gaussian, taking into account how different parameters of the Gaussian model impact the error probability. Such analysis is beyond the scope of (HUA et al., 2005).

## 4.3 ERROR PROBABILITY AS A FUNCTION OF THE NUMBER OF FEATURES AND OBSERVATIONS

In (HUGHES, 1968), Hughes analyzes the classification of discrete features, presenting some results similar to (HUA et al., 2005). Instead of considering the dimensionality $d$, referring to the number of features, in (HUGHES, 1968) the concept of "complexity of the measurement pattern"$c$ is introduced, namely the total number of possible values of the feature vector. The author states that for each dataset cardinality $n$, there is an ideal $c$ for which the classifier's error probability is minimized. Thus, by replacing $d$ with $c$, the conclusions of (HUGHES, 1968) and (HUA et al., 2005) are analogous to each other.

Both our work and (HUA et al., 2005) and (HUGHES, 1968) point out that, due to the occurrence of *overfitting*, if the dataset cardinality is relatively small, a high dimensionality can substantially impair the expected classifier performance.

Our work suggests that, in many situations, there may be a trade-off between samples and features regarding minimizing the error probability. In other words, an eventual scarcity of features could be compensated by increasing the sample size. This generally tends to happen when the new features that could eventually be added do not have as high a discriminative power as those already present in the model. On the other hand, an eventual scarcity of samples can also be compensated by adding new features to the dataset. This requires the availability of new features whose inclusion significantly improves the model's overall discriminative power.

# 5 CONCLUSION AND FUTURE WORK

In this work, we considered a binary classification problem involving univariate, bivariate, and trivariate Gaussians. In particular, we presented a simulator to evaluate how the classification error varies as a function of different problem parameters.

Among the applications of the proposed simulator, we analyzed how the cardinality $n$ of the dataset, along with the values of the parameters $\boldsymbol{\sigma}$ and $\boldsymbol{\rho}$, simultaneously influence the decision to use an additional feature in defining the classifier, with the goal of minimizing its error probability. Using the simulator, we verified the existence of a threshold for the cardinality $n$, below which it is preferable to forego an additional feature.

In this work, we explored 1, 2, and 3-dimensional spaces, but with the developed simulator, we can consider higher dimensions. To illustrate such an extension, we produced Figures 22 and 23. In future work, we intend to assess whether our conclusions about the threshold for $n$ can be extended to higher-dimensional spaces. We also intend to experiment with the trade-off between features and samples in a real environment, for example, involving a wireless sensor network in a test setting. In this case, we aim to verify under what conditions it is possible to model the dataset using multivariate Gaussians and subsequently apply the methods presented in this work to develop classifiers for such scenarios.

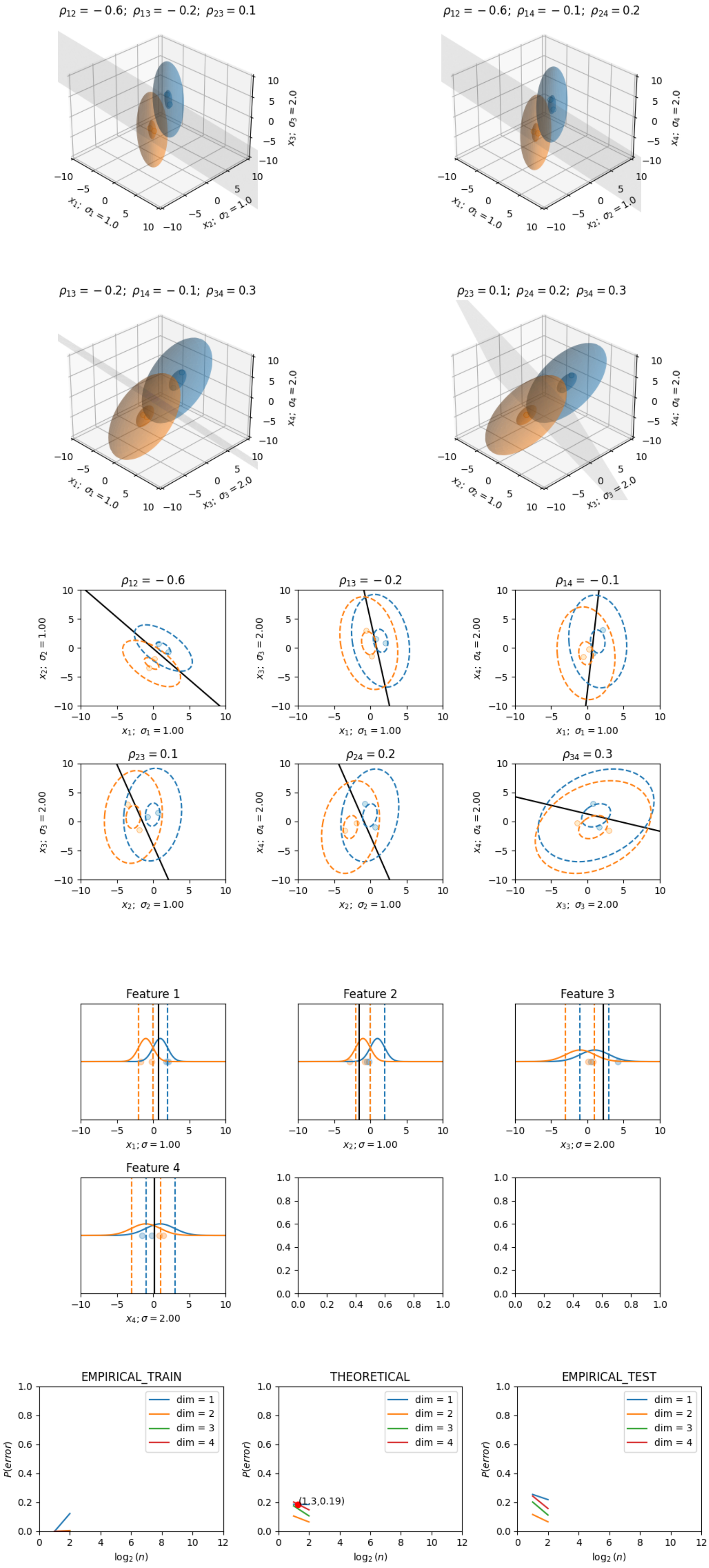


Figura 22 – Example $d > 3$; $\boldsymbol{\sigma} = [1, 1, 2, 2]$; $\boldsymbol{\rho} = [-0.3, -0.2, -0.1, 0.1, 0.2, 0.3]$; $n = 4$

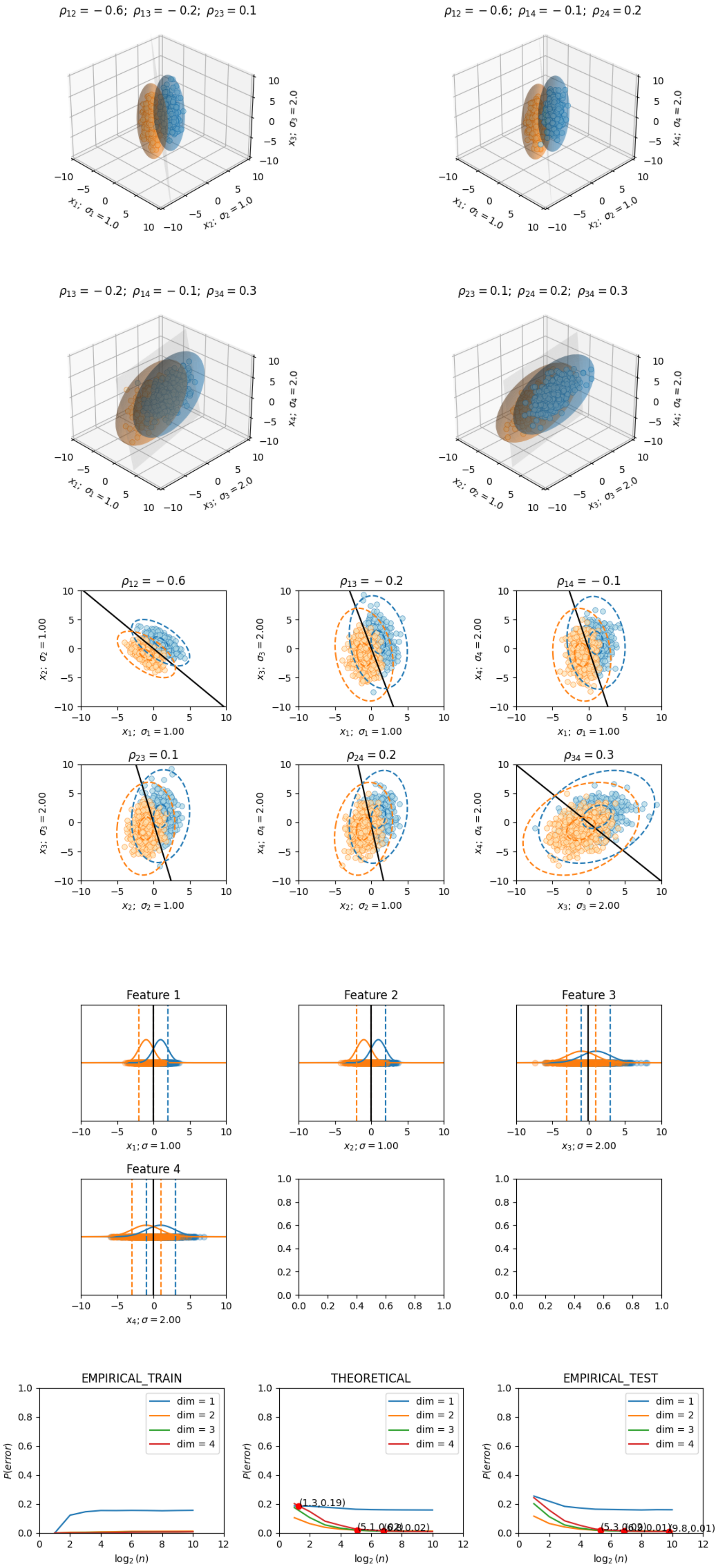


Figura 23 – Example $d > 3$; $\boldsymbol{\sigma} = [1, 1, 2, 2]$; $\boldsymbol{\rho} = [-0.3, -0.2, -0.1, 0.1, 0.2, 0.3]$; $n = 1024$

# REFERÊNCIAS

BELKIN, M. et al. Reconciling modern machine-learning practice and the classical bias–variance trade-off. **Proceedings of the National Academy of Sciences**, National Acad Sciences, v. 116, n. 32, p. 15849–15854, 2019.

BLANCHARD, G.; BOUSQUET, O.; MASSART, P. Statistical performance of support vector machines. **The Annals of Statistics**, Institute of Mathematical Statistics, v. 36, n. 2, p. 489–531, 2008.

BOTTOU, L.; LIN, C.-J. Support vector machine solvers. **Large scale kernel machines**, Citeseer, v. 3, n. 1, p. 301–320, 2007.

BROWN, T. et al. Language models are few-shot learners. **Advances in neural information processing systems**, v. 33, p. 1877–1901, 2020.

D'ASCOLI, S.; SAGUN, L.; BIROLI, G. Triple descent and the two kinds of overfitting: where & why do they appear? In: **NeurIPS**. [S.l.: s.n.], 2020.

