Master's Degree

in

Data Science

Evaluation of binary confusion matrices classical and novel approaches The Confusion Tetrahedron

Supervisor

Giuseppe Jurman

Candidate

Isabella Dario

Anno accademico 2020/2021

Isabella Dario: *Evaluation of binary confusion matrices: classical and novel approaches of visualization* | Tesi di Laurea Magistrale in Data Science
© Copyright Marzo 2022

Università degli studi di Trento:
www.unitn.it

Data Science:
datascience.unitn.it

Contents

Abstract **3**

1 Introduction **5**

1.1 Context 5

1.2 The added value 7

2 Classical approaches **11**

2.1 Confusion matrices 11

2.2 Concept of class of TP, TN, FP, FN 14

2.3 Performance metrics 15

2.4 Visualizations of the confusion matrices 25

2.4.1 First approach 25

2.4.2 Second approach 28

2.4.3 Third approach 33

2.4.4 Fourth approach 36

3 The confusion tetrahedron **43**

3.1 Shortcomings of classical approaches 43

3.2 The Confusion Tetrahedron: theoretical explanation 44

3.3 The Confusion Tetrahedron: implementation 49

3.3.1 Dataset 50

3.3.2 Data exploration and preparation 55

3.3.3 Fitting the models and standard evaluation 63

3.3.4 New approach to the evaluation 69

3.3.5 Technical specifications 71

4 Applications **75**

4.1 Global landscape for classifiers 76

4.2 Trajectories and ROC in Confusion tetrahedron 79

4.3 Trajectories in training epochs 83

4.4 Stratified vs non stratified resampling 91

5 Conclusions **95**

Bibliography **97**

Acronyms **107**

Appendices **111**

Appendix A Materials **113**

Appendix B Code **115**

Abstract

Data visualization can be a precious tool in supporting Machine Learning research, and in this thesis we introduce a novel graphical solution towards the evaluation of predictive algorithm effectiveness, as a *de facto* example of an actionable dataviz tool proactively useful in ML. Whatever the (machine) learning task, the model selection is a crucial step for providing an optimal solution. In the binary classification case, the best algorithm is identified by evaluating one or more performance metrics: among them are worth mentioning Accuracy, Sensitivity, Specificity, F_1-score, Matthews Correlation Coefficients as the more widespread, but the number of available alternatives is quite huge. All these functions rely as their input on the confusion matrix (CM), *i.e.*, the 2×2 table partitioning all the dataset's elements into the four cluster collecting, for both classes, the correctly and wrongly predicted samples. Since all performance metrics are summarizing the confusion matrix into a single figure, even those involving all four CM's entries present some limitations and can lead to biased conclusions on the model behaviour. Here we present the *Confusion Tetrahedron*, a construction for the three-dimensional visualization of a family of confusion matrices. This geometric view offers an intuitive and immediate tool to visually evaluate the classification model, thus allowing the researcher to assess the full behavior of the performance metrics. In particular, we detail the theoretical support for the construction of the confusion tetrahedron, and we demonstrate its usefulness on four practical use cases.

1 Introduction

1.1 Context

As the number of alternative models viable for the resolution of a Machine Learning (ML) problem increases, the greater the need for a reliable technique appropriate to assess the quality of the learning methods. After gathering trustworthy data from reliable sources and cleaning it to remove elements that might hamper the predictions and fitting different models, ML practitioners face the need to assess which is the best model to employ (*i.e.* model selection), or select the optimal set of parameters (*i.e.* hyperparameter tuning). In this perspective, the end of the learning process it is actually just the beginning of the workflow because after coping with the workload of data preprocessing and model fitting, comes the challenge of evaluating their performance and choosing the best one, triggering a whole new surge in the amount of work.

Currently, the canonical way to assess the effectiveness of machine learning models is to focus on the optimization of a certain performance metric. This metric usually stems from some flavour of accuracy function which aims to summarize one of the possible aspects of correctness of the model: a number of common alternatives are discussed in Section 2.3. In fact, several different metrics are available, tailored to deal with the diverse application scenarios and for each of these metrics the proper threshold needs to be chosen for the confidence level of the prediction. As a result, the proper evaluation of the fitted models is a considerable challenge for researchers who needs to decide upon the best model to employ and there is a critical need to make the evaluation more interpretable and accessible, especially in machine learning applications where selecting the optimal alternative is crucial. To this end, in an endeavour to propose a solution to such issue, here we suggest data visualization as an effective tool to be employed in order to aid machine learning operations and especially the assessment stage of the classification workflow. Data visualization has been widely used for supporting decision making especially in the business world (*e.g.* Business Intelligence (BI) and dashboards). However, efficient and effective data visualization techniques still have not yet been applied to the evaluation of the learned models. In fact, so far, most of the evaluation is done at tabular level, as most practitioners display data with contingency tables with the eventual addition of informative decorators while good design can instead make a huge difference in better grasping the behaviour and the performance of the learned models.

In this perspective, introducing visual elements in the learning pipeline could have a

profound impact on a wide range of tasks. In particular, data visualization offers invaluable opportunities not just at the end of the process, with the goal of evaluating the fitted models, but also all along the pipeline, in order to better interpret the behavior and to some extent to uncover what is going on under the hood. By breaking down the machine learning workflow (*see* Figure 1.1), one can better identify the different stages and areas where visualization may contribute to add value either for the exploration of the data, the training of the model or for monitoring purposes during the deployment phase.

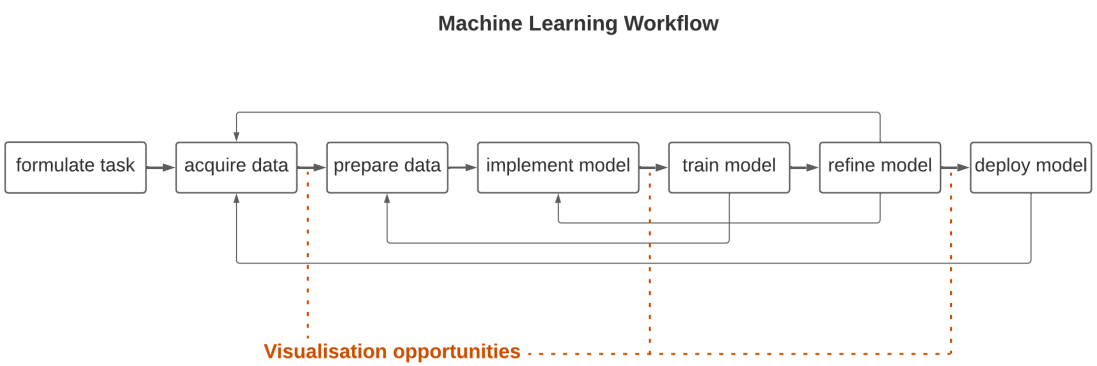


Figure 1.1: Context: opportunities for visualizations in ML

More specifically, this practice of building a visual context by translating the information being analyzed (*i.e.* data visualization) should be integrated into a typical iterative machine learning pipeline for five main reasons [1]. These areas of value can be summarized into the following:

1. *to visualize the training data*
   Through a proper exploratory data analysis and the visualization of the dataset, there is a greater chance to spot mistakes intrinsic to the starting dataset (*e.g.* instances incorrectly classified by human labelling) as well as better grasp the structure and the characteristics of the dataset which aids in performing a better choice for the algorithms to apply.

2. *to monitor and represent model performance*
   A visual element representing what the model is doing and how it is behaving contributes to the comprehension of its behaviour and helps in bridging the gap between the analytical framework (*i.e.* application of the algorithms) and the interpretation of the results (*i.e.* awareness of the outcomes).

3. *to aid interpretability and model inspection*
   Translating information into a graphical representation of the model behaviour to

some extent helps in cracking open the so called black box representing a convenient starting point for the debugging.

4. *to represent high-dimensional data*
   In machine learning scenarios datasets with a staggering number of features are extremely common and their analysis can be difficult to tackle. One way to face this challenge is through the visualization of the datasets also because such representations could allow researchers to reasonably cut out some features effectively managing the high dimensionality issue.

5. *for education and communication purposes*
   Having more interpretable results means also to have a model that is more accessible for a broader public hence better unleashing its positive impacts. This perk is particularly useful for all the non technical stakeholders that are generally involved in ML deployment (*i.e.* financial managers). It allows the effective transformation of the math behind the model into a visual environment that can be understood more easily without having to numb down the information.

As a result, the application of data visualization in machine learning operations is pivotal to provide practitioners with greater guidance on how to decide upon the adoption and deployment of machine learning algorithms. The present work aims at showing an effective case study demonstrating the helpful support that an actionable data visualization tool can provide to a machine learning pipeline, extending this research area by focusing on the visualisation of confusion matrices towards the model interpretation and evaluation stage.

1.2 The added value

Building up on the opportunities for data visualization in the machine learning field, this study concentrates on *classification tasks* and more specifically on *binary* classification algorithms. This choice is done for two main reasons. The first is that binary classification is deemed as the most common task in machine learning. The second is that it is also relatively general, since multiclass problems and regression problems can be both reduced to the binary case, although not lossless.

Within this framework, we demonstrate that there is room for improvement in the *evaluation* of classification algorithms and that *data visualization* can aid in filling this gap. In particular, we will show how the confusion matrix – the basis for every performance metric – can be redesigned with a visualization that better depicts the geometrical environment in which the model learns.

To illustrate this, first we undertake an extensive and systematic study of the tools available to assess the quality of binary classification algorithms. The evaluation is generally done by looking at a specific measure and so we analyzes a set of most widespread different *performance metrics* currently adopted. Then, we provide a structured *review of the*

classical approaches to the visualization of the confusion matrix published in the literature. These visualizations can have several decorators and alternative designs they all indeed share the same property, that is the bidimensionality. Although this feature can be to some extent more useful for human readability, it comes at a cost in terms of quantity and quality of information delivered, since the confusion matrix has four independent entries. Hence, in order to overcome the shortcoming of currently available designs, a new *three-dimensional* visualization is presented, called the *Confusion Tetrahedron*, a geometric environment which allows the interpretation of the performance metric within a visual structure. In particular, each point of the Confusion Tetrahedron (CT) corresponds to all possible equivalence classes of confusion matrices that the learned model could lead to. A full three dimensional (3D) view endowed with an adequate color palette for the behavior of the performance metric through the placement of these points within the tetrahedron presents three main advantages in the evaluation of a classification algorithm with respect to the current approaches. The following are:

1. *A more effective comparison of the performance metrics*
   For instance, the colored tetrahedron clearly explains why the trustworthiness of the information provided by the Matthew's Correlation Coefficient (MCC) is higher than other performance metrics (e.g F_1-score, accuracy)

2. *The possibility to view information that would not be observable otherwise*
   The confusion tetrahedron delivers further information with respect to the standard confusion matrix. Without the tetrahedral structure, it would not be possible to observe the environment in which the model learns and how the specific use case places with respect to the best and worst outcome. The possibility to give an holistic view of the performance of the model is undoubtedly the greatest advantage of the tetrahedron.

3. *A full snapshot of the dynamics of the algorithm for ML applications*
   The tetrahedron provides an accurate visualization of the behavior of the algorithm depicting the progressive evolution of the performance. Being able to visualize the change in the various phases of the learning process is extremely useful in analyzing the model and assessing the best one.

To better communicate how the confusion tetrahedron is an upgrade with respect to the current situation, the rest of the work is organized as follows. The classical approaches employed to assess the quality of the learned models and the four main visualizations for the confusion matrix are illustrated in Chapter 2. The following Chapter 3 introduces the Confusion Tetrahedron formally defining its geometrical features and properties as well as the mathematical procedure leading to its construction. The explanation of the confusion tetrahedron is deepened in specific application scenarios: as a global landscape for

classifiers in Chapter 4; as an environment for trajectories alternative to the ROC curves in Section 4.2; as a temporal analysis enabler for Deep Learning applications in Section 4.3 and as an exploration tool for resampling strategies in Section 4.4. Lastly, Chapter 5 steps back from the specifics to summarize the role of the tetrahedron in the evaluation of machine learning algorithms and to discuss issues that are still open such as further steps that time allowing would have been undertaken, limitations to take into account and further research opportunities.

2 Classical approaches

2.1 Confusion matrices

Data Science (DS) is an interdisciplinary field that uses methods, models, algorithms and technologies to extract knowledge and insights from data in order to solve a specific problem. Within the realm of data science, Machine Learning (ML) is the area that focuses on the study of statistical models that can be used by computers to perform a certain task without having explicit instructions on how to solve it. Essentially, through machine learning, computers are able to build a goodness function predicting a certain outcome simply starting from a set of input data [2]. By doing so computers become able to learn as a human usually does, that is through experience.
On one hand, this has the perk of automating the learning process as well as the resolution of a wide range of problems cutting the workload and resources invested. On the other hand, without relying on a pre-established approach to the learning task, several different models can be adopted to tackle the resolution. As the number of methods aiming at solving the same problem increases, it is important to be able to decide upon the best method to employ, choosing the one that best predicts the outcome. The issue is in the definition of *best* as there is no predetermined better way to assess the learned models. Hence the need for the definition of relevant criteria for the evaluation of algorithm's performance.

There are several factors affecting the choice of the preferred model such as the Central Processing Unit (CPU) time required to get a result or the level of interpretability, complexity or scalability of the model. There is no single way to measure the performance of a model as different criteria are appropriate in different settings and all various kinds of errors can have a range of disparate costs. Hence, several performance metrics can be found in the scientific literature, either based on a threshold, on a probability of error or on ranks [3, 4]. However, usually, the goal is for the algorithm to generalize the underlying explanatory laws of the training data to independent test data. For any type of prediction algorithm, it is of prime interest to secure that it will be able to correctly predict the label for the test data, that is data that have not been used to train the algorithm under evaluation. The ability of the algorithm to generalize to new examples from the same data domain is generally assessed by evaluating the accuracy of the model; that is, the number of instances in the test set on which the algorithm's prediction is correct divided by the total number of instances. However, accuracy has several shortcomings [5], hence different metrics have been developed to better represent the performance of the model and to better assess which

model is superior. These metrics are outlined in Section 2.3: so far the key aspect is that the reliable application of any machine learning model to a specific use case is possible only after the determination of the model performance. The process of performance evaluation requires researchers to first run and test the algorithms on many relevant datasets and then assess the performance of each model not only to select the one with the best metric as the preferred model but also to obtain a figure to be optimized by tuning all the parameters of the classifier.