DENG, Z.; KAMMOUN, A.; THRAMPOULIDIS, C. A model of double descent for high-dimensional binary linear classification. **Information and Inference: A Journal of the IMA**, 2021.

DEVROYE, L.; GYÖRFI, L.; LUGOSI, G. **A probabilistic theory of pattern recognition**. [S.l.]: Springer Science & Business Media, 2013. v. 31.

ENTEZARI-MALEKI, R.; REZAEI, A.; MINAEI-BIDGOLI, B. Comparison of classification methods based on the type of attributes and sample size. **J. Convergence Inf. Technol.**, Citeseer, v. 4, n. 3, p. 94–102, 2009.

FARAGÓ, A.; LUGOSI, G. Strong universal consistency of neural network classifiers. **IEEE Transactions on Information Theory**, IEEE, v. 39, n. 4, p. 1146–1151, 1993.

GLASMACHERS, T. Universal consistency of multi-class support vector classification. **Advances in Neural Information Processing Systems**, v. 23, p. 739–747, 2010.

HANCZAR, B.; DOUGHERTY, E. R. The reliability of estimated confidence intervals for classification error rates when only a single sample is available. **Pattern Recognition**, Elsevier, v. 46, n. 3, p. 1067–1077, 2013.

HE, F. et al. One-Shot Distributed Algorithm for PCA With RBF Kernels. **IEEE Signal Processing Letters**, IEEE, v. 28, p. 1465–1469, 2021.

HSIEH, C.-J. et al. A dual coordinate descent method for large-scale linear SVM. In: **Proceedings of the 25th international conference on Machine learning**. [S.l.: s.n.], 2008. p. 408–415.

HUA, J. et al. Optimal number of features as a function of sample size for various classification rules. **Bioinformatics**, Oxford University Press, v. 21, n. 8, p. 1509–1515, 2005.

HUGHES, G. On the mean accuracy of statistical pattern recognizers. **IEEE Transactions on Information Theory**, IEEE, v. 14, n. 1, p. 55–63, 1968.

LIST, N.; SIMON, H. U. SVM-optimization and steepest-descent line search. In: CITESEER. **Proceedings of the 22nd Annual Conference on Computational Learning Theory**. [S.l.], 2009.

LOOG, M.; VIERING, T.; MEY, A. Minimizers of the empirical risk and risk monotonicity. **Advances in Neural Information Processing Systems**, v. 32, p. 7478–7487, 2019.

MILLER, E. G.; MATSAKIS, N. E.; VIOLA, P. A. Learning from one example through shared densities on transforms. In: IEEE. **Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)**. [S.l.], 2000. v. 1, p. 464–471.

NAKKIRAN, P. et al. Optimal regularization can mitigate double descent. In: **International Conference on Learning Representations**. [S.l.: s.n.], 2020.

NOH, H.; RAJAGOPAL, R.; KIREMIDJIAN, A. Sequential structural damage diagnosis algorithm using a change point detection method. **Journal of Sound and Vibration**, Elsevier, v. 332, n. 24, p. 6419–6433, 2013.

PINHEIRO, J. I. D. Aprendendo a classificar com poucos atributos e amostras. **Tese de doutorado**, 2021. Universidade Federal do Rio de Janeiro.

R Documentation. 2021. <https://www.rdocumentation.org/packages/e1071>.

VERT, R.; VERT, J.-P.; SCHÖLKOPF, B. Consistency and convergence rates of one-class SVMs and related algorithms. **Journal of Machine Learning Research**, v. 7, n. 5, 2006.

WANG, Y. et al. Generalizing from a few examples: A survey on few-shot learning. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 53, n. 3, p. 1–34, 2020.

WILLETT, P.; SWASZEK, P. F.; BLUM, R. S. The good, bad and ugly: distributed detection of a known signal in dependent gaussian noise. **IEEE Transactions on signal processing**, IEEE, v. 48, n. 12, p. 3266–3279, 2000.

ZHANG, S. et al. Polygon structure-guided hyperspectral image classification with single sample for strong geometric characteristics scenes. **IEEE Transactions on Geoscience and Remote Sensing**, IEEE, 2021.

# A PROBLEM CONTEXTUALIZATION AND FORMALIZATION

## A.1 PROBLEM FORMULATION

The classification process involves two stages: training the classifier with labeled samples and then inferring the classes of unlabeled samples. Our two main variables of interest are the dataset cardinality, denoted by $n$, which corresponds to the number of samples in the training set, and the dataset dimensionality, denoted by $d$, which corresponds to the number of features in each sample. One of our objectives is to evaluate how $n$ and $d$ simultaneously affect the classifier's performance, as measured by its expected error probability.

### A.1.1 Context

We consider a binary classification problem, where classes are denoted by labels -1 and +1. The dataset $\mathcal{D}$ contains $n$ ordered pairs $(\boldsymbol{x}_i, y_i)$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, for $i = 1, \ldots, n$. The dataset $\mathcal{D}$ is used to train a classifier that minimizes the error probability. Let $X$ and $Y$ be the random variables corresponding to the feature vector and target class, respectively, assuming the joint probability distribution for the pair $(X, Y)$ is unknown.

A classifier is a function $h : \boldsymbol{x} \to h(\boldsymbol{x})$ that assigns a label $h(\boldsymbol{x}) \in \{-1, +1\}$ to any feature vector $\boldsymbol{x} \in \mathbb{R}^d$. The classifier's error probability (or loss) is denoted by $L(h)$ and is given by $L(h) = P(Y \neq h(X))$. A dictionary $H$ (also known as a hypothesis set or search space bias) is a set of classifiers (e.g., linear, quadratic, or rectangular classifiers). The learning problem is to choose a classifier from the dictionary $H$ as the "solution"to our classification problem. It is well known that the Bayes classifier, which classifies each input according to $\text{argmax}_y P(Y = y|X = x)$, is globally optimal (DEVROYE; GYÖRFI; LUGOSI, 2013; FARAGÓ; LUGOSI, 1993); the Bayes error is the minimum error probability. We denote the Bayes classifier and its error probability by $h^{(B)}$ and $L(h^{(B)})$, or $h^*$ and $L(h^*)$.

### A.1.2 How to Evaluate the Efficiency of a Classifier

What should be our goal when facing a binary classification problem? Common sense suggests that we should look for a classifier with the lowest possible error rate. Suppose we have a real dataset $D$ on a given topic. In that case, what is the error rate that really matters? Let's assume that the available data is used to calibrate a classifier whose empirical error rate is as low as possible when applied to the dataset $D$. Then, of course, the best way to evaluate the efficiency of this classifier would be to predict its empirical error rate whenever applied to another independent dataset $D'$. If we are lucky enough to obtain a dataset with many observations, the usual solution would be to randomly split the available dataset into training and test sets. This would avoid underestimating the true error rate by using the same data to calibrate the classifier and also evaluate its effectiveness. In this context, it is important to distinguish between the empirical and theoretical error rates associated with a given classifier: the empirical rate is the training rate (which tends to underestimate the true error rate), while the test rate provides a more reliable estimate of its expected error rate. Furthermore, if it is not possible to obtain a dataset with many observations, a resampling technique (such as cross-validation, bootstrap, etc.) could be used to avoid such bias in estimating classification error rates. Now, what if we work with simulated data coming from a known probabilistic model? In this case, for a given classifier, it is always possible to use probability theory to calculate (eventually using numerical methods) its theoretical error rate without having to evaluate its effectiveness through test data. As this is a methodological work, we chose to work with simulated data, which greatly facilitates the task of evaluating the effectiveness of each classifier impartially. Fortunately, conclusions based on this type of approach are also applicable to concrete situations where we work with real data. On the other hand, when facing a binary classification problem, what are our "control knobs"? Among them, we can mention:

- The dataset cardinality $n$, i.e., the number of available observations.
- The problem dimensionality $d$, i.e., the number of features available in each observation.
- The discriminative power of each feature, alone or in the presence of other features.
- Dictionary $H$, from which we will choose our classifier.

It is worth asking how a particular choice of combination of these input parameters will affect the expected classification error rate in our mathematical model.

# B BIVARIATE GAUSSIAN AND SVM CLASSIFIER

To evaluate the generalization power of the classifiers considered, we have at least three alternatives:

1. Sample $D$ from a known probabilistic model, as in (NAKKIRAN et al., 2020; D'ASCOLI; SAGUN; BIROLI, 2020), and use the training sample $D$ itself to evaluate the empirical classification performance of $h$, as in (C.1).
2. Evaluate the performance of $h$ using equations from the probabilistic linear model like (D.1), (D.6), or (D.28).
3. Evaluate the performance of $h$ on a test dataset $D'$, generated from the same probabilistic model used to sample $D$.

## B.1 BIVARIATE AND TRIVARIATE GAUSSIANS CONSIDERED IN THIS WORK

In particular, consider a two-dimensional classification problem ($d = 2$) or a three-dimensional problem ($d = 3$), in which samples from each class are drawn from a bivariate or trivariate Gaussian distribution. We assume equal priors for the two classes, i.e., $P(Y = +1) = P(Y = -1) = 0.5$.

### B.1.1 Bivariate Case

The conditional distribution of the feature vector $X = (X_1, X_2)$ given $Y$ is given by

$$(X_1, X_2) \sim \begin{cases} \mathcal{N}((+1, +1), \Sigma), & \text{if } Y = +1, \\ \mathcal{N}((-1, -1), \Sigma), & \text{if } Y = -1, \end{cases} \tag{B.1}$$

where $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ denotes a bivariate Gaussian distribution with mean vector $\boldsymbol{\mu} \in \mathbb{R}^2$ and covariance matrix $\Sigma$,

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} \tag{B.2}$$

with $\sigma_1 = 1$, $\sigma_2 \geq 1$ and $|\rho| \leq 1$. Note that $\sigma_2$ is the conditional standard deviation of feature $X_2$, given $Y$. If $\sigma_2 > 1$, the feature $X_2$ has less discriminative power than feature $X_1$: the larger the value of $\sigma_2$, the less informative feature $X_2$ is.

Note that we consider balanced datasets, with the same number of samples in each class.

### B.1.2 Trivariate Case

In the three-dimensional case, we have as parameters $\sigma_1, \sigma_2, \sigma_3, \rho_{12}, \rho_{13}, \rho_{23}$ that constitute the covariance matrix,

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 & \rho_{13}\sigma_1\sigma_3 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 & \rho_{23}\sigma_2\sigma_3 \\ \rho_{13}\sigma_1\sigma_3 & \rho_{23}\sigma_2\sigma_3 & \sigma_3^2 \end{bmatrix}.$$

Unless stated otherwise, we focus on the two-dimensional and three-dimensional cases in this work.

## B.2 CLASSIFIER **SVM**

Our dictionary $H$ consists of all linear classifiers, that is, straight lines. A naive approach to learning involves finding the line that minimizes the empirical error rate, for example, through a grid search. However, the problem may admit multiple solutions, requiring a well-founded strategy to break ties (BLANCHARD; BOUSQUET; MASSART, 2008). Therefore, we consider linear support vector machines (**SVM**), which have a series of desirable properties, including:

1. a well-founded strategy to determine the best separator, maximizing the distance, known as the *margin*, between the separating line and the two classes of points to be separated (BLANCHARD; BOUSQUET; MASSART, 2008);

2. polynomial computational cost (BOTTOU; LIN, 2007; LIST; SIMON, 2009), and

3. convergence to an optimal solution under mild conditions (HSIEH et al., 2008; GLASMACHERS, 2010; VERT; VERT; SCHÖLKOPF, 2006).