While machine learning models offer a wide range of applications, this work focuses on *classification*, disregarding other kinds of tasks such as *regression* or *clustering*. The term classification means that the learning method should attribute the correct label to each data instance. The classifier maps an unlabeled instance to a set of discrete classes requiring the model to recognize the label (Y) of a specific object characterized by a set of features (X). Class labels are usually string values that gets encoded to numeric values before being fed to an algorithm for modelling. Hence, the output y consists in a suitable code for the class and it belongs to a finite set.
When the cardinality of this finite set equals to two, there are only two possible labels for the data and the classification is said to be binary. For example, $Y = \{0, 1\}$ in the case of the categorization of an email as spam or not, the classification of a patient as ill or healthy or the labelling of a user behavior as churn or not. When the cardinality is greater than two, there are more than two possible labels for the data and the classification is said to be multiclass. For example, $y_i = \{0, 1, ...N\}$ in the case of the classification of the model of a car or the labelling of a certain product in the inventory among others applications.
For the sake of generality, this work focuses on binary classification as the multiclass case can be brought down to the binary one using a strategy of fitting multiple binary classification models for each class versus all other classes (One-versus-All) or one model for each pair of classes (One-versus-One).

These binary classification tasks can be efficiently addressed by supervised machine learning techniques. Supervised ML is a type of machine learning where a specifically known dataset is provided to make predictions. In this case, the dataset includes two types of variables. The first type is the input variables ($X_i = \{x_1, x_2, ..., x_n\}$ for a dataset with N datapoints) which are the features characterizing the data instances of the phenomenon being studied. For example, following up on the example of the classification of an ill or healthy patient, input variables can be blood pressure, the weight, height, body temperature and so on. The second type of variable is the output label ($Y_i = \{0\}$ or $Y_i = \{1\}$ for supervised binary classification), that is the label associated to each data instance with a certain set of features. In the example of the patient, for each patient the dataset includes the health state of that person (eg. encoded as $y_i = 0$ if healthy, $y_i = 1$ if ill). To reach a prediction, first the dataset is divided into a train set (used to build the model) and a test set (used to test the model) usually assigning a fraction (e.g. 2/3) of the datapoints to the former and the

remaining portion to the latter. Then, the supervised learning method uses the train set to build an association $y = f(x)$ between input x and output y and the test set to assess the quality of this newly developed function. Ideally, this process should allow the researcher to uncover the underlying laws behind the behavior of the data, successfully predicting the class for the unseen data. Popular supervised machine learning techniques for classification include Logistic Regression [6], Naive Bayes [7], K-Nearest Neighbors [8], Support Vector Machines [9], Random Forest [10] and Gradient Boosting [11].

In order to select the best technique for a specific use case and assess the performance of the learned method, there is the need for a metric and an associated visualization that is able to reliably represent the quality of the algorithm.
Let $D = \{d_1, d_2, ..., d_N\}$ be a dataset of N elements where d_k represents the k-th element and let Θ be the set of the c classes $\Theta = \{\theta_1, \theta_2, ..., \theta_c\}$ defining the elements of the dataset. For each d_k, θ_i is the label that defines the i-th class for that datapoint. Suppose now a classifier C is applied to D with the task of discriminating elements belonging to different classes. The classifier C is a function that assigns to d_k, the k-th element of D, a label θ_j representing the predicted class for that element. If the function $C(d_k) = \theta_j$ estimates that d_k belongs to the j-th class while it really belongs to the i-th class and $i \neq j$ then the predicted class differs from the actual one and the element is misclassified reducing the quality of the learned method.
Hence, let $A = \{\alpha_1, \alpha_2, ..., \alpha_N\}$ be the set composed of the *actual* classes for the N datapoints of the dataset D and let $\varepsilon = \{\varepsilon_1, \varepsilon_2, ..., \varepsilon_N\}$ be the set composed of the *predicted* classes by the classifier C for each element in D. The quality of the classifier can be evaluated using a measuring function $\mathcal{M}$ that assigns a metric $\mu \in \mathbb{R}$ to the pair (A, ε) that is $(A, \varepsilon) \xrightarrow{\mathcal{M}} \mu$.
The metrics analyzed in this thesis are derived from the **confusion matrix** that describes the performance of a classification model on a set of test data for which the true values are known [12] [13]. The confusion matrix (or error matrix) is a contingency table used to summarize the performance of a classifier laying the fundamental foundations necessary to assess the performance of a specific classifier. It displays the actual class of the datapoints and the class predicted for them by the model in a matrix with columns and rows listing the number of instances according to the actual and predicted classes. It consists of a $N \times N$ matrix where N is the number of possible classes or outputs and is defined as

$$CM \equiv \begin{bmatrix} m_{11} & m_{12} & m_{13} & \dots & m_{1C} \\ m_{21} & m_{22} & m_{23} & \dots & m_{2C} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ m_{C1} & m_{C2} & m_{C3} & \dots & m_{CC} \end{bmatrix}$$

Here m_{ij} represents the number of elements actually belonging to the i-th class (θ_i) but that are classified as belonging to the j-th class (θ_j). In the binary case, the confusion matrix can be written as

$$CM \equiv \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}$$

Where $m_1 1$ and $m_2 2$ are the numbers of correctly predicted datapoints and $m_1 2$ and $m_2 1$ are the numbers of misclassified elements.
In the common terminology, the two classes of the binary case are referred to as the "Positive" and "Negative" class so that the confusion matrix can be rewritten as

$$CM \equiv \begin{bmatrix} m_{PP} & m_{PN} \\ m_{NP} & m_{NN} \end{bmatrix}$$

Usually the matrix is represented in a more intuitive visualization facilitating the comparison between actual and predicted values

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	True Positives (TP)	False Negatives (FN)	A_P
	Negative	False Positives (FP)	True Negatives (TN)	A_N
Estimations	Total	ε_P	ε_N	N

It can be seen how every column of the matrix corresponds to a category of predicted values (ε_P, ε_N) and these predictions are distinguished into those that are correctly predicted (correspond with the actual class) and those that are wrongly predicted (do not correspond with the actual class). Hence, every row corresponds with the actual instances (A_P, A_N). As a result of this structure, correct values are always placed on the leading diagonal of the matrix, while false predictions are placed on the antidiagonal.
To provide an example, following up on the application to the classification of a patient, consider a classifier that discriminates between ill and healthy state of each individual. There are two types of labels that are "ill" and "healthy" respectively encoded into 1 (Positive) and 0 (Negative). A sample of 1015 people is considered, out of which 235 are ill and 780 are healthy. After fitting the model, the classifier correctly classified 168 ill and 709 healthy subjects. On the other hand, it misclassified 71 ill individuals and 67 healthy persons. From these results, the consequent confusion matrix is

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	168	67	235
	Negative	71	709	780
Estimations	Total	239	776	1015

2.2 Concept of class of TP, TN, FP, FN

In binary classification predicted values are described as either Positive and Negative and True or False. As anticipated, "Positive" and "Negative" are used to distinguish the two

classes, one is considered as positive and the other as negative. This choice depends on the objective of study but is generally arbitrary. In order to process the model, these two classes gets encoded into numerical values, here with $Y_i = \{0, 1\}$. As for "True" or "False", this refers to the ability of the algorithm to correctly identify the actual class.
Hence, after applying a classifier to the dataset D, it results partitioned into four non overlapping subsets:

$$\begin{aligned} \mathcal{D} = &\{s \in \mathcal{D} | \text{actual class}(s) = 1 \text{ and predicted class}(s) = 1\} \cup \\ &\{s \in \mathcal{D} | \text{actual class}(s) = 0 \text{ and predicted class}(s) = 0\} \cup \\ &\{s \in \mathcal{D} | \text{actual class}(s) = 0 \text{ and predicted class}(s) = 1\} \cup \\ &\{s \in \mathcal{D} | \text{actual class}(s) = 1 \text{ and predicted class}(s) = 0\} \\ = &\, \text{TP} \cup \text{TN} \cup \text{FP} \cup \text{FN}. \end{aligned} \tag{2.1}$$

Samples belonging to:

- TP are called *true positive*. This number represents the cardinality of the set of datapoints for which the predicted label is positive and this coincides with the actual class (es. the subject is sick and the model accurately reports this).
- TN are called *true negative*. This number represents the cardinality of the set of datapoints for which the predicted label is negative and this coincides with the actual class (es. the subject is sick and the model accurately reports this).
- FP are called *false positive* (or *Type I error* in statistics - i.e. incorrectly rejecting a true null hypothesis). This number represents the cardinality of the set of datapoints for which the predicted label is positive when it is in fact negative (es. the subject is not sick, but the model inaccurately reports that they are).
- FN are called *false negative* (or *Type II error* in statistics - i.e. failing to reject a false null hypothesis). This number represents the cardinality of the set of datapoints for which the predicted label is negative when it is in fact positive (es. the subject is sick, but the model inaccurately reports that they are not).

2.3 Performance metrics

The confusion matrix is a tool to illustrate the result of the application of a model to a dataset. However, by itself it does not allow the discrimination between models (training stage) and the evaluation of the performance of the classifier (testing stage). Hence, the information contained in the confusion matrix needs to be encapsulated in a single figure to be used as a metric to assess the quality of the learned methods. Such metric represent a measurement tool that measures the performance of the classifier and is generally computed by calculating ratios between the entries of the confusion matrix categories.

In particular, to deliver more information each category of the confusion matrix (TP, TN, FP, FN) should not be evaluated independently but rather with respect to the other entries. This should occur with a function summarizing the joint behavior:

$$f\colon M(2\times 2,\mathbb{N}_0)\to D\subseteq\mathbb{R} \tag{2.2}$$

$$\begin{bmatrix}\mathrm{TP} & \mathrm{FN}\\ \mathrm{FP} & \mathrm{TN}\end{bmatrix}\mapsto f\left(\begin{bmatrix}\mathrm{TP} & \mathrm{FN}\\ \mathrm{FP} & \mathrm{TN}\end{bmatrix}\right) \tag{2.3}$$

where $D=[0,1]$.

This function can take the shape of several different performance metrics that have been introduced in the literature with the goal of selecting the best model, being it best in general or for a specific application area [14] [15] [16] [17] [18]. This section presents the most adopted metrics based on the confusion matrix used to assess the quality of binary classification algorithms.

Such metrics can be basic confusion matrix rates involving only two confusion matrix entries as the case of the following metrics. Here, each of the performance measures is illustrated, reporting the equation as well as the entries of the matrix that are relevant for the computation colouring with green the entries that should be maximized (vs red) and in a more marked shade to emphasize the focus of the metric (vs subtler shade).

- **True Positive Rate (TPR)**

  The TPR, or Recall (REC) or Sensitivity (SENS) or Hit Rate (HR), is the result of

  $$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}=1-\mathit{FalseNegativeRate}(\mathrm{FNR}) \tag{2.4}$$

  It's the proportion of correctly predicted positive datapoints (i.e. how many positive cases out of all the truly positive ones the algorithm has been able to detect) [19].

  It ranges from 0 to 1, respectively the worst and best case.

  One should prioritize sensitivity over other metrics when the goal is to identify all elements with a given characteristic in the dataset (es. detection of deadly diseases in a population). However, this metric suffers of one shortcoming that is not accounting for indeterminate results.

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **True Negative Rate (TNR)**

  The TNR, or Specificity (SPEC) or selectivity, is the result of

$$\frac{\text{TN}}{\text{FP} + \text{TN}} = 1 - FlsePositiveRate(\text{FPR}) \tag{2.5}$$

It's the proportion of negative datapoints that are correctly predicted as negative (*i.e.* how many negative cases out of all the truly negative ones the algorithm has recognized).

It ranges from 0 to 1, respectively the worst and best case.

One should prioritize specificity over other metrics when the goal is to accurately identify people who do not have a condition rather than those who have it (es. recognition of diseases for which there is great stigma).

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **Positive Predictive Value (PPV)**

  The PPV, or Precision (PCR), is the result of

$$\frac{\text{TP}}{\text{TP} + \text{FP}} = 1 - FalseDiscoveryRate(\text{FDR}) \tag{2.6}$$

  It's the proportion of true positive in the total of positive prediction (i.e. how many are correct out of all the positive prediction).

  It ranges from 0 to 1, respectively the worst and best case.

  One should look for high positive predictive value especially when there are great consequences for being classified as positive (es. illness detection).

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **Negative predictive value (NPV)**

  The NPV is the result of

$$\frac{\text{TN}}{\text{FN} + \text{TN}} = 1 - FalseOmissionRate(\text{FOR}) \tag{2.7}$$

  It's the ratio of correctly predicted negatives with respect to the total of negative predictions (*i.e.* how many are correct out of all the datapoints predicted as negatives).

It ranges from 0 to 1, respectively the worst and best case.
One should prioritize a high value of Negative Predictive Value when it's important that the subject is reassured about the result of the classification (*i.e.* assurance in testing negative for a deadly disease).

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

These above are the four *basic confusion matrix rates* that are particularly useful to synthetize the results. If the focus is more on false results, then one can look at the following counterparts:

- **FNR**
  TheFNR, or Miss Rate (MR), is the result of

$$\frac{\text{FN}}{\text{FN} + \text{TP}} = 1 - \text{TPR} \tag{2.8}$$

  It's the proportion of all positives that still yield a negative outcome (*i.e.* the probability of a negative prediction given a positive datapoint).
  It ranges from 0 to 1, respectively the best and worst case.
  A low False Negative Rate means that the researcher can be more confident to actually detect the phenomenon associated with the positive label.