The linear **SVM** depends on a single parameter, the cost of incorrect classification, which appears as a regularization term in the Lagrange formulation and is denoted by $\gamma$ (R..., 2021). Throughout this work, we keep $\gamma = 1$, its default value; we experimented with other values, but the results remained approximately the same.

# C TYPES OF ERRORS

Next, we consider the types of errors, also called loss functions. In particular, we are interested in understanding how:

- the classifier's accuracy increases as the number of samples increases,
- the accuracy of error measurement improves as the number of samples increases.

Another aspect that should not be neglected is the precision with which we can estimate the error rate of our classification procedure. And, of course, it is also interesting to investigate how each input parameter affects this accuracy. This is what we will discuss now. In particular, we will investigate what happens to the accuracy of the error rate estimate as the cardinality of the dataset $n$ tends to infinity.

Thus, from now on, let $H$ be a specific dictionary. Furthermore, suppose we have a dataset $D$, which is representative of the phenomenon under study and consists of pairs $(x_i, y_i)$, $i = 1, 2, ..., n$, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, for all $i = 1, 2, ..., n$.

Then, for each classifier $h$ in $H$, we can calculate its empirical error rate $\hat{L}(h)$, that is, the proportion of misclassified observations in the dataset $D$:

$$\hat{L}(h) = \frac{1}{n} \sum_{i=1}^{n} 1(y_i \neq h(x_i)) \tag{C.1}$$

Let $\hat{h}^{(D)}$ be the best empirical classifier in the dictionary $H$, that is, the classifier in $H$ with the minimum empirical error rate. Under these conditions, we can define four error rates:

- $\min_{h \in H} L(h)$ = the lowest **Theoretical Error** rate achievable in $H$
- $L(\hat{h}^{(D)})$ = the theoretical error rate of the classifier $\hat{h}^{(D)}$ (eq. (2.4))
- $\hat{L}(\hat{h}^{(D)})$ = the empirical error rate of the classifier $\hat{h}^{(D)}$ (eq. 2.1)
- $\hat{L}(\hat{h}^{(D)}, D')$ = the error rate of the classifier $\hat{h}^{(D)}$ using a test dataset $D'$ (eq. 2.2)

It is known that these losses necessarily obey an order, for any size of $D$:

$$0 \leq \hat{L}(\hat{h}^{(D)}) \leq \min_{h \in H} L(h) \leq L(\hat{h}^{(D)}),$$

From the results of our experiment, we observe that, for a very large dataset ($n \to \infty$):

$$0 \leq \hat{L}(\hat{h}^{(D)}) \leq \min_{h \in H} L(h) \leq \hat{L}(\hat{h}^{(D)}, D') \leq L(\hat{h}^{(D)});$$

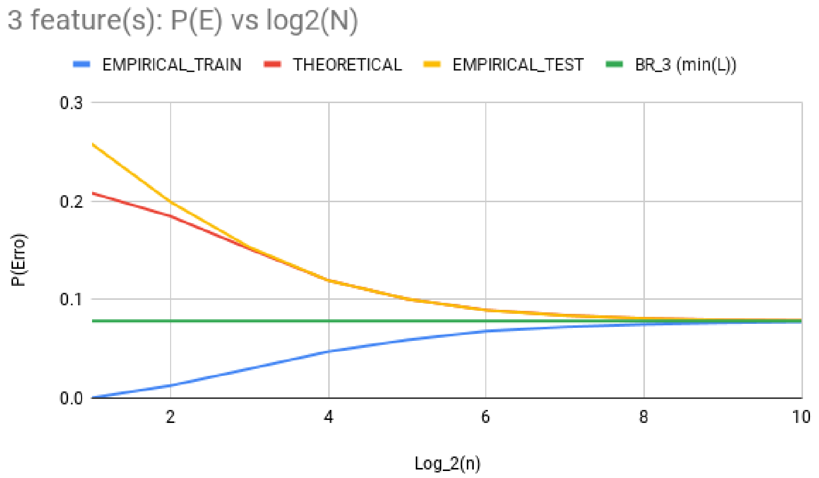


Figura 24 – Types of errors; $\boldsymbol{\sigma} = [1, 1, 2]$ and $\boldsymbol{\rho} = [0, 0.3, 0.3]$

For smaller datasets, the theoretical error rate is lower than the empirical error rate using a distinct test dataset $D'$:

$$0 \leq \hat{L}(\hat{h}^{(D)}) \leq \min_{h \in H} L(h) \leq \hat{L}(\hat{h}^{(D)}, D') \leq L(\hat{h}^{(D)}).$$

$\min_{h \in H} L(h)$ is also referred to in this work as the Bayes error rate (also known as Bayesian risk), which is the minimum theoretical loss generally achievable for this particular classification problem, detailed in appendix D.

In Figure 24, we have an example of a graph produced by simulating a case. This graph ensures the correctness of the simulator concerning the calculation of Bayes risk, performed analytically, as proposed by the studies of Professor João I. Pinheiro. We can observe that as the number of observations increases, the loss function curves approach Bayes risk.

# D ANALYTICAL MODEL FOR THE MINIMUM ACHIEVABLE THEORETICAL CLASSIFICATION ERROR: SPECIAL CASES

Next, we consider the Bayes Risk of a classifier for a sample of infinite size ($n = \infty$). Specifically, we focus on deriving an expression for the minimum theoretical error achievable by a classifier in the 1D, 2D, and 3D cases.

## D.1 1D: ONE ATTRIBUTE

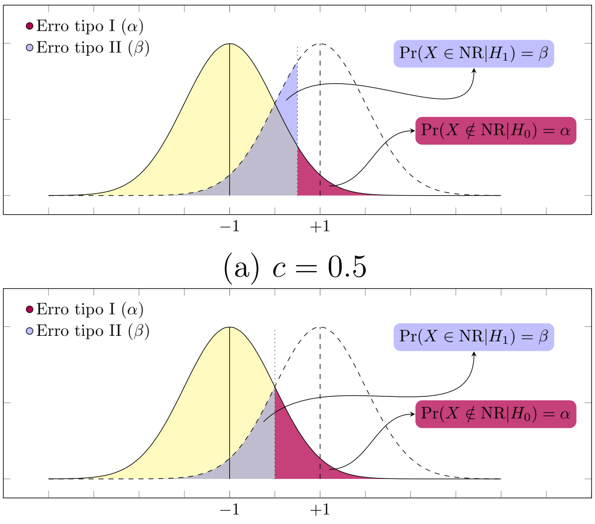
We begin by considering the case where only the attribute $x_1$ is available. We have a one-dimensional classification problem, in which we seek the best cutoff point $c$. The probability of error, in the case where the samples of the two classes are given by Gaussian distributions with variance 1 and means -1 and +1, is given by:

$$L_1(c) = \frac{1}{2}\Phi(c-1) + \frac{1}{2}\left(1 - \Phi(c+1)\right) = \frac{1}{2}(1 + \Phi(c-1) - \Phi(c+1)) \tag{D.1}$$

where $\Phi(\cdot)$ is the cumulative distribution function of a Gaussian with mean 0 and variance 1. The optimal separator is $c^* = 0$, and

$$L_1(0) = 1 - \Phi(1) = 0.1586553. \tag{D.2}$$

Below, we present the derivation of (D.1). To do so, Figure 25 illustrates the concepts above using hypothesis testing terminology. The null hypothesis $H_0$ corresponds to the



(a) $c = 0.5$

(b) $c = 0$

Figura 25 – The non-rejection region of $H_0$ is NR, where $\alpha$ is the level of significance, i.e., the probability of incorrectly rejecting the null hypothesis $H_0$. $1 - \beta$ is the power of the test, i.e., the probability of correctly rejecting the null hypothesis $H_0$.

sample $X$ belonging to class -1, i.e., $X \sim \mathcal{N}(-1, 1)$, while the alternative hypothesis $H_1$ corresponds to the sample $X$ belonging to class +1, i.e., $X \sim \mathcal{N}(+1, 1)$. The non-rejection region of $H_0$ is denoted by NR. The probability of error is given by:

$$\begin{aligned} L_1(c) &= P(H_1)P(X \in NR|H_1) + P(H_0)P(X \notin NR|H_0) = && \text{(D.3)} \\ &= \frac{1}{2}\beta + \frac{1}{2}\alpha = && \text{(D.4)} \\ &= \frac{1}{2}\left(\Phi(c-1) + (1 - \Phi(c+1))\right). && \text{(D.5)} \end{aligned}$$

Indeed, observing Figure 25, we see that the blue region, to the left of $c$, with probability $\beta$, is such that:

$$P(X \in NR|H_1) = \beta = \Phi(c-1).$$

Similarly, we see that the red region, to the right of $c$, with probability $\alpha$, is such that:

$$P(X \notin NR|H_0) = \alpha = 1 - \Phi(c+1).$$

## D.2 2D: TWO ATTRIBUTES

### D.2.1 Error Expression

Next, we consider the case where two attributes are available. Given a classifier characterized by the line $x_2 = a + bx_1$:

$$L_2(a, b) = \frac{1}{2}\left(1 - \Phi\left(\frac{|-a-b+1|}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{|a-b+1|}{\sqrt{\Delta}}\right)\right). \tag{D.6}$$

The above error expression considers two classes, with means $\mu_+ = (+1, +1)$ and $\mu_- = (-1, -1)$.

### D.2.2 Relation Between (D.6) and Section 2.4.2

Next, we relate the error expression (D.6) with the terminology from Section 2.4.2. We have:

$$bx_1 - x_2 + a = 0$$

or equivalently,

$$w_1x_1 + w_2x_2 - \tilde{b} = 0.$$

Thus,

$$\tilde{b} = -a, \quad \kappa = -a$$

and

$$w_1 = -b/(-1) = b, \quad w_2 = -1.$$

The distance calculations are as follows:

$$\text{Distance}(+) = \frac{\left|w_1 - 1 - \tilde{b}\right|}{\sqrt{\Delta}} = \frac{|b - 1 + a|}{\sqrt{\Delta}}, \quad \text{Distance}(-) = \frac{\left|w_1 - 1 + \tilde{b}\right|}{\sqrt{\Delta}} = \frac{|b - 1 - a|}{\sqrt{\Delta}}.$$

Thus, we can express the **Theoretical Error** as a function of the weight vector **w** and the bias $b$:

$$L_d(\mathbf{w}, b) = \frac{1}{2}\left(1 - \Phi\left(\frac{|\sum_{i=1}^{i=d} w_i - \tilde{b}|}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{|\sum_{i=1}^{i=d} w_i + \tilde{b}|}{\sqrt{\Delta}}\right)\right) \tag{D.7}$$

Note that

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} = \boldsymbol{\lambda}\boldsymbol{\lambda}^T.$$

$$\boldsymbol{\lambda} = \begin{bmatrix} \sigma_1 & 0 \\ \frac{\rho\sigma_1\sigma_2}{\sigma_1} & \sqrt{\sigma_2^2 - (\rho\sigma_1\sigma_2/\sigma_1)^2} \end{bmatrix}, \quad \boldsymbol{\lambda}^T = \begin{bmatrix} \sigma_1 & \frac{\rho\sigma_1\sigma_2}{\sigma_1} \\ 0 & \sqrt{\sigma_2^2 - (\rho\sigma_1\sigma_2/\sigma_1)^2} \end{bmatrix}.$$

$$\tilde{\boldsymbol{\delta}} = \boldsymbol{\lambda}^T \cdot \begin{bmatrix} b \\ -1 \end{bmatrix} = \begin{bmatrix} \tilde{\delta}_1 \\ \tilde{\delta}_2 \end{bmatrix}.$$

Thus,

$$\Delta = \tilde{\delta}_1^2 + \tilde{\delta}_2^2$$

or equivalently,

$$\Delta = (-b\sigma_1 + \rho\sigma_2)^2 + \sigma_2^2 - (\rho\sigma_2)^2. \tag{D.8}$$

#### D.2.2.1 Probability of Gaussian Generating a Point Above a Given Line

Next, we seek $P(X_2 > bX_1 + a)$ where $(X_1, X_2) \sim N(\mu_1, \mu_2, \Sigma)$.

**Teorema 1.** *If* $(X_1, X_2) \sim N(\mu_1, \mu_2, \Sigma)$*, then*

$$P(X_2 > bX_1 + a) = P(Z_2 > z_2),$$

*where* $Z_2 \sim N(0, 1)$ *and*

$$z_2 = \frac{b\mu_1 + a - \mu_2}{\sqrt{(\sigma_2\rho - b\sigma_1)^2 + \sigma_2^2(1-\rho^2)}} = \frac{b\mu_1 + a - \mu_2}{\sqrt{\Delta}}.$$

The proof proceeds in four steps:
1. **First Step:** Let $(X_1, X_2) \sim N(\mu_1, \mu_2, \Sigma)$ where

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}.$$

Then,

$$X_1 = \mu_1 + \sigma_1 X_1', \quad X_2 = \mu_2 + \sigma_2 X_2'$$

where $X_1', X_2' \sim N(0,1)$ with correlation coefficient $\rho$.
2. **Second Step:**

$$X_2' = \rho X_1' + \sqrt{1-\rho^2} Z$$

where $Z \sim N(0,1)$ is independent of $X_1'$.
3. **Third Step:** Write the event $X_2 > bX_1 + a$ in the form:

$$X_1' \geq \beta Z + \alpha.$$

In detail,

$$X_2 > bX_1 + a \tag{D.9}$$
$$\mu_2 + \sigma_2\left(\rho X_1' + \sqrt{1-\rho^2}Z\right) > b(\mu_1 + \sigma_1 X_1') + a \tag{D.10}$$
$$\sigma_2\rho X_1' > b(\mu_1 + \sigma_1 X_1') + a - \mu_2 - \sigma_2\sqrt{1-\rho^2}Z \tag{D.11}$$
$$(\sigma_2\rho - b\sigma_1)X_1' > b\mu_1 + a - \mu_2 - \sigma_2\sqrt{1-\rho^2}Z \tag{D.12}$$

Thus, if $\sigma_2\rho - b\sigma_1 > 0$:

$$\alpha = \frac{b\mu_1 + a - \mu_2}{\sigma_2\rho - b\sigma_1}, \quad \beta = \frac{-\sigma_2\sqrt{1-\rho^2}}{\sigma_2\rho - b\sigma_1}.$$

4. **Fourth Step:**

$$Z_1 = \frac{X_1' - \beta Z}{\sqrt{1+\beta^2}}, \quad z_1 = \frac{\alpha}{\sqrt{1+\beta^2}}.$$

Note that $Z_1 \sim N(0,1)$ and:

$$P(X_2 > bX_1 + a) = P(Z_1 > z_1).$$

The reasoning above holds if $\sigma_2\rho - b\sigma_1 > 0$. In this case:

$$P(X_2 > bX_1 + a) = P(Z_1 > z_1)$$

where:

$$z_1 = \frac{\alpha}{\sqrt{1+\beta^2}},$$

and:

$$\alpha = \frac{b\mu_1 + a - \mu_2}{\sigma_2\rho - b\sigma_1}, \quad \beta = \frac{-\sigma_2\sqrt{1-\rho^2}}{\sigma_2\rho - b\sigma_1}.$$

If $\sigma_2\rho - b\sigma_1 < 0$, then:

$$P(X_2 > bX_1 + a) = P(Z_1 < z_1),$$

and the final result still holds because:

$$P(Z_1 < z_1) = P(Z_1 > -z_1).$$

In this case, where $\sigma_2\rho - b\sigma_1 < 0$, we have:

$$z_2 = -z_1 = \frac{b\mu_1 + a - \mu_2}{\sqrt{(\sigma_2\rho - b\sigma_1)^2 + \sigma_2^2(1-\rho^2)}}.$$

An intuitive way to understand the proof above is to visualize the original Gaussian, the normalized Gaussian, and the normalized Gaussian with the error region rotated. This visualization is presented in Figure 26.

#### D.2.2.2 Special Case 1: Proof of (D.6)

Recall that in this work, we consider two classes. In class 1, we have $\mu_1 = \mu_2 = 1$, and in class 2, we have $\mu_1 = \mu_2 = -1$. Thus, $-b + a + 1$ and $b + a - 1$ have opposite signs.

**Case a)** $b \leq 1$, i.e., $b + a - 1 \leq 0 \leq -b + a + 1$.

In this case, applying Theorem 1 for class 2:

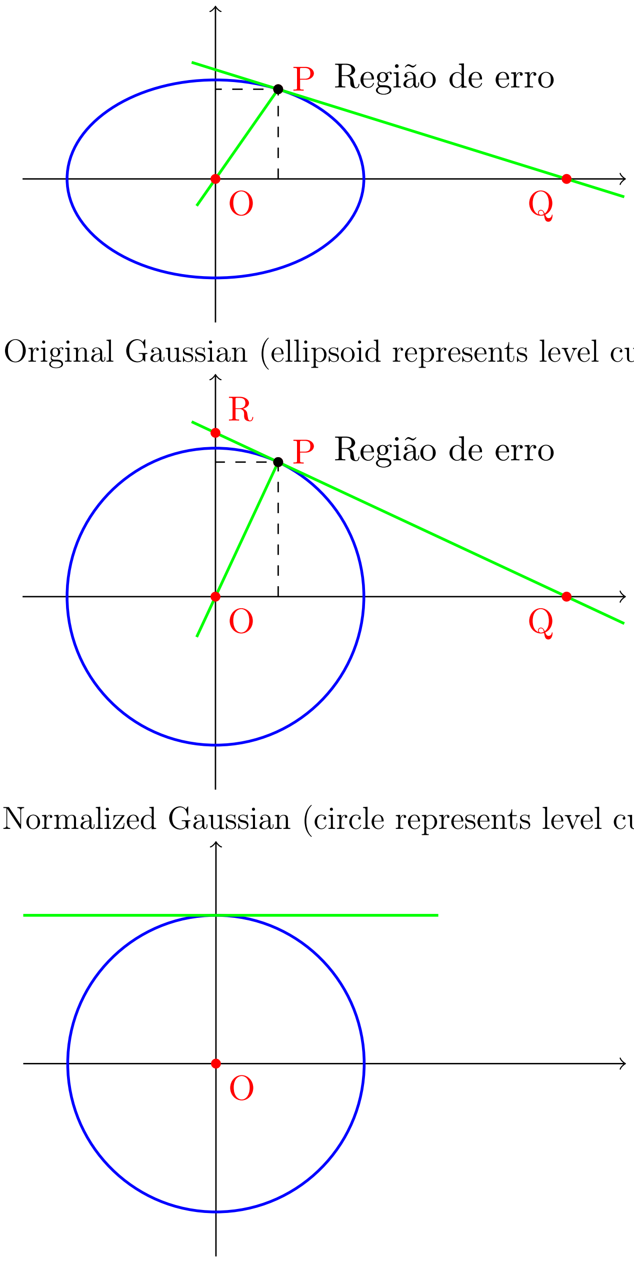
$$z_2 = \frac{b\mu_1 + a - \mu_2}{\sqrt{\Delta}} = \frac{-b + a + 1}{\sqrt{\Delta}} = \frac{|-b + a + 1|}{\sqrt{\Delta}}.$$

Thus:

$$P(\text{error}|-) = P(Z_2 > z_2) = 1 - P(Z_2 < z_2) = 1 - P\left(Z_2 < \frac{|-b+a+1|}{\sqrt{\Delta}}\right)$$

as expected (rightmost term in (D.6)).

Next, for class 1, by Theorem 1:



(a) Original Gaussian (ellipsoid represents level curve)

(b) Normalized Gaussian (circle represents level curve)

(c) Normalized Gaussian with rotated error region

Figura 26 – Finding the **Theoretical Error** for a bidimensional linear classifier

$$z_2 = \frac{b\mu_1 + a - \mu_2}{\sqrt{\Delta}} = \frac{b + a - 1}{\sqrt{\Delta}} = -\frac{|b + a - 1|}{\sqrt{\Delta}} = -\frac{|-b-a+1|}{\sqrt{\Delta}}.$$

The rest of the derivation follows similarly:

$$P(\text{error}|+) = 1 - P(Z_2 > z_2) = 1 - P(Z_2 < -z_2) = 1 - P\left(Z_2 < \frac{|-b-a+1|}{\sqrt{\Delta}}\right)$$

as expected (leftmost term in (D.6)).

In conclusion:

$$\begin{aligned} L_2(a,b) &= \frac{1}{2}P\left(X_2^{(+)} < bX_1^{(+)} + a\right) + \frac{1}{2}P\left(X_2^{(-)} > bX_1^{(-)} + a\right) && \text{(D.13)}\\ &= \frac{1}{2}\left(1 - P\left(X_2^{(+)} > bX_1^{(+)} + a\right)\right) + \frac{1}{2}P\left(X_2^{(-)} > bX_1^{(-)} + a\right) && \text{(D.14)}\\ &= \frac{1}{2}\left(1 - \Phi\left(\frac{|-b-a+1|}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{|a-b+1|}{\sqrt{\Delta}}\right)\right) && \text{(D.15)}\\ &= \frac{1}{2}\left(1 - \Phi\left(\frac{-a-b+1}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{a-b+1}{\sqrt{\Delta}}\right)\right) && \text{(D.16)} \end{aligned}$$

The above expression matches the one we aimed to derive, i.e., (D.6).

**Case b)** $b \geq 1$, i.e., $b + a - 1 \geq 0 \geq -b + a + 1$.

For class 2, by Theorem 1:

$$z_2 = \frac{b\mu_1 + a - \mu_2}{\sqrt{\Delta}} = \frac{-b + a + 1}{\sqrt{\Delta}} = -\frac{|-b+a+1|}{\sqrt{\Delta}}.$$

Thus:

$$P(\text{error}|-) = 1 - P(Z_2 > z_2) = 1 - P(Z_2 < -z_2) = 1 - P\left(Z_2 < \frac{|-b+a+1|}{\sqrt{\Delta}}\right)$$

as expected (leftmost term in (D.6)).