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **FPR**
  The FPR, or Fall Out Rate , is the result of

$$\frac{\text{FP}}{\text{FP} + \text{TN}} = 1 - \text{TNR} \tag{2.9}$$

  It's the proportion of all negatives that still yield positive test outcomes (*i.e.* the probability of a positive result given a negative datapoint)

It ranges from 0 to 1, respectively the best and worst case.
This metric is particularly relevant in the cases in which incorrectly identifying an element as positive is more problematic than not recognizing it. For instance, in *spam detection* classifying an email as spam causing the receiver not to read the email is considered a worst situation than that in which few spam emails arrive in the inbox of the receiver. In such cases, the focus is on minimizing false positives.

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **FDR**

$$\frac{\text{FP}}{\text{FP} + \text{TP}} = 1 - \text{PPV} \tag{2.10}$$

It's the proportion of actually negative positive predictions over all positive predictions.
It ranges from 0 to 1 being respectively the best and worst case.
A low False Discovery Rate means that the researcher can be more confident that when an element is classified as positive, it is actually positive.

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **False omission rate** (FOR)

$$\frac{\text{FN}}{\text{FN} + \text{TN}} = 1 - \text{NPV} \tag{2.11}$$

It's the proportion of actually positive negative predictions over all negative predictions.
It ranges from 0 to 1 being respectively the best and worst case.
A low False Omission Rate means that there is a small proportion of actually positive elements being predicted as positive.

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

These measures are useful to grasp the behavior of the model but they cannot be considered fully informative as they use only two matrix entries. The more categories are involved in the computation of the metric, the more complete and informative the metric and thus the ability of the researcher to correctly interpret the performance of the model. Hence, more comprehensive indicators have been developed involving 3 or all 4 confusion matrix categories:

- **Prevalence Threshold (PT)**
  The results for the same model are strongly affected by the Prevalence (ϕ), that is the relative presence of positive cases in the test set. [20] This factor is independent from the classifier and is bound to the characteristics of the dataset. It is defined as:

$$\phi = \frac{\mathrm{TP} + \mathrm{FN}}{N} \tag{2.12}$$

  where N is the cardinality of the test set. Depending on the prevalence of positive cases in the dataset, the Positive Predictive Value will change. In particular, for the same learned model, the fewer the positive cases in the dataset, the smaller the positive predictive value and hence the worst the judged performance. Hence, the need for a metric that controls for this interdependence between prevalence and the rate of false positives. The prevalence threshold represent the prevalence level below which the rate of false positive outcomes (and the FDR) increases and the PPV decreases more strongly relative to disease prevalence. [21]

$$\frac{\sqrt{\mathrm{TPR} \cdot \mathrm{FPR}} - \mathrm{FPR}}{\mathrm{TPR} - \mathrm{FPR}} \tag{2.13}$$

  Essentially it tries to find a balance between the FPR and the TPR.

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **Critical Success Index (CSI)**

  The CSI, or Threat Score (TS), is the result of

$$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}+\mathrm{FP}} \tag{2.14}$$

  It's the ratio between the number of correctly identified positive elements and the total number of actually positive elements plus the false alarms. [22]
  It ranges from 0 to 1 with the latter being the optimal value.
  The CSI is not affected by correct rejections emphasizing more the importance of managing false alarms. Since the CSI is dependent upon the frequency of the event, an unbiased version has been developed called the Gilbert Skill Score (GS).

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **Accuracy (ACC)**

$$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{FP}+\mathrm{FN}+\mathrm{TN}} \tag{2.15}$$

  It's the proportion of correctly classified observations out of all the total number of samples.
  It ranges from 0 to 1, being the latter the optimal case.
  Accuracy is deemed as a reasonable performance metric since it keeps into account all 4 entries of the confusion matrix. However, there is a relevant amount of literature highlighting its shortcomings [23] [24] [25] [5]. Its weakness arises in the presence of imbalanced datasets, that is datasets where the number of observations belonging to one class is much higher than those belonging to the other (e.g. anomaly detection). In these cases, accuracy tends to overoptimistically estimate the classifiers' ability to predict the class that compose the majority of the dataset [26] [27].

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **Balanced Accuracy (BA)**

  BA attempts to tackle the issue with imbalanced datasets that accuracy suffers from. It is defined as

$$\frac{\text{TPR} + \text{TNR}}{2} = \frac{\frac{\text{TP}}{\text{TP+FN}} + \frac{\text{TN}}{\text{TN+FP}}}{2} \tag{2.16}$$

  and it's the arithmetic mean of sensitivity and specificity calculating the average of the correctly identified proportion of individual classes.

  It ranges from 0 to 1 being respectively the worst and best value.

  The added value of balanced accuracy is that it overcomes the trade off between sensitivity and specificity and can be applied when they are both equally important. Moreover, it accomodates for datasets in which the classes are imbalanced [28].

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **F1 score/ F1 measure**

  If BA tries to find a balance between sensitivity and specificity, the F1 score tries to balance sensitivity with recall. In fact, it is defined as

$$\frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} = 2 \times \frac{\text{PPV} \times \text{TPR}}{\text{PPV} + \text{TPR}} \tag{2.17}$$

  and it's the harmonic mean of precision and sensitivity [29].

  It ranges between 0 and 1, respectively worst (all positive samples are misclassified, TP=0) and best case (perfect classification, FN=FP=0).

  Essentially its advantage is that it attempts at summarizing the exactness (precision) and completeness (recall) of a model [29].

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **Bookmaker Informedness (BM)**

  The BM, or Youden's J statistic), is the result of

$$\text{TPR} + \text{TNR} - 1 \tag{2.18}$$

It represents the probability of an informed decision as opposed to a random guess allowing to evaluate the discriminative power of a test.
It ranges from 0 and 1, being the latter the ideal case.
Its added value is that, taking into account both real positives and negatives, it is a suitable performance metric also for imbalanced datasets.

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **Markedness (MK)**

$$\text{PPV} + \text{NPV} - 1 \tag{2.19}$$

It's the arithmetic mean of precision and negative predictive value measuring the trustworthiness of positive and negative predictions.
It ranges from -1 to 1 being respectively worst and best value.
The added value is that it considers both predicted positives and predicted negatives at the same time [30]. However, being sensitive to data changes, it is not suitable for imbalanced data [31].

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **Diagnostic Odd Ratio (DOR)**

$$\frac{\text{TP}/\text{FP}}{\text{FN}/\text{TN}} = \frac{\text{TPR} \cdot \text{TNR}}{(1 - \text{TPR}) \times (1 - \text{TNR})} \tag{2.20}$$

It is defined as the ratio of the odds of the classification being positive if the datapoint is actually affected by the phenomenon at test relative to the odds of the classification being positive if the datapoint is not actually affected by the phenomenon. (ratio of the positive likelihood ratio to the negative likelihood ratio- but then I should define the likelihood). [32]
It ranges between 0 and ∞ with 0 being the worst case (TP or TN or both are zero) and 1 being the case of random classification.

The rationale for the DOR is that it is a single performance metric (like accuracy and bookmaker informedness) but which is independent of prevalence (unlike accuracy) and it is generally more intuitive (odds ratios are more common). The shortcoming is that

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	TP	FN	A_P
	Negative	FP	TN	A_N
Estimations	Total	ε_P	ε_N	N

- **MCC**

$$\frac{Cov(c,l)}{\sigma_c \cdot \sigma_l} = \frac{\mathrm{TP} \cdot \mathrm{TN} - \mathrm{FP} \cdot \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}} \tag{2.21}$$

(with $Cov(c,l)$ being the covariance of the true classes c with the predicted labels l and σ_c and σ_l the standard deviations respectively of the actual and predicted classes).
It measures the correlation of the true classes c and predicted labels l.
It ranges from -1 and 1 being respectively the worst and best case.
The added value of MCC is that it is unaffected by the imbalance in the dataset and it's the only binary classification indicator that correctly predicts the majority of both the positive and the negative data instances [33][34]

In the literature other possible quantitative indicators can be found, such as mutual information [35], Cohen's Kappa [36], H measure [37] or K measure [38]. However, these have not been analyzed in this work because of their limited popularity among the scientific community.

No shared consensus has been reached on the elective function to employ in order to evaluate the model's discriminatory performance. Generally, the most common metric used for binary classification in machine learning are accuracy and F_1 score. However, despite their relative popularity among researchers, several papers highlight their shortcomings. [23],[24],[25]. Essentially, they tend to provide an unfair estimate of the models' performance in the case of imbalanced datasets because they do not consider the ratio between positive and negative elements.
As shown in the literature [39] [40] [41] [42], the MCC appears to be the preferable options. This is due to two main reasons:

1. it is more robust with imbalanced datasets providing a more fair estimate of the model discriminatory ability between classes (it generates a high score only if the classifier correctly predicted most of the positive data instances and most of the

negative data instances and if most of the its positive predictions and most of its negative predictions are correct);

2. it is more reliable since if the MCC is high then also the four basic confusion matrix rates (TPR, TNR, PPV, NPV) which is not always the case for other metrics.

2.4 Visualizations of the confusion matrices

As anticipated, the confusion matrix allows the researcher to display the results of a classification model. The matrix itself is not extremely informative about the performance of the model without an additional step, a modification of the confusion matrix. The need to extract the insights can be satisfied by *computing performance metrics* (as seen so far) that is by computing statistical rates among the entries of the matrix (and here the more the entries involved, the more truthful the picture delivered). However, the metrics further synthesise the information contained in the matrix and the indicator alone is not too intuitive, thus possibly leading to misunderstanding. Moreover, even after computing such metrics, the technical concepts need explanations. The interpretation is potentially error prone and misleading especially for beginners since different metrics can have different relevance according to the use case in which the model is applied (not immediate to know which metrics suit their use case). Hence, a useful strategy to be adopted in order to gain insights about the model's performance and to actually describe the quality of the prediction is to *find alternative visualizations* of the confusion matrix. Additional dataviz elements (such as distribution bars and difference in colours) or completely new designs could help evaluate the classification model and interpret the results. To illustrate these approaches we can follow up on the example of confusion matrix introduced in Section 2.1 that is reported here below.

		Predicted outcome		*Instances*
		Positive	Negative	Total
Actual values	Positive	168	67	235
	Negative	71	709	780
Estimations	Total	239	776	1015

This is the baseline visualization for the confusion matrix. The present section surveys techniques that seek to make the visualization of the information contained in the confusion matrix more comprehensible and effective. Here four main alternative designs are illustrated highlighting the perks and shortcoming of each proposal.

2.4.1 First approach

The first approach [43] [44] aims at improving the readability and comprehension of the confusion matrix by modifying the visualization of the matrix as it is. This approach suggests several manipulations to be applied to the core baseline confusion matrix. The resulting visualizations for this first approach inspire from the literature but the code has

been modified to enrich the result with features that were not originally included (*e.g.* colour, labels, metrics)
The contingency table can be turned into a heatmap so that each entry of the matrix gets associated to a different hue of colour (*see* Figure 2.1). This allows to give immediate visual cues about the magnitude of each subset of the contingency table.

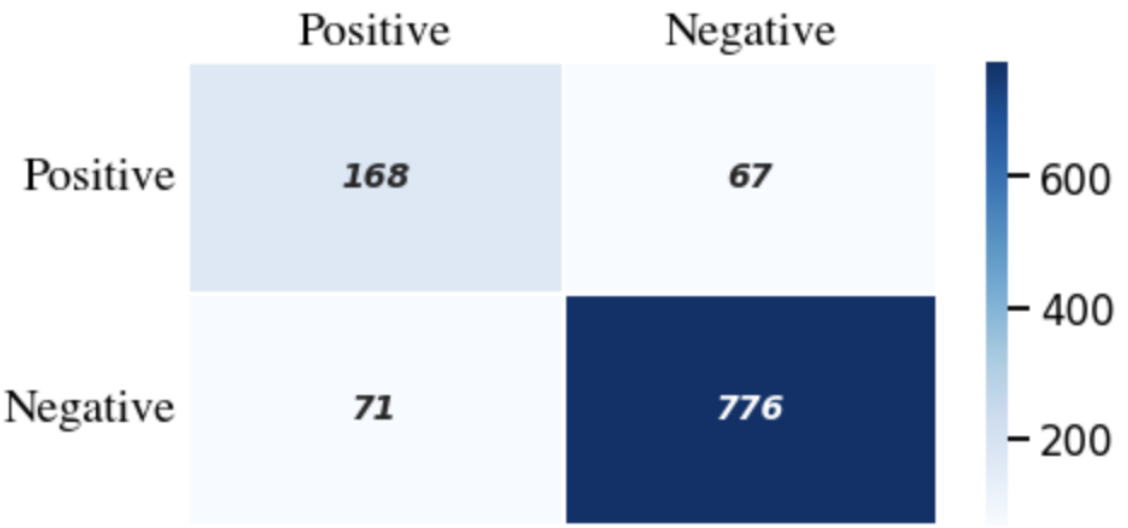


Figure 2.1: First approach: heatmap of the baseline confusion matrix

Secondly, the label for the category (TP,TN,FP,FN) can be added in each quadrant as well as the percentage that each category makes of the total.

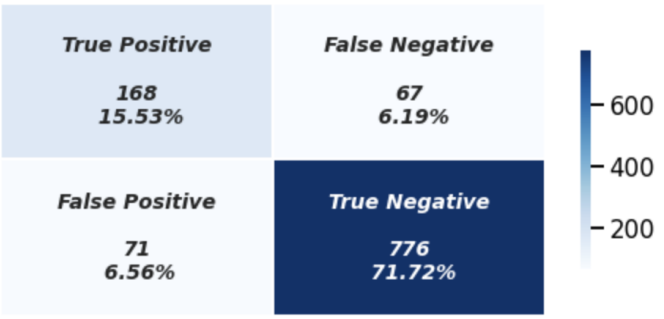
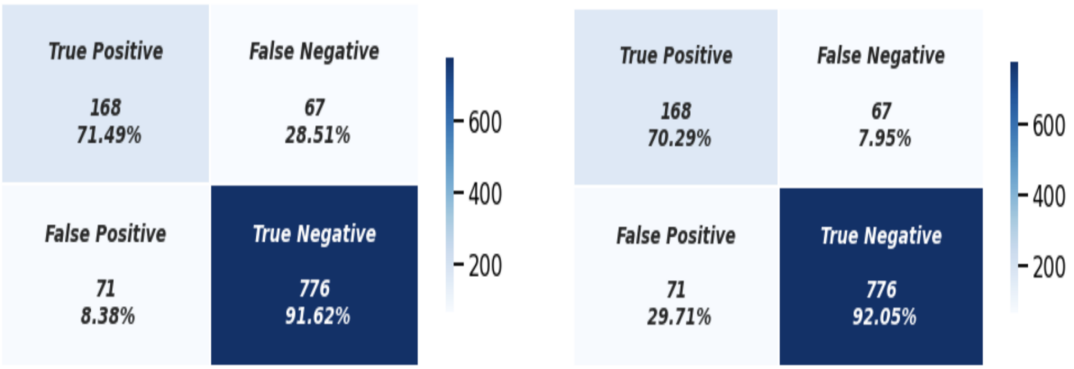


Figure 2.2: First approach: addition of labels and percentages

Alternatively, the percentage can be computed also column wise or row wise (*see* Figure 2.3) for a more focused understanding of the distribution. Lastly, the synthesizing performance metrics are added to the panel so to allow immediate evaluation of the quality of the model and the colour is distinguished according to the type of entry (*see* Figure 2.4). In this case, the four basic matrix rates (TPR, TNR, PPV and NPV) as well as the more comprehensive ones (*i.e.* accuracy and F_1 score) are added.



(a) row-wise normalization (b) column-wise normalization

Figure 2.3: First approach: normalization

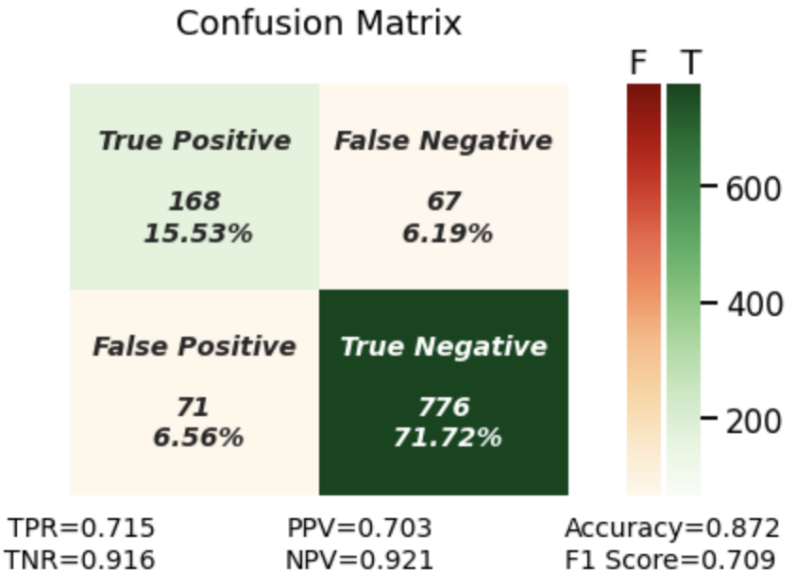


Figure 2.4: First approach: compact visualization

From this visualization one can conclude that the model performs well and it is accurate in predicting the data since there are low false predictions and all the performance metrics are close to 1. However, the color aids in noticing the large cardinality of true negatives which tend to increase accuracy possibly leading to an artificial overestimation of the quality of the model. This approach is relatively simple but still improves the representation of confusion matrices catering to the needs of researchers to represent the performance of classification algorithm. Although useful, this visualization is not groundbreaking and above all it does not aid in the interpretation of the metrics, it simply adds them to the visualization. Hence, the need to look for alternative designs.

2.4.2 Second approach

The manipulations of the first approach give a multifaceted portrait of a model's performance but they do not offer an effective visualization, showing all relevant information at a glance. Hence, the second approach [45] is one that adds to the single contingency table different additional components useful to illustrate and represent several concepts. In particular, this design allows for a more intuitive interpretation of the performance metrics by adding explanatory decorators for these rates.
The proposed visualization results [1] in a composition of the confusion matrix plus supplementary explanatory elements (*see* Figure 2.5).

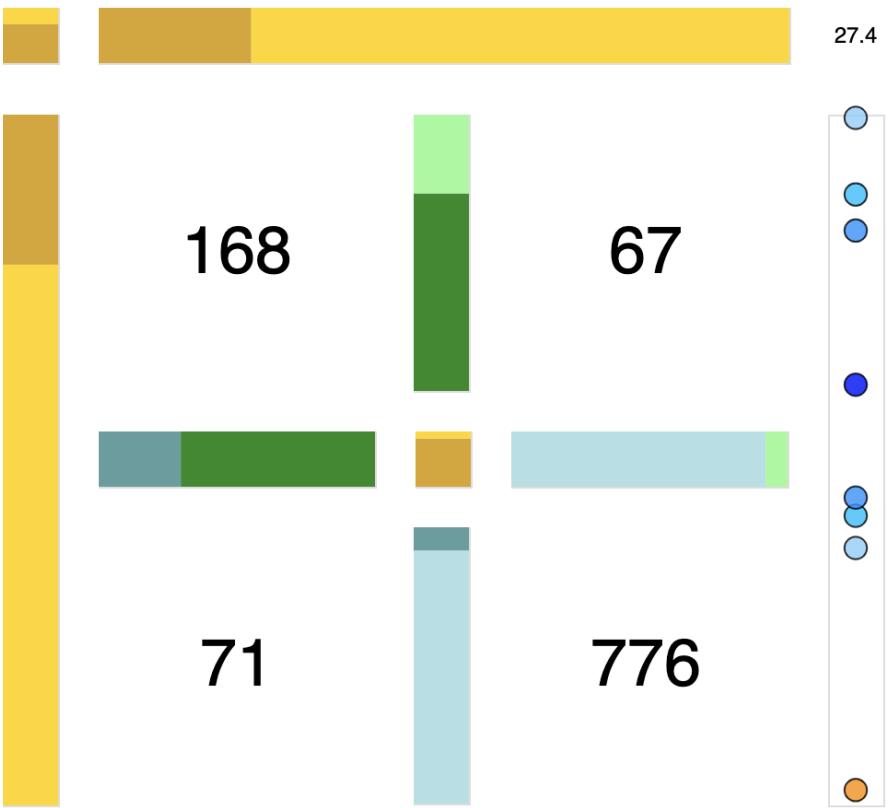


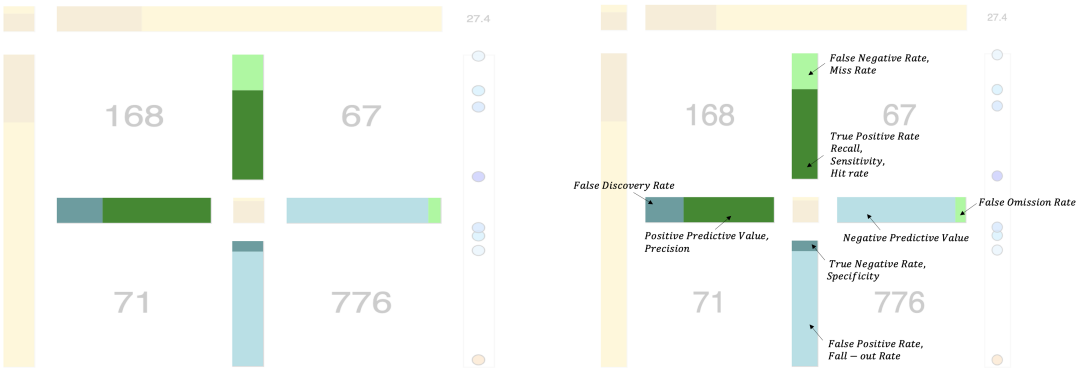
Figure 2.5: Second approach: composition with a number of decorators

The additional decorators, that allow for an easier understanding of the results of the model and a more intuitive interpretation of the performance metrics, are the following:

1. **Bars on the inside**
   The first decorator added to the visualization is a normalized stacked bar of the two adjacent numbers illustrating the ratio of each entry to the pairwise sum. Since each

[1] if testing the code, notice that it needs to be opened in the Vega Editor to have a proper visualisation of the result

of these *pair bars* is essentially a conditional probability of the adjacent numbers, it is computed with only two of the entries and hence it is placed in between the associated numbers (*see* Figure 2.6). For example, looking at pairwise bar between TP and FN (middle-top) one can appreciate their nature as conditional probabilities as the darker green color represents the conditional probability of being predicted positive when actually positive (y|Y) whereas the lighter shade for the False Negative Rate is the conditional probability of being predicted negative when actually positive (n|Y).



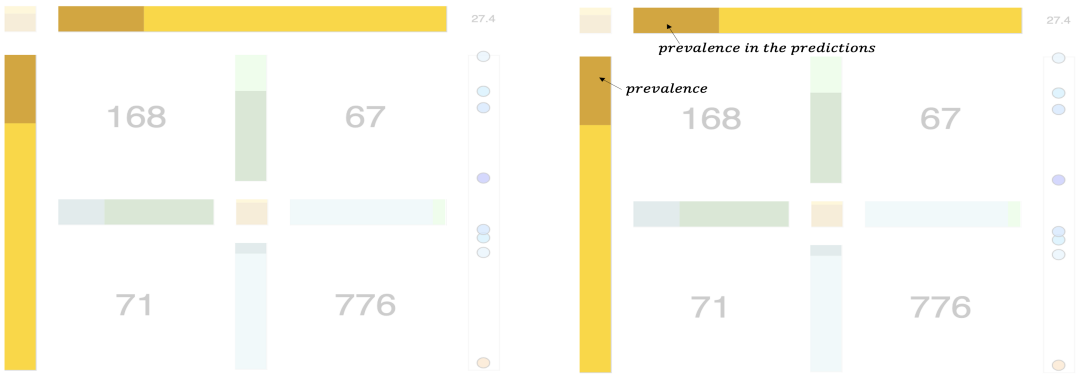
(a) bars placement (b) metrics

Figure 2.6: Second approach - First graphical element: bars for the pairwise ratio and the consequent visualisation of the performance metrics

This graphical approach allows the visualization the four basic confusion matrix rates (*TPR, TNR, PPV and NPV*) and their complement (*FNR, FPR, FDR* and FOR respectively). Notice that each section within the bar is ordered according to the convention that the segments that are closest to the center are those involving the numbers in the diagonal and those further from the center are the metrics that include the entries on the off diagonal. For example, again looking at the middle-top bar, the TPR, which includes TP, is placed closer to the center because TP are the entry of the diagonal. On the other hand, the FNR is closer to the edges as it is emphasizes the role of FN, which are on the off-diagonal.

2. **Bars on the outside**

   The second decorator is a bar to represent the prevalence of positive elements in the actual dataset as well as one that illustrates the same proportion among the predictions (*see* Figure 2.7). Hence, the golden bars on the left and at the top of the table communicate the row sums and the column sums of the baseline confusion matrix. Due to the involvement of all four matrix entries in their computation, the *golden bars* stands on the outside of the plain matrix.

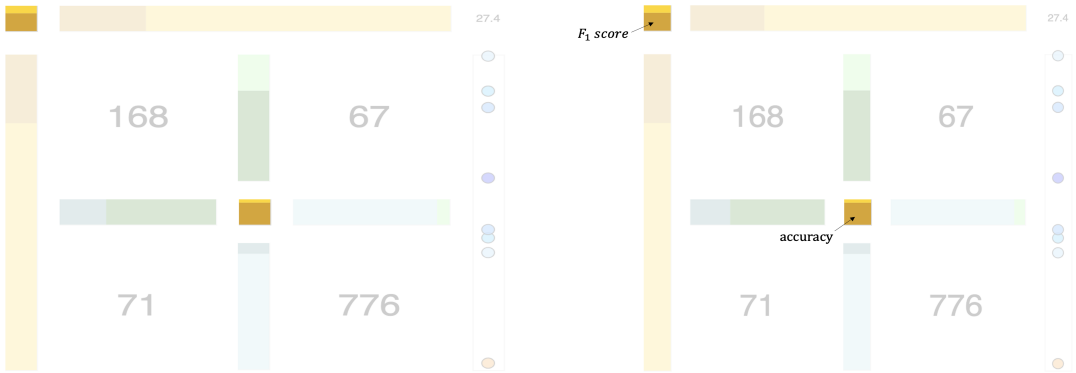


(a) bars placement (b) bars interpretation

Figure 2.7: Second approach - Second element: outside bars placement to display the prevalence in the among the actual records as well as the predictions

Following the standard for which the predictions are on the columns of the contingency table, the proportion of positives in the predictions is placed on top of the composition. Similarly, since in the baseline confusion matrix, actual values are on the rows, the prevalence in the input dataset stands vertically on the left encompassing all rows.

3. **Squares for** *accuracy* **and** F_1
   This proposal then adds a square for the performance metrics involving all four matrix entries that are *accuracy* and F_1 score (*see* Figure 2.8) .