Next, for class 1, by Theorem 1:

$$z_2 = \frac{b\mu_1 + a - \mu_2}{\sqrt{\Delta}} = \frac{b + a - 1}{\sqrt{\Delta}} = \frac{|b+a-1|}{\sqrt{\Delta}} = \frac{|-b-a+1|}{\sqrt{\Delta}}.$$

The rest of the derivation follows similarly:

$$P(\text{error}|+) = P(Z_2 > z_2) = 1 - P(Z_2 < z_2) = 1 - P\left(Z_2 < \frac{|-b-a+1|}{\sqrt{\Delta}}\right)$$

as expected (rightmost term in (D.6)).

In conclusion:

$$\begin{aligned} L_2(a,b) &= \frac{1}{2}P\left(X_2^{(-)} < bX_1^{(-)} + a\right) + \frac{1}{2}P\left(X_2^{(+)} > bX_1^{(+)} + a\right) && \text{(D.17)}\\ &= \frac{1}{2}\left(1 - P\left(X_2^{(-)} > bX_1^{(-)} + a\right)\right) + \frac{1}{2}P\left(X_2^{(+)} > bX_1^{(+)} + a\right) && \text{(D.18)}\\ &= \frac{1}{2}\left(1 - \Phi\left(\frac{|-b+a+1|}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{|-a-b+1|}{\sqrt{\Delta}}\right)\right) && \text{(D.19)}\\ &= \frac{1}{2}\left(1 - \Phi\left(\frac{-a+b-1}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{a+b-1}{\sqrt{\Delta}}\right)\right) && \text{(D.20)} \end{aligned}$$

The above expression matches the one we aimed to derive, i.e., (D.6).

#### D.2.2.3 Special Case 2

In the particular case where $\sigma_1 = 1$, $\sigma_2 = \delta$, $\rho_{12} = \rho$, we have:

$$\Sigma = \begin{bmatrix} 1 & \rho\delta \\ \rho\delta & \delta^2 \end{bmatrix} = \boldsymbol{\lambda}\boldsymbol{\lambda}^T.$$

$$\boldsymbol{\lambda} = \begin{bmatrix} 1 & 0 \\ \rho\delta & \delta\sqrt{(1-\rho^2)} \end{bmatrix}$$

$$\tilde{\boldsymbol{\delta}} = \boldsymbol{\lambda}^T \cdot \begin{bmatrix} b \\ -1 \end{bmatrix} = \begin{bmatrix} \tilde{\delta}_1 \\ \tilde{\delta}_2 \end{bmatrix}$$

Thus:

$$\Delta = \tilde{\delta}_1^2 + \tilde{\delta}_2^2$$

or equivalently:

$$\Delta = (b - \rho\delta)^2 + \delta^2(1 - \rho^2)$$

In this particular case, with $\sigma_1 = 1$, $\sigma_2 = \delta$, $\rho_{12} = \rho$, it can be verified that the optimal solution $(a^*, b^*)$ that minimizes the error (D.6) is given by a line through the origin, $a^* = 0$, with a slope of $b^* = \delta(\delta - \rho)/(\delta\rho - 1)$.

We can also derive an analytical expression for the Bayes error:

$$\frac{|0 - b^* + 1|}{\sqrt{\Delta}} = \frac{b^* - 1}{\sqrt{S}} = \frac{\delta(\delta - \rho)/(\delta\rho - 1) - 1}{\sqrt{(\rho\delta - \delta(\delta - \rho)/(\delta\rho - 1))^2 + \delta^2(1 - \rho^2)}} \tag{D.21}$$

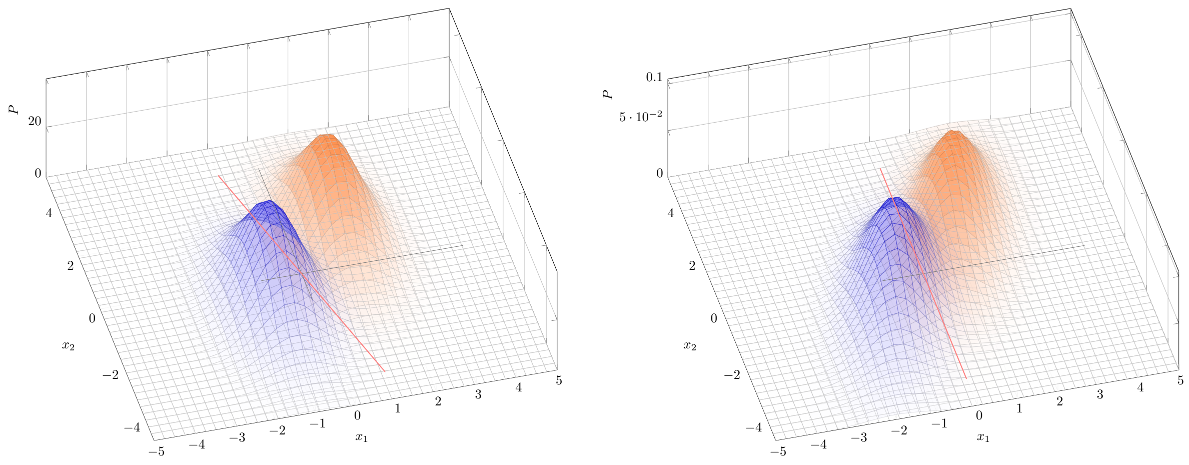
$$= \frac{1}{\delta}\sqrt{\frac{\delta^2 - 2\rho\delta + 1}{1 - \rho^2}}. \tag{D.22}$$

Thus:

$$L_2\left(0, \frac{\delta(\delta - \rho)}{\delta\rho - 1}\right) = 1 - \Phi\left(\frac{1}{\delta}\sqrt{\frac{\delta^2 - 2\rho\delta + 1}{1 - \rho^2}}\right). \tag{D.23}$$

### D.2.3 Illustrating Results

Two scenarios with $\delta = 2$ are illustrated in Figure 27. We can see that as $\rho$ increases between 0 and 0.5, the relevance of the second attribute decreases and the problem becomes more challenging. In particular, when $\rho = 0.5$, the second attribute is irrelevant for classification purposes, and it is clearly advantageous to collect more observations rather than an additional attribute for training a classifier.



(a) $\rho = 0.01$

(b) $\rho = 0.5$

Figura 27 – Gaussians separated by a line, where $\delta = 2$ and (a) $\rho \approx 0$ and (b) $\rho = 0.5$

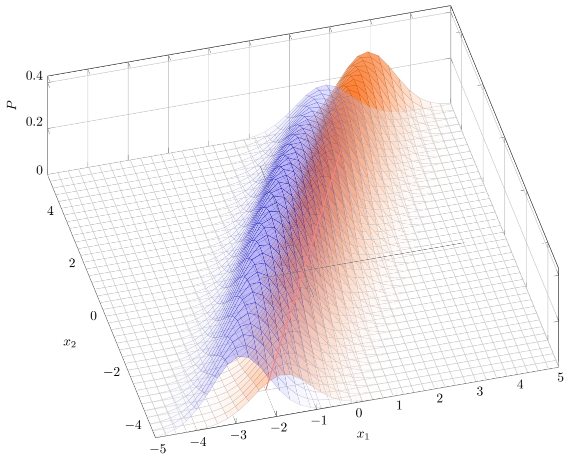
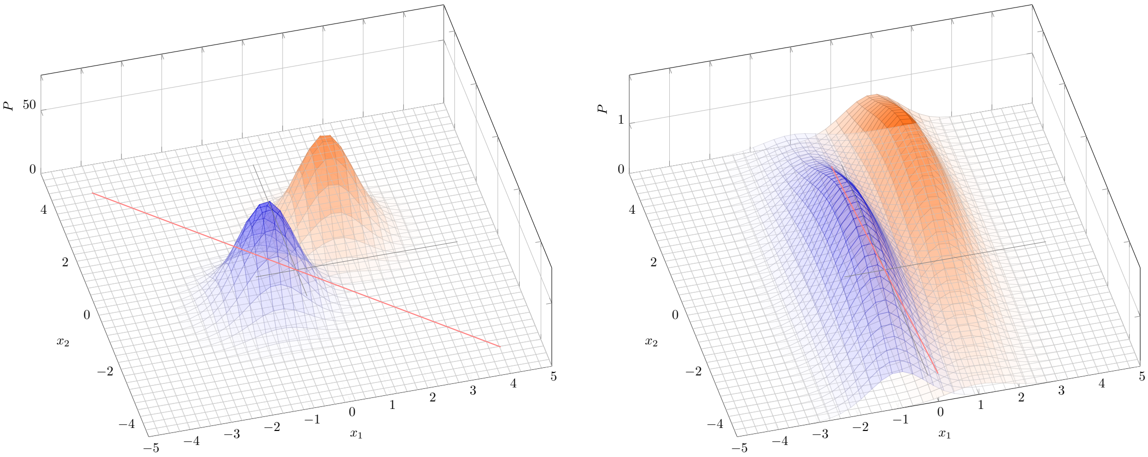


Figura 28 – Gaussians separated by a line when $\rho = 0.95$

In Figure 28, still with $\delta = 2$, we consider the case $\rho = 0.95$. We note that increasing $\rho$ causes the second attribute to become relevant again. In the extreme case where $\rho = 1$, we obtain a local minimum for the classification error. Intuitively, this occurs because the utility of the second attribute for classification purposes increases as $\rho$ grows from 0.5 to 1. With a useful second attribute to aid in classifier construction, the classification error decreases.

Figure 29 illustrates the impact of $\delta$ on the utility of $X_2$. The greater the value of $\delta$, the lower the utility of $X_2$. Intuitively, when the attribute $X_2$ has high dispersion, it does not significantly contribute to increasing the power of the classifier. In this case, collecting additional samples is more useful for improving the classifier than adding the attribute $X_2$.



(a) $\delta = 1$

(b) $\delta = 4$

Figura 29 – Impact of $\delta$ on the utility of $X_2$, assuming $\rho = 0.01$

D.3 3D: THREE ATTRIBUTES

### D.3.1 General Solution

As in previous sections, we assume that the conditional covariance matrices of the two considered classes are equal. The 3D classification problem can be formulated as follows:

$$(X_1, X_2, X_3) \sim \begin{cases} \mathcal{N}((\mu_1^+, \mu_2^+, \mu_3^+), \Sigma), & \text{if } Y = +1 \\ \mathcal{N}((\mu_1^-, \mu_2^-, \mu_3^-), \Sigma), & \text{if } Y = -1 \end{cases} \tag{D.24}$$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 & \rho_{13}\sigma_1\sigma_3 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 & \rho_{23}\sigma_2\sigma_3 \\ \rho_{13}\sigma_1\sigma_3 & \rho_{23}\sigma_2\sigma_3 & \sigma_3^2 \end{bmatrix} \tag{D.25}$$

We will also assume, as in previous sections:

$$P(Y = +1) = P(Y = -1) = \frac{1}{2} \tag{D.26}$$

and

$$\mu^+ = (1, 1, 1), \quad \mu^- = (-1, -1, -1) \tag{D.27}$$

Given a 3D classifier, defined by the plane: $X_3 = d + eX_1 + fX_2$, the corresponding expected error probability is calculated by:

$$\begin{aligned} L_3(d, e, f) &= \frac{1}{2}(1 - \Phi(\text{Distance}(+))) + \frac{1}{2}(1 - \Phi(\text{Distance}(-))) = \\ &= \frac{1}{2}\left(1 - \Phi\left(\frac{|d + e + f - 1|}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{|-d + e + f - 1|}{\sqrt{\Delta}}\right)\right) \end{aligned} \tag{D.28}$$

where:

- $\Phi(\cdot)$ is the cumulative distribution function of a Gaussian with zero mean and unit variance.
- Distance($+$) and Distance(-) are the distances of the separating hyperplane $h$ to the mean of the positive and negative classes, respectively.