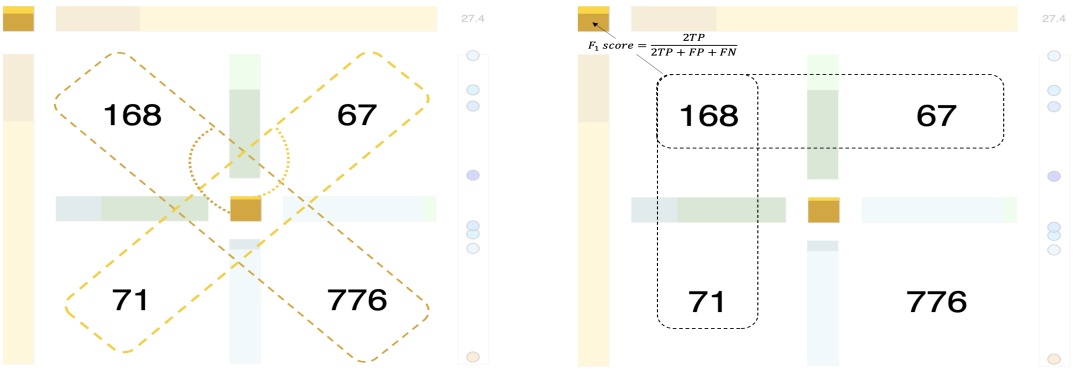


(a) squares placement (b) squares interpretation

Figure 2.8: Second approach - Third element: squares for performance metrics computed with more than two entries

The squares are placed coherently with the entries of the matrix that are included in the formula used to obtain the indicator (*see* Figure 2.9) As for accuracy, the associated square is placed in the center since this metric stems from a computation that involves all four entries (*see* Equation 2.15) hence a placement in the middle communicates its comprehensiveness.

As for the F_1 score, being the result of the two dark green segments, the square is placed on the top left corner, further from False Negatives (FN) due to their insensitivity to this entry (*see* Equation 2.17).



(a) accuracy computation (b) F_1 score computation

Figure 2.9: Second approach - Third element: further explanation of the squares linking the interpretation with the graphical choices

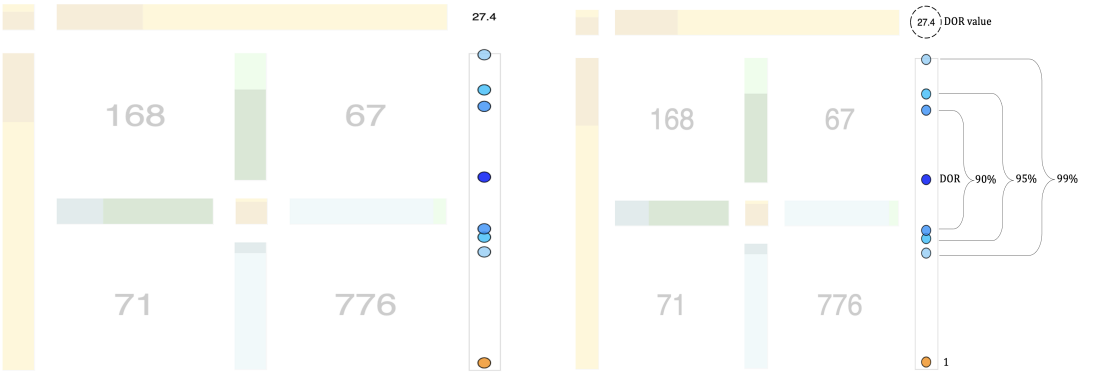
Similarly to the convention adopted for the pairwise bars, the component associated with the values on the diagonal is shaded with a darker colour (*see* Figure 2.9a)

Presenting both metrics is particularly useful in extremely imbalanced datasets since F_1 score, differently from accuracy, is sensitive to differences in the composition of the sample. With imbalanced datasets (*e.g.* many TN) accuracy could be misleading but having also the $F_1 score$ displayed in the visualization would permit to notice the imbalance issue affecting the performance of the model (*see* Figure 2.10).

Figure 2.10: Second approach - Third element: added value of displaying both metrics

4. **Bar for the odds ratio and confidence intervals**
   A fourth decorator is added on the right with a white square displaying the computed DOR and a bar reporting its confidence intervals at 90%, 95% and 99% (*see* Figure 2.11). Together with the dots for the confidence intervals (with a lighter blue shade

as the confidence decreases), the bar includes also the dot for a DOR equal to 1 (in a striking yellow shade) which would be the case of a random classification. Visualizing also the metric for the random classifier, helps the user to more intuitively understand how the model stands with respect to the baseline case and providing environmental information it better communicates the performance of the fitted model.



(a) DOR placement

(b) DOR interpretation

Figure 2.11: Second approach - Fourth element: DOR analysis

This graphical solution presents two main advantages:

1. it facilitates the understanding of the performance of the model, allowing to visually grasp the insights provided by the confusion matrix and the performance metrics;
2. one can intuitively conclude the dependency and association of classes through the odds ratio and confidence interval.

Essentially, it allows the researcher to display a considerable amount of information in a compact and intuitive way. Although the design is not extremely groundbreaking, the preservation of the foundations of the standard visualization should be seen as an added value. In fact, it allows researchers to operate with a familiar tool but with additional components that ease the interpretation of the performance metrics of the model.

2.4.3 Third approach

If these first two approaches emphasize more the graphical representation of the plain end result of the contingency table, the next two visual approaches tend to enter in the core of the classification process by emphasizing the focus on displaying the consequences of the choice for different thresholds.
This third approach, published in the literature in [46], aims at better visualizing the information included in the confusion matrix as different values of the classification threshold are selected. In fact, generative classifiers do not output automatically the predicted label but the probability that the records belongs to a certain class. Depending on the value of the threshold more or less records will be assigned to one or the other label. Hence, this third solution displays how the confusion matrix, which is essentially condensed into a stacked bar chart, changes according to different levels of the threshold (*see* Figure 2.12). True negatives are not included as they are not really relevant for the evaluation representing elements that are not affected by the phenomenon being investigated.

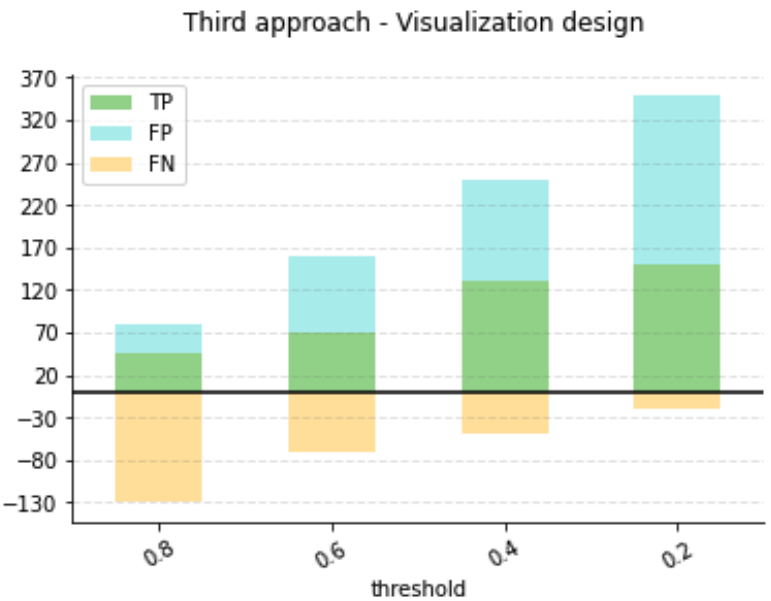


Figure 2.12: Third approach: overall design

This solution presents a completely different design that actually includes more than one single confusion matrix and starts the progression of the approaches towards visualisations that include different alternatives of the same model and that project the perspective on how one specific model stands with respect to other possible options.

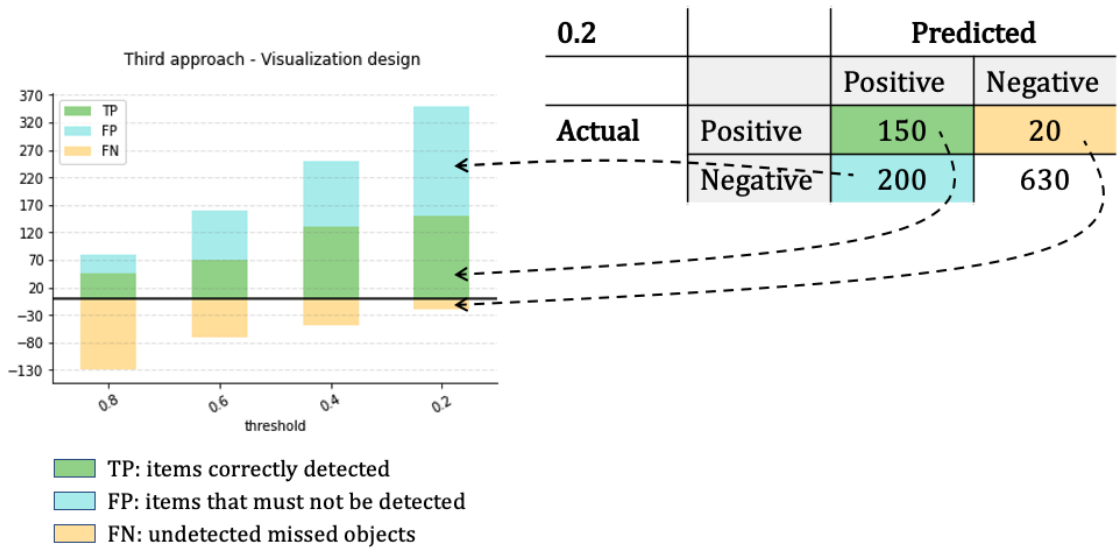
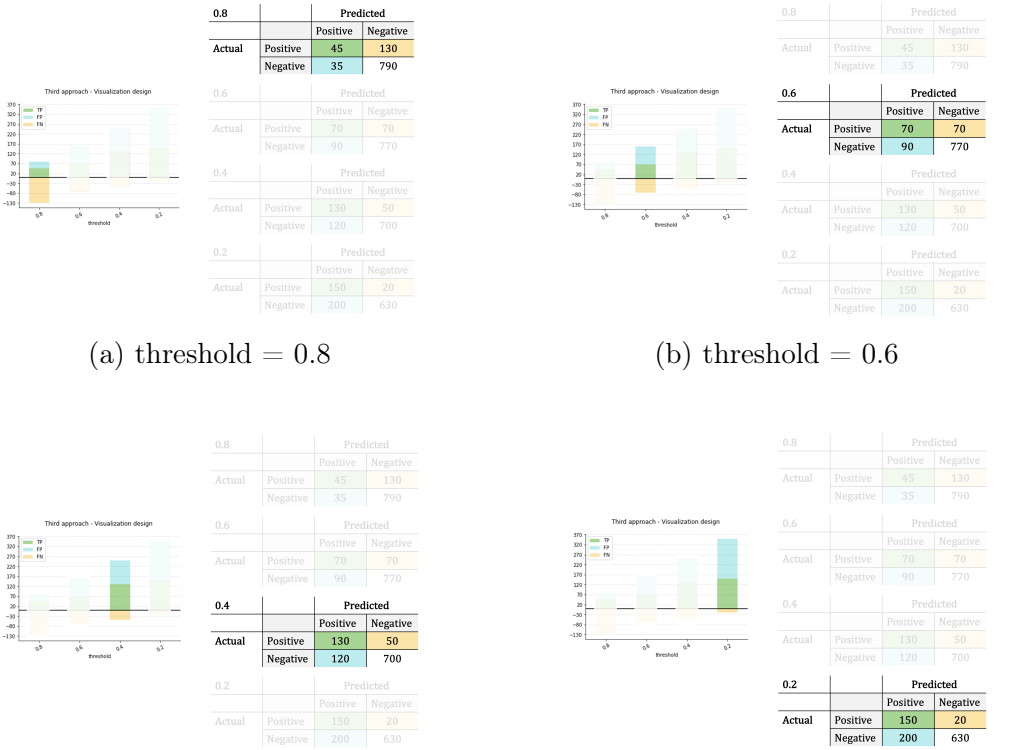


Figure 2.13: Third approach: composition of one single stacked bar chart

The composition stems from the collection of the confusion matrix resulting for each of the chosen threshold values which define the number of columns displayed in the graph. In this case (*see* Figure 2.13), the thresholds are selected at 0.8, 0.6, 0.4 and 0.2 to have an

heterogeneous view of the results. However, depending on the use case, one could focus on a wider or narrower range restricting the view to a specific focal point.
If Figure 2.13 illustrates the workflow for the visualisation of one single confusion matrix and hence the build up of one single stacked bar chart, in Figure 2.14 the step for each threshold is illustrated resulting in the final composition of the four stacked bar chart.



(a) threshold = 0.8

(b) threshold = 0.6

(c) threshold = 0.4

(d) threshold = 0.2

Figure 2.14: Third approach: breakdown of the graphical composition

This solution represents a simple visual encoding with the perks of:

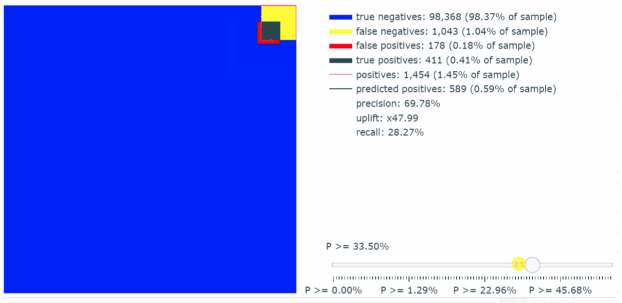
1. avoiding excessive synthesis of the information contained in the matrix while at the same time eliminating the distorting effect of true negatives;
2. reading the matrix both column and row-wise;
3. personalising the output through the choice of the threshold tailoring the range to one's own specific application scenario;
4. allowing to appreciate the evolution of the matrix as different probability thresholds are selected.

On the other hand, being more focused on the display of the evolution of the outcome according to the thresholds, it does not allow an intuitive interpretation of the performance

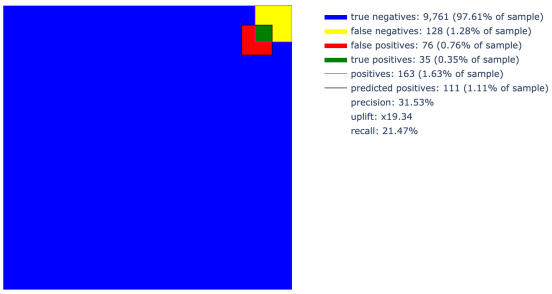
for one single model. It consists more in a different visual encoding of the baseline matrix but no summary statistic is included and no active aid is provided to the user who needs to assess the model. Also, while attempting at tapping into the dimension of visualising more than one confusion matrices, the composition is relatively static preventing from an intuitive interpretation of the results.

2.4.4 Fourth approach

Other graphical approaches can be found in the web [47], with particular regard to data science portals such as Medium [2] or Towards Data Science [3]. The fourth approach has been found in such platforms and aims to offer a single graphical representation able to deliver all the relevant information with respect to the performance of a binary classification task. What makes it particularly worthy is the level of interactivity achieved as it consists in a dynamic visualization of the outcome of any binary classifier that synchronously adapts as the value of the threshold is modified (*see* Figure 2.15a). Once a threshold is chosen, in its static view the composition displays a two dimensional plot where each of the areas refer to TP,FP,TN and FN and these change proportionally to the cardinality of each matrix entry as resulting from the threshold selected (*see* Figure 2.15b).



(a) Interactive visualisation



(b) Composition of each static view

Figure 2.15: Fourth approach: single graphical and interactive visualisation

The composition presents several elements that when combined together serve the ultimate purpose of facilitating the understanding of the model performance. In particular, the

[2]Medium - `medium.com`
[3]Towards Data Science - `towardsdatascience.com`

arrangement merges the following elements:

- *a 2D graph* where the areas for each matrix entry are colored with different shades and their size synchronously changes with modifications in the probability threshold;
- *a slider* that allows to more seamlessly change the threshold for the probability of belonging to a certain outcome class ($P > threshold$). As one moves the slider from the left to the right the associated threshold increases (i.e. probability of belonging to the positive class) and the selection becomes more restrictive;
- *the list* of the actual values for each matrix entries together with their percentage with the respect to the total;
- the values for *precision* and *recall* to have a more immediate look at the effect of changes on the Precision-Recall tradeoff. In fact, as the slider is moved to the right, precision improves but recall decreases. The more one is asking to be sure of the positive label the smaller the fraction of positives identified (i.e. lower recall) but for those identified there is less risk of a wrong prediction (i.e. higher precision)
- the value of the *uplift* which is the improvement in the precision with respect to the random case as different values for the threshold are selected.

By breaking down each area of the composition one can read several informations. First of all, focusing on the two dimensional plot, one can read the total number of actually positive elements as given by the square with pink edges (Figure 2.16). Being the sum of the actually positive elements (*i.e.* TP+FN) it reads the confusion matrix row-wise and visualises the sum as an area of the two dimensional plot. Notice, however, that the area is not filled, only the perimeter is colored as it is a sum of more than one matrix entry.

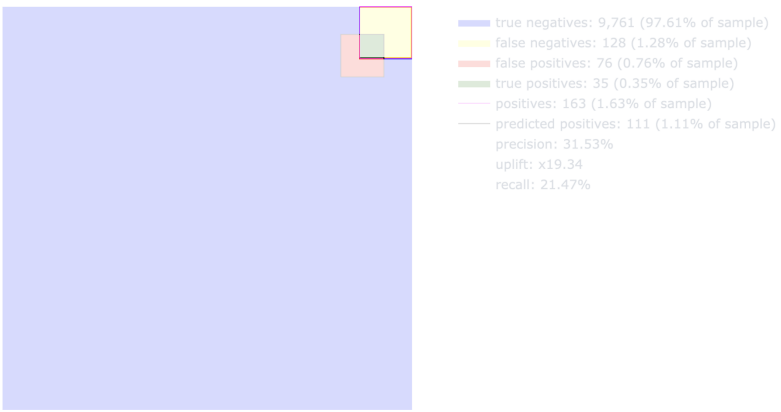


Figure 2.16: Actually positive elements

Similarly, it reports the total number of predicted positive elements (i.e. $TP + FP$) through a square with black edges (Figure 2.17). In this case, the baseline confusion matrix is read

column wise and the sum is visualised as an empty square. Similarly to the previous case, being the sum of more than one matrix entry, the emphasis is on the border not the area. Their areas are then colored according to the correctness of the prediction.

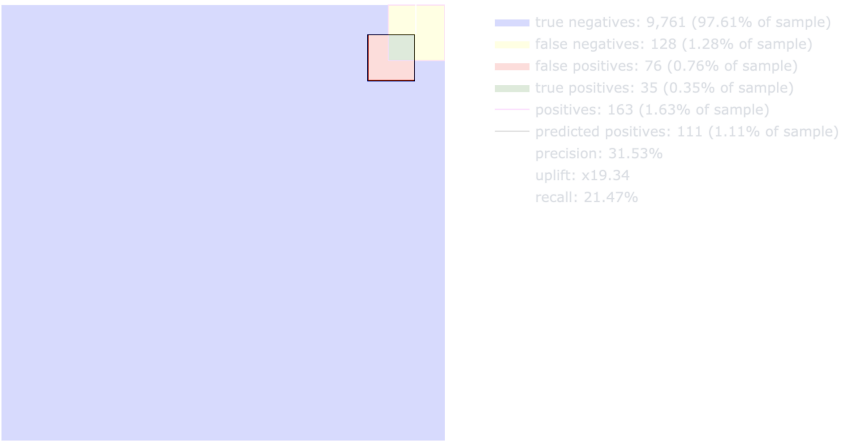


Figure 2.17: Fourth approach: predicted positive elements

The black square intersects the actually positive elements and hence, by combining the two previous information, it follows that the filled green square represents one specific matrix entry that is the number of True Positives. This value is given by those elements that are actually positive elements (*i.e.* within the pink empty square), are predicted as positive (*i.e.* within the black empty square) and the prediction is correct (*i.e.* green filled area) (*see* Figure 2.18).

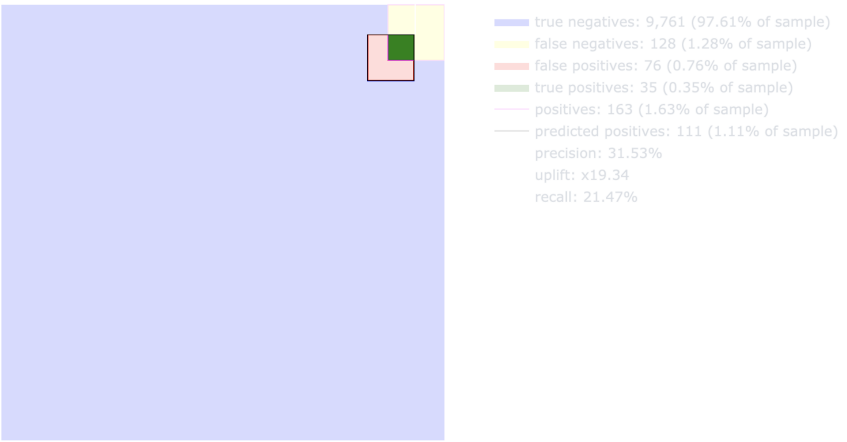


Figure 2.18: Fourth approach: true positives (TP)

It follows that the complement of the green area (TP) within the black square (*i.e.* predicted positives) is the red shape (*see* Figure 2.19) which represents the number of false positives (FP). This value is given by those elements that are actually negative elements (*i.e.* outside

the pink empty square), are predicted as positive (*i.e.* within the black empty square) and for which the prediction is wrong (*i.e.* red filled area).

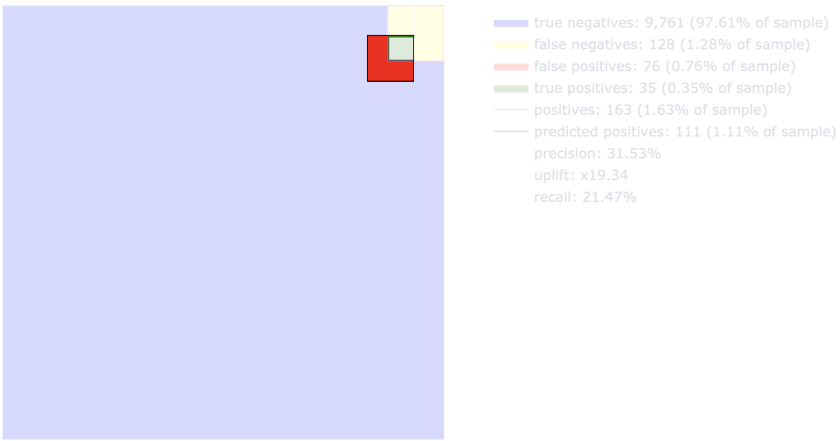


Figure 2.19: Fourth approach: false positives (FP)

Similarly, the complement of the green filled area (TP) within the pink square (i.e. actually positives) is the yellow shape (*see* Figure 2.20) which represents the number of false negatives (FN). This value is given by those elements that are actually positive elements (*i.e.* within the pink empty square), are predicted as negative (*i.e.* outside the black empty square) and for which the prediction is wrong (*i.e.* yellow filled area).

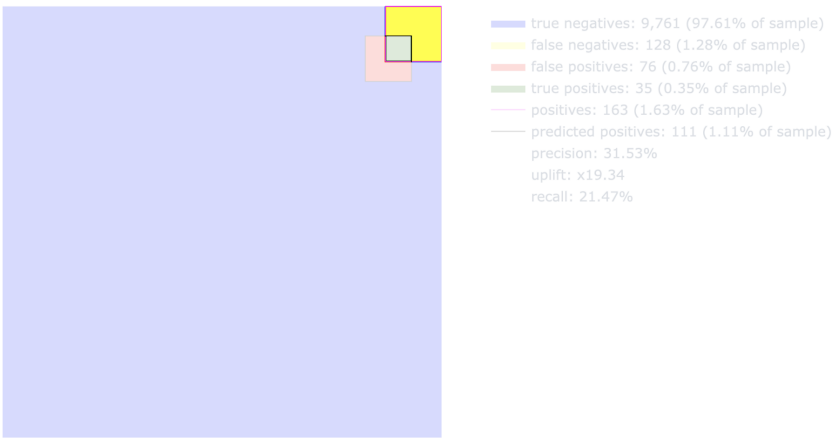


Figure 2.20: Fourth approach: false negatives (FN)

Lastly, as a result, the remaining of the four matrix confusion entries, that is the number of true negatives (TN), is encapsulated in the blue shape (*see* Figure 2.21). Hence, the whole square includes all records in the dataset and each single element will be attributed to the coherent shape according to the actual and predicted value. It results that one can have a clear snapshot the magnitude of each entry with respect to the total and this dynamically adapts to the choice of the threshold.

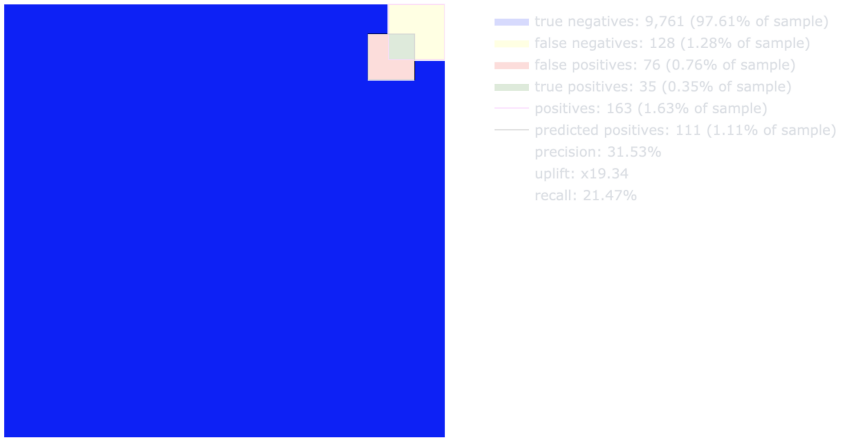


Figure 2.21: Fourth approach: true negatives (TN)

As anticipated, together with the two dimensional plot, a list of values and performance metrics is provided. Partially, this takes the form of a legend reporting the colour used for each area together with the actual number of the matrix entry and the percentage with respect to the total cardinality of the dataset.