The term $\Delta$ that appears in the denominator of the above expression is obtained through the Cholesky decomposition of the matrix $\Sigma$.

**Teorema 2.** *A covariance matrix $\Sigma$ of dimension $M \times M$ can be decomposed, via Cholesky decomposition, into a lower triangular matrix $\boldsymbol{\lambda}$ such that $\boldsymbol{\lambda}\boldsymbol{\lambda}^T = \Sigma$.*

Thus, $\Delta$ is given by:

$$\Delta = A^2 + B^2 + \lambda_{33}^2 \tag{D.29}$$

where:

- $A = e\lambda_{11} + f\lambda_{21} - \lambda_{31}$
- $B = f\lambda_{22} - \lambda_{32}$
- $\lambda_{11} = \sigma_1$, $\lambda_{12} = 0$, $\lambda_{13} = 0$
- $\lambda_{21} = \rho_{12}\sigma_2$, $\lambda_{22} = \sigma_2\sqrt{1 - \rho_{12}^2}$, $\lambda_{23} = 0$
- $\lambda_{31} = \rho_{13}\sigma_3$, $\lambda_{32} = \frac{(\rho_{23} - \rho_{12}\rho_{13})\sigma_3}{\sqrt{1-\rho_{12}^2}}$, $\lambda_{33} = \sigma_3\sqrt{1 - \rho_{13}^2 - \frac{(\rho_{23}-\rho_{12}\rho_{13})^2}{(1-\rho_{12})^2}}$
- $\lambda_{ij}$ are elements of $\boldsymbol{\lambda}$
- $\boldsymbol{\lambda}$: is the lower-triangular Cholesky factor of $\Sigma$ such that $\Sigma = \boldsymbol{\lambda} \cdot \boldsymbol{\lambda}^T$
- $\tilde{\boldsymbol{\delta}} = \boldsymbol{\lambda}^T \cdot \begin{bmatrix} e \\ f \\ -1 \end{bmatrix} = \begin{bmatrix} e\lambda_{11} + f\lambda_{21} - \lambda_{31} \\ f\lambda_{22} - \lambda_{32} \\ -\lambda_{33} \end{bmatrix}$
- $\Delta = \sum_{\tilde{\delta}_i \in \tilde{\boldsymbol{\delta}}} \tilde{\delta}_i^2 = \tilde{\delta}_1^2 + \tilde{\delta}_2^2 + \tilde{\delta}_3^2 = (e\lambda_{11} + f\lambda_{21} - \lambda_{31})^2 + (f\lambda_{22} - \lambda_{32})^2 + (-\lambda_{33})^2$

Note that this formulation includes many (six) parameters, and the mathematical optimization problem becomes complex. In Appendix F, we consider four special scenarios in which we can simplify the Bayes error expression and present it in closed form. In general, the complexity of the above problem motivates the use of simulation, as considered in this work, to evaluate the impact of different parameters on the quality of the obtained classifier.

### D.3.2 Optimal Solution

Now, suppose that the equation of the optimal separating plane is:

$$X_3 = d^* + e^* X_1 + f^* X_2 \tag{D.30}$$

To minimize $P(\text{Error})$ with respect to $d^*$, $e^*$, and $f^*$, we need to set their partial derivatives to zero. Then, it can be shown that:

- $d^* = 0$ (due to the general symmetry around the origin $(0, 0, 0)$)
- Both $e^*$ and $f^*$ depend on the six parameters that define the matrix $\Sigma$, namely: $\sigma_1, \sigma_2, \sigma_3, \rho_{12}, \rho_{13}, \rho_{23}$. Unlike the 1D and 2D cases, in the 3D case we were unable to obtain a closed-form mathematical expression for $e^*$ and $f^*$ (see Appendix F for special cases where closed-form solutions were possible).

In the particular case where $d = 0$, we obtain from (D.28):

$$L_3(0, e, f) = 1 - \Phi\left(\frac{|1 - e - f|}{\sqrt{\Delta}}\right) \tag{D.31}$$

**Teorema 3.** *The minimum probability of* ***Theoretical Error*** *(Bayes Risk) in the 3D case (trivariate Gaussian) is*

$$P(\mathit{Error}) = 1 - \Phi\left(\frac{|1 - e^* - f^*|}{\sqrt{\Delta}}\right)$$

*The denominator $\Delta$ is characterized as a function of the Cholesky decomposition of the matrix $\Sigma$, as given in* (D.29).

## D.4 GENERAL CASE WITH $d$ DIMENSIONS

**Teorema 4.** *If $(X_1, X_2, \ldots, X_d) \sim N(\mu_1, \mu_2, \ldots, \mu_d, \Sigma)$, then*

$$P\left(\tilde{b} > \sum_{i=1}^{d} X_i w_i\right) = P(Z_2 > z_2)$$

*where $Z_2 \sim N(0, 1)$ and*

$$z_2 = \frac{-\tilde{b} + \sum w_i \mu_i}{\sqrt{\Delta}}.$$

*In the above expression, we have:*

$$\Delta = \sum \tilde{\delta}_i^2$$

*and*

$$\Sigma = \boldsymbol{\lambda}\boldsymbol{\lambda}^T, \quad \tilde{\boldsymbol{\delta}} = \boldsymbol{\lambda}^T \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} = \begin{bmatrix} \tilde{\delta}_1 \\ \tilde{\delta}_2 \\ \vdots \\ \tilde{\delta}_d \end{bmatrix}$$

**Prova:** The theorem above is a consequence of basic properties of affine transformations of multivariate Gaussians.[1] Indeed,

$$\sum X_i w_i \sim \mathcal{N}\left(\boldsymbol{w}^T\boldsymbol{\mu}, \boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}\right) \tag{D.32}$$

Thus,

$$\frac{\sum X_i w_i - \boldsymbol{w}^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}} \sim \mathcal{N}(0,1) \tag{D.33}$$

Therefore,

$$P\left(\tilde{b} > \sum_{i=1}^{d} X_i w_i\right) = P\left(\frac{\tilde{b} - \boldsymbol{w}^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}} > \frac{\sum_{i=1}^{d} X_i w_i - \boldsymbol{w}^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}}\right) = P(Z_2 < -z_2) = P(Z_2 > z_2) \tag{D.34}$$

Note also that

$$\sqrt{\boldsymbol{w}^T\Sigma\boldsymbol{w}} = \sqrt{\boldsymbol{w}^T\boldsymbol{\lambda}\boldsymbol{\lambda}^T\boldsymbol{w}} = \sqrt{\Delta}. \tag{D.35}$$

□

Recall that in this work we consider two classes. In class 1, we have $\mu_1 = \mu_2 = \ldots = \mu_d = 1$, and in class 2, we have $\mu_1 = \mu_2 = \ldots = \mu_d = -1$. Thus, $-\sum w_i + \tilde{b}$ and $\sum w_i + \tilde{b}$ have opposite signs. Given the above assumptions and the fact that both classes share the same covariance matrix $\Sigma$, we have the following theorem.

**Teorema 5.** *Given two classes of points generated by multivariate Gaussians, with means $(+1,+1,\ldots,+1)$ and $(-1,-1,\ldots,-1)$ and covariance matrix $\Sigma$, the theoretical error of the classifier characterized by the separating hyperplane $\sum_{i=1}^{d} w_i X_i = \tilde{b}$ is given by*

$$P(error) = \frac{1}{2}\left(1 - \Phi\left(\frac{|\tilde{b} - \sum_{i=1}^{d} w_i|}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{|-\tilde{b} - \sum_{i=1}^{d} w_i|}{\sqrt{\Delta}}\right)\right) \tag{D.36}$$

*where $w_d = -1$.*

**Case a)** $\sum w_i \leq 0$. Thus, since $\sum w_i + \tilde{b}$ and $-\sum w_i + \tilde{b}$ have opposite signs, $\sum w_i + \tilde{b} \leq 0 \leq -\sum w_i + \tilde{b}$.

In this case, the classification rule is as follows. The estimated class, $\hat{Y}$, is given by:

$$\hat{Y} = \begin{cases} +1, & \text{if } \sum X_i w_i < \tilde{b} \\ -1, & \text{otherwise.} \end{cases} \tag{D.37}$$

In particular, in the two-dimensional case, we have $X_2 = a + bX_1$ as the separating line, $w_2 = -1$, $w_1 = b$, $\tilde{b} = -a$, so we classify as $+1$ if $-X_2 + bX_1 < -a$ and as $-1$ otherwise.

Then, by Theorem 4, for class 2 we have:

$$z_2 = \frac{-\tilde{b} + \sum w_i \mu_i}{\sqrt{\Delta}} = \frac{-\tilde{b} - \sum w_i}{\sqrt{\Delta}} = \frac{|-\tilde{b} - \sum w_i|}{\sqrt{\Delta}}.$$

[1] <https://en.wikipedia.org/wiki/Multivariate_normal_distribution>

We conclude that:

$$P(\text{error}|-) = P(Z_2 > z_2) = 1 - P(Z_2 < z_2) = 1 - P\left(Z_2 < \frac{|-\tilde{b} - \sum w_i|}{\sqrt{\Delta}}\right)$$

as expected (the term on the right of (D.36)).

Now consider class 1. By Theorem 4,

$$z_2 = \frac{-\tilde{b} + \sum w_i \mu_i}{\sqrt{\Delta}} = \frac{-\tilde{b} + \sum w_i}{\sqrt{\Delta}} = -\frac{|-\tilde{b} + \sum w_i|}{\sqrt{\Delta}} = -\frac{|\tilde{b} - \sum w_i|}{\sqrt{\Delta}}.$$

The rest of the derivation follows similarly to conclude that:

$$P(\text{error}|+) = 1 - P(Z_2 > z_2) = 1 - P(Z_2 < -z_2) = 1 - P\left(Z_2 < \frac{|\tilde{b} - \sum w_i|}{\sqrt{\Delta}}\right)$$

as expected (the term on the left of (D.36)).

Concluding:

$$L_d(\boldsymbol{w}) = \frac{1}{2}P\left(X_d^{(+)} < \sum_{i=1}^{d-1} w_i X_i^{(+)} - \tilde{b}\right) + \frac{1}{2}P\left(X_d^{(-)} > \sum_{i=1}^{d-1} w_i X_i^{(-)} - \tilde{b}\right) \tag{D.38}$$

$$= \frac{1}{2}\left(1 - P\left(X_d^{(+)} < \sum_{i=1}^{d-1} w_i X_i^{(+)} - \tilde{b}\right)\right) + \frac{1}{2}P\left(X_d^{(-)} > \sum_{i=1}^{d-1} w_i X_i^{(-)} - \tilde{b}\right) \tag{D.39}$$

$$= \frac{1}{2}\left(1 - \Phi\left(\frac{|\tilde{b} - \sum w_i|}{\sqrt{\Delta}}\right)\right) + \frac{1}{2}\left(1 - \Phi\left(\frac{|-\tilde{b} - \sum w_i|}{\sqrt{\Delta}}\right)\right) \tag{D.40}$$

The above expression is equivalent to the one we want to obtain (D.36).