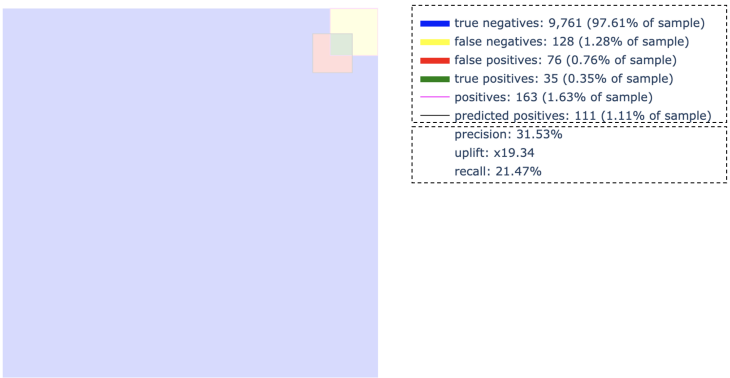
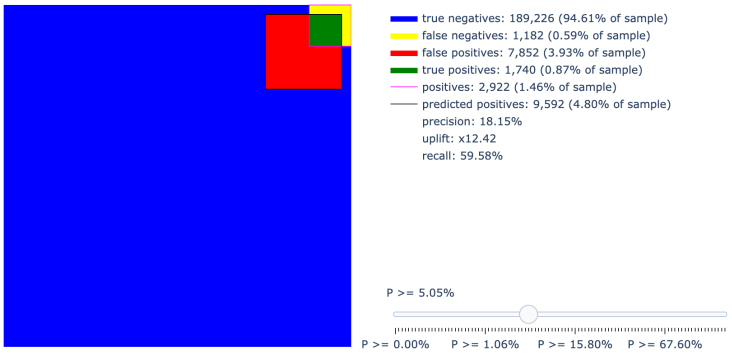
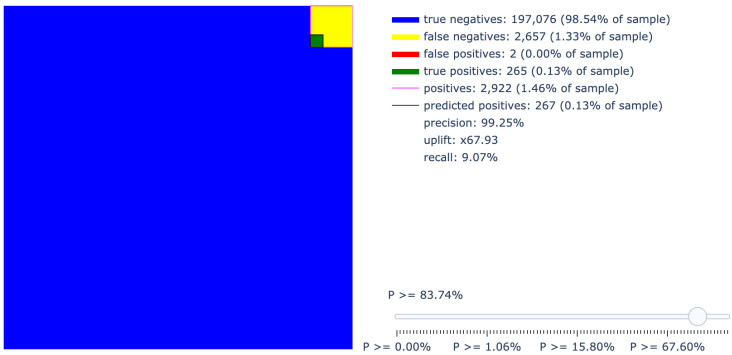


Figure 2.22: Fourth approach: legend and performance metrics (TN)

Besides the composition of the two dimensional plot, what is striking about this approach is its interactivity in the visualisation of the results. As the third approach, its main purpose is to facilitate the understanding of the behavior of the classification as different levels of thresholds are chosen.



(a) Fourth approach: result for the selection of a lower threshold



(b) Fourth approach: result for the selection of a higher threshold

Figure 2.23: Fourth approach: different choices of thresholds synchronously modify the plot

To this end, the implementation in Plotly offers the possibility to represent in one single plot the different confusion matrices through the slider that controls the choice of the probability threshold. In fact, the classification label for each single record is not the primary outcome of a classification algorithm. Instead, the model attribute to each element a probability of it belonging to the positive class. Hence the output is not the actual label but a number ranging from 0 to 1. To obtain the actual classification (*i.e.* binary label), one has to define how likely the predicted class for the record should be. This value is set harnessing the knowledge of the specific application scenario but since there is no predefinite rule it is useful to have a tool that helps in visualizing the consequences of variations in the choice of the threshold. By looking at Figure 2.23, one can notice how, as the slider is moved to the right, the threshold increases so the probability requested for the record to be classified as positive gets higher with the result that less elements will be predicted as positive. The smaller fraction of positives predicted is visualised through a smaller black square (*see* Figure 2.23b) and these changes are reflected also by the metric (*i.e.* higher precision, lower recall). On the other hand, the lower the threshold, the lower the probability requested to classify an element as positive and hence more records are falsely predicted as positive. The

greater fraction of wrong positives predictions is visualised through a growing red square (*see* Figure 2.23a) and changes in the metrics (*i.e.* lower precision, higher recall).

To sum up, this is a graphical solution that allows for great interactive and presents the following pros:

1. including a visual encoding of both the confusion matrix entries as well as summary metrics that dynamically change communicating the effect of different choices of the threshold;
2. allowing for great personalization and interactivity providing the model evaluation with a tool that empowers the researcher to critically reflect on the choice of the threshold suitable to one's own specific application scenario;
3. condensing the visualisation of more than one confusion matrix simultaneously and hence embarking on the path of visualizing how one specific confusion matrix stands with respect to other possible options.

On the other hand, the flaw is that the performance metrics displayed are few and fixed. It would be advisable to have additional metrics and especially those that are more comprehensive. Moreover, to maximize the empowering of the evaluation, it would be preferable to allow for the choice of the performance metric with a widget providing a list of accepted metrics. Moreover, while plotting several confusion matrices, this graphical approach does not tap into the dimension of the learning phase of the model resulting in a relatively nice but not so actionable visualisation. These shortcoming are the dimensions that the confusion tetrahedron attempts at tackling with its 3D interactive visualisation.

3 The confusion tetrahedron

3.1 Shortcomings of classical approaches

The current design of the dataviz solutions for visualizing confusion matrices are valuable as they offer simple visual encodings for the information contained in the table as well as additional decorators that allow for a more immediate interpretation of the performance metrics. Moreover, all the illustrated classical approaches are independent from the model, they can be adopted as long as a confusion matrix is obtained.
However, these approaches are subject to several shortcomings that the proposed novel approach sets to overcome. These limitations are interlinked but can be summarised into the following:

- *Bidimensionality*
  The four classical approaches share a common property that is the bidimensionality of the visualization. The first two methods keep the original design of the baseline contingency table while adding several decorators that aid the interpretation. The third and the fourth dare to change the design from its root, with the fourth providing even more interactivity for the choice of the threshold. However, all four are bidimensional and relatively static representations of the information provided by one or a set of confusion matrices. This fact represents an issue because such features are not advisable for graphical approaches that intend to aid in the interpretation and the understanding of the evaluation of the results. In order to have a better understanding of the model, it would be preferable to represent the learning process within a geometrical environment that gives a sense of where the model stands with respect to the other possible outcomes.

- *Relative loss of information*
  Due to the bidimensionality and the specific choices of each design, all approaches tend to flatten the information provided to the researcher evaluating the models. This is particularly true for the first and second approach where explanatory decorators are added to the composition but one single confusion matrix is still being plotted. However, it is true also for the third and fourth approach. In fact, while offering the visualisation of more than one confusion matrix at a time (as resulting from different choices in the threshold), they still perform restrictive choices (e.g. the performance metric computed in the uplift for the fourth approach or the values of the threshold

selected in the third approach among the others) and hence reducing the information provided.

- *Low interactivity*
  The classical approaches tend to be static representations of the performance without any opportunity to modify or actively interacting with the visualisation. This reduces the chances of actually engaging the user which is a particularly severe problem given that machine learning applications are increasing in popularity and hence a broad audience of non technical stakeholders will have to deal at least with the evaluation stage of the algorithms.

The novel approach of the Confusion Tetrahedron aims at overcoming these three limitations with an innovative geometrical plot of the environment in which the binary classification models learn.

3.2 The Confusion Tetrahedron: theoretical explanation

Consider a generic confusion matrix $CM \equiv \begin{bmatrix} \mathrm{TP} & \mathrm{FN} \\ \mathrm{FP} & \mathrm{TN} \end{bmatrix}$ given that $\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN} = S$.
The objective is to obtain a tool for the evaluation of the performance of a classifier. To evaluate usually means to compare (among available alternatives) but for the comparison not to be biased, the geometrical environment needs to be *scale independent* (*i.e.* independent from the scale of the system).
Hence, to enable the evaluation, the values are *normalized* dividing by S (*i.e.* the number of samples in the dataset) obtaining the Normalized Confusion Matrix (nCM)

$$\mathrm{nCM} \equiv \begin{bmatrix} \frac{\mathrm{TP}}{S} & \frac{\mathrm{FN}}{S} \\ \frac{\mathrm{FP}}{S} & \frac{\mathrm{TN}}{S} \end{bmatrix} \tag{3.1}$$

and

$$\begin{aligned} \frac{\mathrm{TP}}{S} + \frac{\mathrm{TN}}{S} + \frac{\mathrm{FP}}{S} + \frac{\mathrm{FN}}{S} &= \frac{S}{S} \\ \frac{\mathrm{TP}}{S} + \frac{\mathrm{TN}}{S} + \frac{\mathrm{FP}}{S} + \frac{\mathrm{FN}}{S} &= 1 \end{aligned} \tag{3.2}$$

That is, for the purpose of evaluating classification models, each confusion matrix $\left(\begin{smallmatrix} \mathrm{TP} & \mathrm{FN} \\ \mathrm{FP} & \mathrm{TN} \end{smallmatrix}\right) \in \mathcal{M}(2\times 2, \mathbb{N}_0)$ is transformed into $\left(\begin{smallmatrix} \frac{\mathrm{TP}}{S} & \frac{\mathrm{FN}}{S} \\ \frac{\mathrm{FP}}{S} & \frac{\mathrm{TN}}{S} \end{smallmatrix}\right) \in \mathcal{M}(2\times 2, \mathbb{Q}\cap[0,1])$, with $\frac{\mathrm{TP}}{S}+\frac{\mathrm{TN}}{S}+\frac{\mathrm{FP}}{S}+\frac{\mathrm{FN}}{S} = 1$.
The equation in Equation 3.2 defines the geometrical multidimensional space of all possible combinations of TP, TN, FP, FN that can result from a binary classifier algorithm.
Now, the goal is to develop a tool enabling the performance of the classifier consisting of more than two dimensions so to overcome the current bidimensional approach. Yet, the ultimate objective is to aid the intepretation (other than the evaluation of the model) and

four spatial dimensions cannot be visualized by human's vision system so it would not serve objectives set.
As a result, the tool is to be built on a three dimensional space resulting in the need to choose three out of the four variables in Equation 3.2.

Certainly, the geometrical environment under construction should include the first two elements of Equation 3.2 as TP and TN are the variables on which one is more interested in when classifying (*i.e.* I want the model to lead to true predictions). The third dimension is chosen among the last two elements of Equation 3.2. Here the choice goes for FP performing a design choice (but could alternatively go to FN depending on the specific application scenario).

Hence in Equation 3.2 three variables are assigned

$$\begin{aligned} \frac{\mathrm{TP}}{S} &= x \\ \frac{\mathrm{TN}}{S} &= y \\ \frac{\mathrm{FP}}{S} &= z \end{aligned} \tag{3.3}$$

obtaining

$$\begin{aligned} x + y + z + \frac{\mathrm{FN}}{S} &= \frac{S}{S} \\ x + y + z &= 1 - \frac{\mathrm{FN}}{S} \end{aligned} \tag{3.4}$$

With this variable assignment, the geometrical environment of all possible confusion matrices combination (*see* Equation 3.2) takes the form of a Cartesian equation of the plane (*see* Equation 3.4).

$$\begin{aligned} x + y + z &= k \\ with\ k &= 1 - \frac{FN}{S} \end{aligned} \tag{3.5}$$

This is the mathematical setting behind the environment under construction that leads to the obtaining of the plane. Now, given the specific application scenario of binary classification , there are assumptions and characteristics that needs to be taken into account and that eventually lead to the definition of the geometrical environment for the tetrahedron. In particular,

- $\mathrm{FN} \geq 0$ by definition,

- as a consequence, $\frac{\mathrm{FN}}{S} \geq 0$

,

- which means that $\frac{\text{FN}}{S} \geq 0$

  ,

- in practice as the classifier improves, the $\frac{\text{FN}}{S}$ tends to 0

  ,

- hence the k of the plane, which is the second term in Equation 3.4, and that is equal to $1 - \frac{\text{FN}}{S}$ tends to 1

- with the result that the geometric space is given by all those combinations of x, y, z for which

$$x + y + z \leq 1 \tag{3.6}$$

- considering that in the context of classification tasks the three variables x, y, z are always greater or equal to zero (*i.e* the model always obtains a prediction, being it either correct or wrong)

$$\begin{aligned} x &\geq 0 \\ y &\geq 0 \\ z &\geq 0 \end{aligned} \tag{3.7}$$

- intersecting this plane Equation 3.6 with these conditions one obtains the definitive geometrical environment for the confusion tetrahedron (Equation 3.10)

$$\begin{cases} x + y + z \leq 1 \\ \\ x \geq 0 \\ y \geq 0 \\ z \geq 0 \\ \\ \end{cases} \tag{3.8}$$

It can be formalized that

$$CT = \left\{(x, y, z) \in \mathbb{R}^3 \colon x \geq 0, y \geq 0, z \geq 0, 0 \leq x + y + z \leq 1\right\} \tag{3.9}$$

As a result, I obtain the geometrical space of all possible combinations of TP, TN, FP, FN that is the *Confusion Tetrahedron* (*see* Equation 3.8). To sum up, this process of transforming the multidimensional space into a threedimensional one consists in a projection $\mathcal{P}(\left(\begin{smallmatrix} \text{TP} & \text{FN} \\ \text{FP} & \text{TN} \end{smallmatrix}\right)) = (\frac{\text{TP}}{S}, \frac{\text{TN}}{S}, \frac{\text{FP}}{S})$ that univocally maps a confusion matrix to a rational point in CT.

Similarly, the preimage under $\mathcal{P}$ of a rational point $g = (\frac{a}{b}, \frac{c}{d}, \frac{e}{f}) \in \mathrm{CT}$ is the equivalence class of all integer multiples of the confusion matrix $\gamma = \begin{pmatrix} \frac{ka}{b} & 1-k\left(\frac{a}{b}+\frac{c}{d}+\frac{e}{f}\right) \\ \frac{ke}{f} & \frac{kc}{d} \end{pmatrix}$, that is $\mathcal{P}^{-1}(g) = \mathbb{N} \cdot \gamma$.

Thus, every confusion matrix $\left(\begin{smallmatrix} \mathrm{TP} & \mathrm{FN} \\ \mathrm{FP} & \mathrm{TN} \end{smallmatrix}\right)$ can be geometrically seen as a rational point in the space

$$CT = \left\{(x, y, z) \in \mathbb{R}^3 \colon x \geq 0, y \geq 0, z \geq 0, 0 \leq x + y + z \leq 1\right\} \tag{3.10}$$

where we arbitrarily set the positions $x = \frac{\mathrm{TP}}{S}$, $y = \frac{\mathrm{TN}}{S}$ and $z = \frac{\mathrm{FP}}{S}$, and thus $\mathrm{FN} = S \cdot (1 - x - y - z)$.