**Case b)** $\sum w_i \geq 0$. Thus, $\sum w_i + \tilde{b} \geq 0$.

In this case, the classification rule is as follows. The estimated class, $\hat{Y}$, is given by:

$$\hat{Y} = \begin{cases} -1, & \text{if } \sum X_i w_i < \tilde{b} \\ +1, & \text{otherwise.} \end{cases} \tag{D.41}$$

This case is analogous to the previous one, and we omit it for brevity.

# E ADDITIONAL COMMENTS ON THE BAYES CLASSIFIER FOR THE CASE WHERE SAMPLES ARE DRAWN FROM BIVARIATE GAUSSIANS

Next, we consider the case where the samples are drawn from bivariate Gaussians, and the conditions of Section D.2.2.3 are satisfied.

We have $\sigma_1 = 1$, $\sigma_2 = \delta$, $\rho_{12} = \rho$. We also have:

$$\Sigma = \begin{bmatrix} 1 & \rho\delta \\ \rho\delta & \delta^2 \end{bmatrix} = \boldsymbol{\lambda}\boldsymbol{\lambda}^T$$

$$\boldsymbol{\lambda} = \begin{bmatrix} 1 & 0 \\ \rho\delta & \delta\sqrt{(1-\rho^2)} \end{bmatrix}$$

In addition,

$$\Delta = (b - \rho\delta)^2 + \delta^2(1-\rho^2).$$

Since in this case the optimal classifier $h^{(B)}$ belongs to the dictionary formed by lines in $\mathbb{R}^2$, the Bayes error, corresponding to the optimal classifier, is given by:

$$L(h^{(B)}) = L(h_2^*) = L_2\left(0, \frac{\delta(\delta-\rho)}{\delta\rho - 1}\right) = 1 - \Phi\left(\frac{1}{\delta}\sqrt{\frac{\delta^2 - 2\rho\delta + 1}{1-\rho^2}}\right). \tag{E.1}$$

By taking the partial derivative of the above expression with respect to $\rho$ and setting the result to zero, we can determine the value of $\rho$ that maximizes the loss. In fact, we conclude that the upper limit of $L(h^{(B)})$ is reached at $\rho = 1/\delta$, as detailed below.

Taking the derivative with respect to $\rho$ of the term inside the square root in the above expression and setting the result to zero, we obtain:

$$\frac{d}{d\rho}\frac{\delta^2 - 2\rho\delta + 1}{1-\rho^2} = \frac{-2\delta}{1-\rho^2} - 2\rho\frac{\delta^2 - 2\rho\delta + 1}{(1-\rho^2)^2} = 0 \tag{E.2}$$

Thus, the value of $\rho$ that maximizes the loss satisfies:

$$-2\delta(1-\rho^2) + 2\rho(\delta^2 - 2\rho\delta + 1) = 0 \tag{E.3}$$

The above equation has two solutions, $\rho = 1/\delta$ and $\rho = \delta$, and it can be verified that the first corresponds to a maximum of the loss function $L(h^{(B)})$ while the second corresponds to a local minimum. Since $\rho \in [-1, 1]$ and $\delta \geq 1$, we focus on the upper limit of the error, which occurs when $\rho = 1/\delta$. Thus,

$$0 \leq L(h^{(B)}) \leq 1 - \Phi(1). \tag{E.4}$$

The function $\Phi$ approaches 1 when its argument tends to infinity. The lower limit of the error is reached for $\rho \approx -1$, corresponding to scenarios where $X_2$ is a deterministic function of $X_1$.

As $\rho$ increases from $-1$ to $1/\delta$, the error increases; as $\rho$ increases further, the error decreases to zero.

# F FOUR SCENARIOS OF INTEREST IN THE 3D CASE

This appendix is a continuation of Appendix D.3.

## F.1 INTRODUCTION

In this appendix, we consider the classification problem in the three-dimensional case.

- The conditional distributions of $X$, given the label $Y$, are both trivariate normal distributions.
- The two centroids are fixed at $(1, 1, 1)$ and $(-1, -1, -1)$.
- The covariance matrix $\Sigma$, which depends on 6 parameters, is the same for the two Gaussians considered, one for each class.
- For a given problem, defined by the matrix $\Sigma$, once the sample size $n$ is fixed, there is no theoretical difficulty in programming the simulator to calculate the expected error rate.

Why were the 4 scenarios created?

- The main theoretical difficulty to be resolved is determining the optimal separating plane, whose equation we represent by: $x_3 = d^* + e^* x_1 + f^* x_2$. Once this equation is obtained, the corresponding Bayes Risk can be automatically calculated.
- Given the symmetry of the problem with respect to the origin $(0, 0, 0)$, the optimal separating plane must necessarily pass through the origin. In other words, we know that $d^* = 0$.
- Thus, we are left with a minimization problem with two unknowns: $e^*$ and $f^*$.
- We have attempted to obtain an analytical solution for this problem, i.e., to express $e^*$ and $f^*$ mathematically as functions of the 6 parameters that define the matrix $\Sigma$. So far, we have not been successful.
- This is where the 4 scenarios come in. In each of them, simplifying assumptions about the matrix $\Sigma$ are introduced, so that it depends on only 3 parameters. Under these conditions, we can obtain analytical solutions to the problem.
- Another possible alternative, which would eliminate the need for the 4 scenarios, would be to attempt to solve this minimization problem using numerical analysis. We leave this approach as a topic for future work.

## F.2 FOUR SCENARIOS

To simplify the mathematics, we will consider four specific scenarios. In each scenario, the idea is to reduce the number of free parameters, allowing analytical expressions to be obtained for the optimal coefficients of the plane, $e^*$ and $f^*$, and the Bayes risk.

In what follows, when we say that two attributes are independent, we are referring to conditional independence, given that the class of the observation is known. Similarly, when we say that two attributes are correlated, we are referring to conditional correlation, given that the class of the observation is known.

1. **Scenario 1:** The 3 attributes are pairwise independent, i.e., $\rho_{12} = \rho_{13} = \rho_{23} = 0$. In this case, it can be shown that:

   a) $e^* = -\frac{\sigma_3^2}{2\sigma_1^2}$

   b) $f^* = -\frac{\sigma_3^2}{2\sigma_2^2}$

   c) $P(\text{Error}) = 1 - \phi\left(\frac{\sqrt{1-e^*-f^*}}{\sigma_3}\right)$

2. **Scenario 2:** $X_1$ and $X_2$ are correlated, $(X_1, X_2)$ and $X_3$ are independent

   - Attributes $X_1$ and $X_2$ are correlated, i.e., $\rho_{12} = \rho$ can be $\neq 0$
   - Attributes $X_1$ and $X_3$ are independent, i.e., $\rho_{13} = 0$
   - Attributes $X_2$ and $X_3$ are independent, i.e., $\rho_{23} = 0$
   - $\sigma_1 = \sigma_2 = \sigma$

   In this case, it can be shown that:

   a) $e^* = f^* = -\frac{\sigma_3^2}{2\sigma^2(1+\rho)}$

   b) $P(\text{Error}) = 1 - \phi\left(\sqrt{1 - 2e^*\frac{1}{\sigma_3}}\right)$

3. **Scenario 3:** $X_1$ and $X_2$ are independent, $(X_1, X_2)$ and $X_3$ are correlated

   - Attributes $X_1$ and $X_2$ are independent, i.e., $\rho_{12} = 0$
   - Attributes $X_1$ and $X_3$ are correlated, i.e., $\rho_{13}$ can be different from 0
   - Attributes $X_2$ and $X_3$ are correlated, i.e., $\rho_{23}$ can be different from 0
   - $\rho_{13} = \rho_{23} = r$ (Restriction: $-\frac{\sqrt{2}}{2} < r < \frac{\sqrt{2}}{2}$)
   - $\sigma_1 = \sigma_2 = \sigma$

   In this scenario, it can be shown that:

   a) $e^* = f^* = \frac{\sigma_3}{(\sigma_3 - r\sigma)} \cdot \frac{\sigma}{(2r\sigma_3 - \sigma)}$

   b) $P(\text{Error}) = 1 - \phi\left(\frac{1-2e^*}{\sqrt{\Delta}}\right)$, where $\Delta = 2(e\sigma - r\sigma_3)^2 + \sigma_3^2(1 - 2r^2)$

4. **Scenario 4:** $X_1$ and $X_2$ are correlated, $(X_1, X_2)$ and $X_3$ are correlated

   - All attributes have equal dispersion ($\sigma_1 = \sigma_2 = \sigma_3 = \sigma$)
   - Attributes $X_1$ and $X_2$ are correlated, i.e., $\rho_{12} = \rho$ can be different from 0
   - Attributes $X_1$ and $X_3$ are correlated, i.e., $\rho_{13}$ can be different from 0
   - Attributes $X_2$ and $X_3$ are correlated, i.e., $\rho_{23}$ can be different from 0
   - $\rho_{13} = \rho_{23} = r$
   - Restriction: $-\sqrt{\frac{1+\rho}{2}} \leq r \leq \sqrt{\frac{1+\rho}{2}}$

   In this scenario, it can be shown that:

   a) $e^* = f^* = \frac{1-r}{2r-(1+\rho)}$

   b) $P(\text{Error}) = 1 - \phi\left(\frac{1-2e^*}{\sqrt{\Delta}}\right), \quad \Delta = A^2 + B^2 + \lambda_{33}^2$

   where:

   - $A = \sigma[e^*(1+\rho) - r];$
   - $B = \sigma\left[\sqrt{1-\rho^2}e^* - \frac{r(1-\rho)}{\sqrt{1-\rho^2}}\right];$
   - $\lambda_{33} = \sigma\sqrt{1 - 2r^2/(1+\rho)}$

# G A SIMPLE BISECTOR FOR TWO OBSERVATIONS ($n = 2$) WITH ONE OR TWO ATTRIBUTES

The results from the previous section are asymptotic results for an infinite sample size. Now, we consider the extreme opposite: a dataset composed of two samples, one from each class. We denote by $h_1^{(SVM)}$ and $h_2^{(SVM)}$ the classifier obtained using **SVM**, with one attribute ($X_1$) and two attributes, respectively. Thus, the **SVM** solutions correspond to simple bisectors as separation boundaries. The expected errors are given by

$$E\left(L(h_1^{(SVM)})\right) = \int_{c=-\infty}^{+\infty} L_1(c)\frac{1}{\sqrt{\pi}}e^{-c^2}dc \approx 0.2070336 \tag{G.1}$$

and

$$E\left(L(h_2^{(SVM)})\right) = \int_{\boldsymbol{w}\in\mathbb{R}^2}\int_{\boldsymbol{v}\in\mathbb{R}^2} L_2(g(\boldsymbol{v},\boldsymbol{w}))\phi_{+1}(\boldsymbol{v})\phi_{-1}(\boldsymbol{w})d\boldsymbol{v}d\boldsymbol{w} \tag{G.2}$$

where the expectations are taken over random training sets with two observations each, drawn from bivariate Gaussians, $\boldsymbol{v} = (v_1, v_2)$, $\boldsymbol{w} = (w_1, w_2)$, $\phi_t(\cdot)$ is the probability density function of the bivariate Gaussian corresponding to class $t$, and $g(\boldsymbol{v},\boldsymbol{w})$ is a function that maps a pair of points to their intersection and the slope of the perpendicular bisector.

$$g(\boldsymbol{v},\boldsymbol{w}) = \left(\frac{w_1 - v_1}{w_2 - v_2}\cdot\frac{v_1 + w_1}{2} + \frac{v_2 + w_2}{2}, -\frac{w_1 - v_1}{w_2 - v_2}\right). \tag{G.3}$$

Comparing $L_1(0)$ with $L(h_1^{(SVM)})$, (D.2) with (G.1), we evaluate the impact of finite sample size when only one attribute is available and compare the performance of the loss function $L(h^{(B)})$ with that of the loss function $L(h_2^{(SVM)})$, (D.23) with (G.2), when two attributes are available.

For $\rho = 0$, the corresponding loss values are presented in the rows $n = 1,024$ and $n = 2$ of Table 1, whose values agree up to three decimal places with the expressions (G.1)-(G.2) and (D.2)-(D.23), respectively. Comparing (G.1) with (G.2), which correspond to the elements in the first row of Table 1, it is noted that when the sample size is small and $\delta \geq 2$, it is beneficial to use fewer attributes. On the other hand, comparing (D.2) with (D.23), which correspond to the elements in the last row of Table 1, it is noted that when the sample size is small and $1 \leq \delta \leq 7$, it is beneficial to use more attributes.

# H A MODEL TO UNDERSTAND THE BEHAVIOR OF BAYES RISK

## H.1 ONE AND TWO DIMENSIONS

Suppose that only $X_1$ and $X_2$ are present. Consider an ellipse in $\mathbb{R}^2$ (say, centered at the origin $(0, 0)$) whose equation is

$$x^T \Sigma^{-1} x = 1.$$

For each class, we have an ellipse, and for each ellipse, we have a centroid. The line that connects the two centroids $(1, 1)$ and $(-1, -1)$ is the set of all vectors in $\mathbb{R}^2$ of the type

$$(c, c),$$

whose two coordinates are equal. This line cuts through the ellipse $x^T \Sigma^{-1} x = 1$ at some point with equal and positive coordinates,

$$(c_2, c_2) \cdot \Sigma^{-1} \begin{pmatrix} c_2 \\ c_2 \end{pmatrix} = 1.$$

Let $d_2$ be the distance of this point from the origin,

$$d_2 = \sqrt{2} c_2.$$

Intuitively, we notice that when this distance is large (Figure 30(a)), the ellipse aligns with the line that connects the centroids, making classification more difficult compared to scenarios where the distance is smaller (Figure 30(b)).

We do not have a formal argument to identify the conditions under which an increase in distance $d_2$ implies a decrease in classification power, but we have verified numerically that in several scenarios, the larger this distance, the lower the discrimination power of the classifier when both $X_1$ and $X_2$ are present. In particular, comparing Figure 30(a) with Figure 30(b), we find that classifying points in Figure 30(b) is easier than classifying points in Figure 30(a), bearing in mind that both clouds of points have centroids $(+1, +1)$ and $(-1, -1)$, and both clouds obey the same covariance matrix. In other words, both clouds will resemble the pattern in Figure 30(a), centered at $(+1, +1)$ and $(-1, -1)$, or both clouds will resemble the pattern in Figure 30(b), centered at $(+1, +1)$ and $(-1, -1)$. Classifying in the first case is more difficult than in the second, as illustrated in Figure 31.

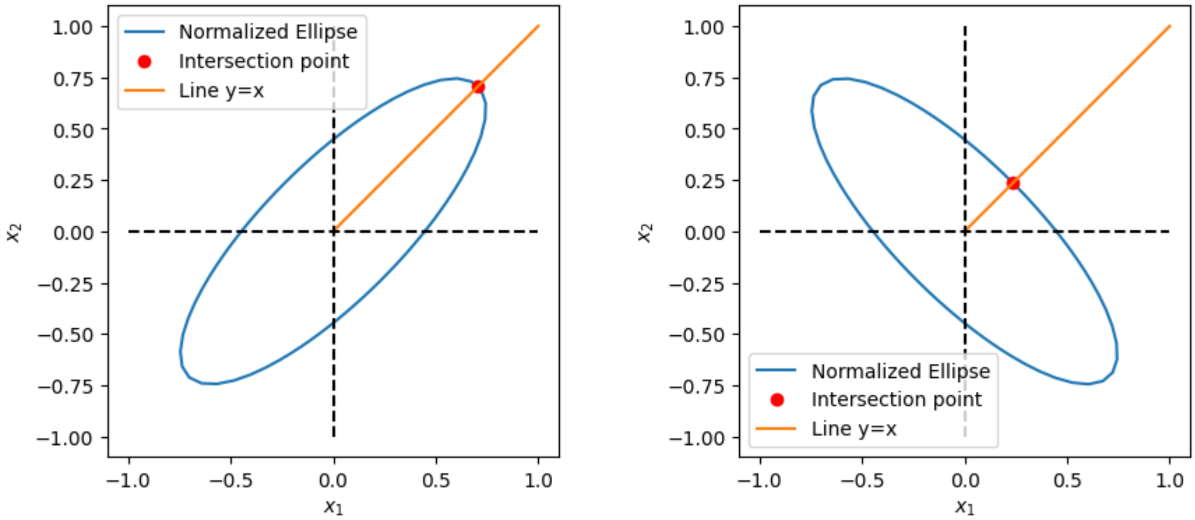
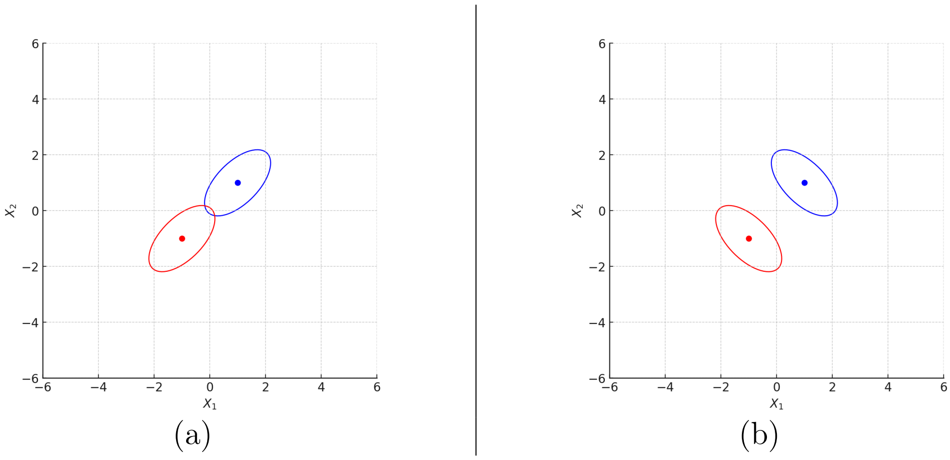


Figura 30 – $d_2$ for $\sigma_1 = 1, \sigma_2 = 1, \rho = 0.8$ and $d_2$ for $\sigma_1 = 1, \sigma_2 = 1, \rho = -0.8$



(a)

(b)

Figura 31 – Classifier with more (a) difficult ($\rho = 0.8$) and (b) easy ($\rho = -0.8$) tasks.

## H.2 EXTENDING TO THE THREE-DIMENSIONAL CASE

Extending the discussion, consider an ellipsoid in $\mathbb{R}^3$ (say, centered at the origin $(0, 0, 0)$) whose equation is

$$x^T \Sigma^{-1} x = 1,$$

where now $x \in \mathbb{R}^3$.

Recall that each class of points (e.g., orange and blue) is associated with an ellipsoid, which contains a centroid. The line connecting the two centroids is the set of all vectors in $\mathbb{R}^3$ of the type $(c, c, c)$, with all three coordinates equal. This line intersects the ellipsoid at a point with all three coordinates equal and positive.

$$(c_3, c_3, c_3) \cdot \Sigma^{-1} \begin{pmatrix} c_3 \\ c_3 \\ c_3 \end{pmatrix} = 1.$$

Let $d_3$ be the distance from this point to the origin,

$$d_3 = \sqrt{3} c_3.$$

Intuitively and numerically, we have verified in numerous scenarios that the larger this distance, the lower the discrimination power of the classifier when $X_1$, $X_2$, and $X_3$ are present. We leave a formal verification of the conditions under which this relationship applies for future work.

Below, we indicate how $d_2$ and $d_3$, which can be easily calculated using scientific computing software, help determine the utility of the attribute $X_3$.

## H.3 COMPARING THE TWO AND THREE ATTRIBUTE SCENARIOS

How can we use $d_3$ and $d_2$ to create, from the matrix $\Sigma$, an indicator of the additional discrimination potential of $X_3$, given that $X_1$ and $X_2$ are already present? The indicator we propose is $d_2/d_3$,

$$U_3(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) = \frac{d_2}{d_3} = \sqrt{\frac{2}{3}} \frac{c_2}{c_3}, \tag{H.1}$$

where $U_3$ is a utility metric for the third attribute.

Let $R_2$ be the Bayes risk with two attributes, and $R_3$ the corresponding risk with three attributes. Empirically, we verify that the ratio between the risks is a smooth and increasing function of the above utility function, i.e.,

$$\frac{R_2}{R_3} = \varphi(U_3), \tag{H.2}$$

where the function $\varphi(\cdot)$ is a smooth, increasing function, to be experimentally determined.

## H.4 EVALUATING THE UTILITY OF THE THIRD ATTRIBUTE

To evaluate the utility of the third attribute, $X_3$, given the attributes $X_1$ and $X_2$, we consider the following plots as outputs from our simulator for a given scenario:

- $\log_2 n^*$ as a function of the values of the considered parameter $\alpha$: recall that the threshold $n^\star$ corresponds to the number of observations beyond which it becomes advantageous to include attribute $X_3$, given that the other two attributes $X_1$ and $X_2$ are already present. The smaller the value of $n^\star$, the greater the range justifying the inclusion of $X_3$. For more details, see Section 3.1;

- $\log_2 n^*$ **vs** $R_2/R_3$: while the above-described plot has the parameter $\alpha$ values on the x-axis, in this and the next plot, we consider alternative metrics that somehow capture the utility of attribute $X_3$ for each value of the considered parameter $\alpha$. Firstly, we consider the ratio between the Bayes risk in the case of having two attributes $(X_1, X_2)$ and in the case of having three attributes $(X_1, X_2, X_3)$. Our goal here is to explain what affects the value of $n^\star$. In particular, we aim to understand to what degree the Bayes risk can be used to justify the behavior of $n^\star$. For more details, see Section 3.1 and Appendix **??**;

- $\log_2 n^*$ **vs** $U_3$: once again, our goal here is to explain what affects the value of $n^\star$. To this end, we propose a new metric, $U_3$, and present plots of $\log_2 n^\star$ as a function of $U_3$. For more details, see Appendix H.3;

- $U_3$ **vs** $R_2/R_3$: as indicated above, we have two attempts to explain the behavior of $n^\star$, the first based on Bayes risk and the second based on a new proposed metric, $U_3$. In this final plot, we relate these two attempts to explain the behavior of $n^\star$ by presenting one against the other. For more details, see Appendix **??** and Appendix H.3.