The space CT is a tetrahedron in the Euclidean space, with vertices $O = (0, 0, 0)$, $A = (1, 0, 0)$, $B = (0, 1, 0)$ and $C = (0, 0, 1)$, as displayed in Figure 3.1. Geometrically, CT

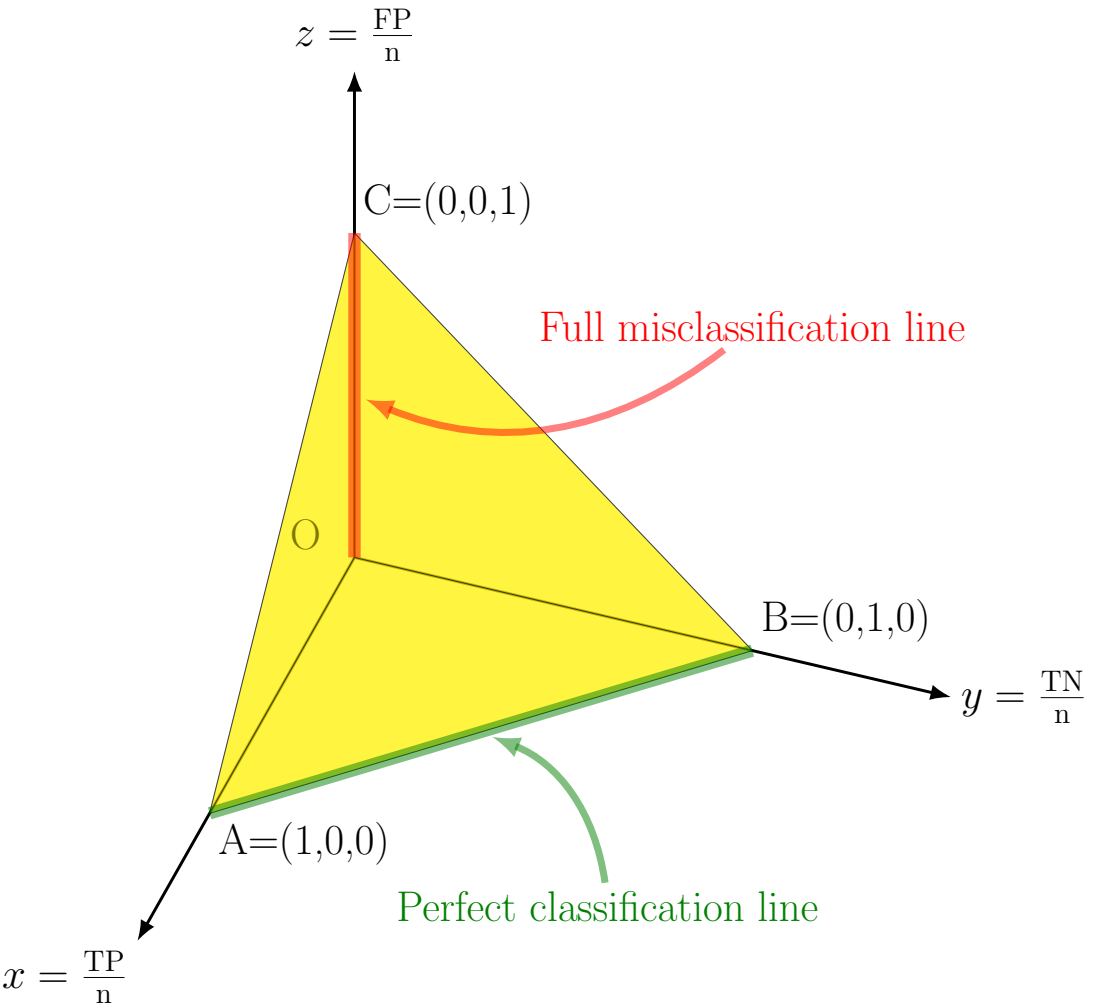


Figure 3.1: The Confusion Tetrahedron

is a 3-simplex whose lower dimensional simplices (in the notation of Figure 3.1) can be meaningfully interpreted in terms of families of particular or extreme confusion matrices.

0-simplices The 4 vertices of CT:

- O=(0,0,0) All S samples in $\mathcal{D}$ are positive, but the model misclassifies them all as false negatives $\left(\begin{smallmatrix} 0 & S \\ 0 & 0 \end{smallmatrix}\right)$;
- A=(1,0,0) All S samples in $\mathcal{D}$ are positive, and the model correctly classifies them $\left(\begin{smallmatrix} S & 0 \\ 0 & 0 \end{smallmatrix}\right)$;
- B=(0,1,0) All S samples in $\mathcal{D}$ are negative, and the model correctly classifies them $\left(\begin{smallmatrix} 0 & 0 \\ 0 & S \end{smallmatrix}\right)$;
- C=(0,0,1) All S samples in $\mathcal{D}$ are negative, but the model misclassifies them all as false positives $\left(\begin{smallmatrix} 0 & 0 \\ S & 0 \end{smallmatrix}\right)$;

1-simplices The 6 edges of CT:

- $\overline{\text{AO}}$ the segment $\{z = 0\} \cap \{y = 0\} \cap \{0 \leq x \leq 1\}$: all the S samples are positive, and C correctly classifies TP of them and misclassifies the remaining FN=S-TP;
- $\overline{\text{BO}}$ the segment $\{z = 0\} \cap \{x = 0\} \cap \{0 \leq y \leq 1\}$: TN samples are negative, correctly classified by C TN and S-TN samples are positive and misclassified by C as false negatives FN=S-TN;
- $\overline{\text{CO}}$ the segment $\{x = 0\} \cap \{y = 0\} \cap \{0 \leq z \leq 1\}$: D has FP negative samples and FN=S-FP positive samples, all misclassified by C; this is the segment of **full misclassification**;
- $\overline{\text{AB}}$ the segment $\{z = 0\} \cap \{x + y = 1\}$: D has TP positive samples and TN negative samples, all correctly classified by C; this is the segment of **perfect classification**;
- $\overline{\text{AC}}$ the segment $\{y = 0\} \cap \{x + z = 1\}$: D has TP positive samples all correctly classified by C and FP negative samples, all misclassified by C;
- $\overline{\text{BC}}$ $\overline{\text{AC}}$ the segment $\{y = 0\} \cap \{x + z = 1\}$: D has TP positive samples all correctly classified by C and FP negative samples, all misclassified by C;

2-simplices The 4 faces of CT:

- $\triangle$ ABC $\{x \geq 0\} \cap \{y \geq 0\} \cap \{z \geq y\} \cap \{x + y + z = 1\}$: all positive samples are correctly classified FN = 0;
- $\triangle$ ABO the triangle $\{z = 0\} \cap \{x \geq 0\} \cap \{y \geq 0\} \cap \{x + y \leq 1\}$: all negative samples are correctly classified FP = 0;
- $\triangle$ AOC the triangle $\{y = 0\} \cap \{x \geq 0\} \cap \{z \geq 0\} \cap \{x + z \leq 1\}$: all negative samples are misclassified TN = 0;
- $\triangle$ OBC the triangle $\{x = 0\} \cap \{y \geq 0\} \cap \{z \geq y\} \cap \{y + z \leq 1\}$: all positive samples are misclassified TP = 0;

It results that the Confusion Tetrahedron is a geometric view of all the possible equivalence classes of confusion matrices, normalized by the number of samples so that all entries lie in [0,1] and their sum is 1. The three axes represent TP, FP, TN.

When evaluating a classification model performance through its Confusion Matrix (CM), being the CT the geometrical view of all the possible confusion matrices, the CM associated with the model under evaluation will necessarily be one of the points composing the geometrical environment of the tetrahedron Equation 3.10. At this point, the evaluation is straightforward because, depending on the **position**, a researcher can visually grasp the goodness of the model.

In particular, the best classification occurs when false predictions are minimized and true predictions maximized. So in the environment $\frac{\text{FP}}{N} \to 0$ and $\frac{\text{TP}}{N} \to 1$ and $\frac{\text{TP}}{N} \to 1$ which is in $\overline{AB}$

$$\begin{aligned} \text{Perfect classification line } \overline{AB} = \big\{(x,y,z) \in \mathbb{R}^3 \colon x \geq 0, y \geq 0, z = 0, x + y = 1\big\} \\ \big\{(x,y,z) \in \mathbb{R}^3 \colon x + y = 1\big\} \end{aligned} \tag{3.11}$$

On the other hand, the worst classification occurs when true predictions are minimized and false predictions maximized. So in the environment $\frac{\text{FP}}{N} \to 1$, $\frac{\text{TP}}{N} \to 0$, $\frac{\text{TP}}{N} \to 0$ which results in

$$\begin{aligned} \text{Worst classification line } \overline{OC} = \big\{(x,y,z) \in \mathbb{R}^3 \colon x = 0, y = 0, z > 0, 0 \leq x + y + z \leq 1\big\} \\ \big\{(x,y,z) \in \mathbb{R}^3 \colon 0 \leq z \leq 1\big\} \end{aligned} \tag{3.12}$$

Additionally, given that the CT offers a view over all possible confusion matrices, it is possible to enrich the information provided by the visualisation through the coloring of each projected CM. The projection of the CM in the CT can be coloured according to one of the performance metrics (μ) illustrated in Section 2.3. Coloring the points of CT according to the value of a scaling invariant performance measure μ, it is possible to have a representation of the environment of the learning model leading to a global view of the behavior of the model according to the chosen metric. As a result, the tetrahedron provides valuable information not just through the *position* of the single model but also through the *coloring* offering a full 3D view of μ-colored CT.

3.3 The Confusion Tetrahedron: implementation

The working environment has been prepared by installing Python,PyCharm, R, Jupyter and the appropriate extensions to make it work with these programming languages. A

virtual environment [1] has been then created to install specific versions of the required libraries [2] independently from other packages already present so to avoid the clash of dependencies. The data has been collected from the UCI repository [3], cleaned and preprocessed for the classification. Then, the workflow has proceeded with the spot check of several classification algorithms. In detail, the tested algorithms are linear models such as Logistic Regression (LR) and Naive Bayes (NB), a non linear model through the K-Nearest Neighbors (KNN), Tree Based Methods with Decision Tree (DT) and their improvements through ensemble learning (Bagging, Boosting, Random Forests) and Support Vector Machines (SVM).
The steps illustrated so far represent just the preparatory phase for the core activity of the present work that is the evaluation of the fitted models both through the standard procedure (*i.e.* assessment of performance) as well as the novel approach of the CT. Hence, the following steps consisted of visualizing the tetrahedron according to several performance metrics and with different tools. Eventually, the work has focused on the exemplification of the advantages of such three-dimensional view through the four applications.
The rest of the paper deepens each of the anticipated steps of the workflow and explains the perks provided by the tetrahedron in all four applications scenarios.

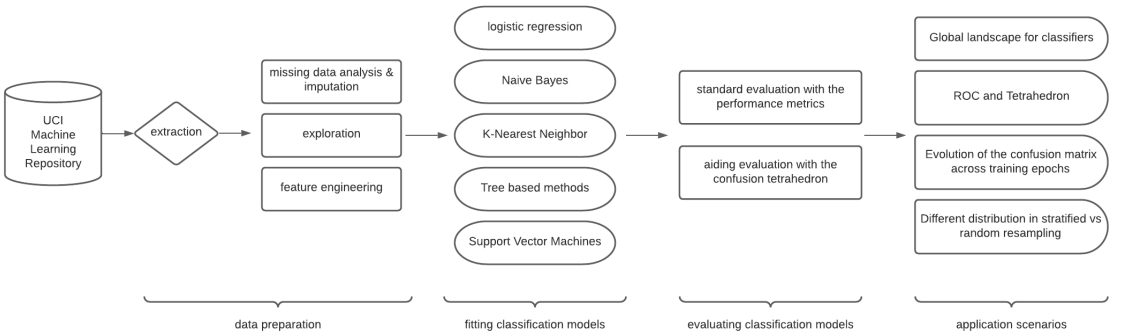


Figure 3.2: Implementation: workflow diagram

3.3.1 Dataset

The data was collected from the UCI Machine Learning Repository which offers databases, domain theories, and data generators to the machine learning community to allow them to empirically analyse the algorithms. The archive was created as an ftp archive in 1987 by David Aha and fellow graduate students at UC Irvine. Since that time, it has been widely used by students, educators, and researchers all over the world as a primary source of machine learning data sets. As an indication of the impact of the archive, it has been cited over 1000 times, making it one of the top 100 most cited resources in all of computer science.

[1] venv - creation of virtual environments
[2] list of items to be pip-installed can be found in the file requirements.txt
[3] the datasets used for the presented implementation is available at following link

The UCI repository represents the best source for the input data as it builds a solid foundation for the analysis pipeline avoiding the risk of unexpected anomalies and errors in the starting dataset that might negatively affect the last stages of the workflow without providing any added value (*i.e.* collection errors that propagates until the results of the fitted models). By taking advantage of the UCI repository, one can benefit from a faster preprocessing dealing with relatively trustworthy datasets while at the same time experiencing the perks of real-world datasets with realistic features and traits leading to a reliable picture of the performance of the models. Moreover, datasets from the UCI repository are typically well studied so one can expect good results and can more easily spot oddities or irregularities in the results effectively acting as control measures for one owns analysis. Lastly, a public repository as data source represents a wise choice because it means to have reproducible results offering a baseline analysis for the proposed visualization. Researchers who will want to understand more about the confusion tetrahedron will be able to re-execute my study and, where needed, tailor it to their needs.

The requirements for the dataset were to have a binary output (as the aim is to improve the evaluation of binary classification algorithms through a visualisation approach) and to be of a relatively small size (so to cut unnecessary time waste in the processing of the models which is not the focus of this work). Moreover, it was auspicable for the chosen input dataset to be about medical data because of its popularity in today's machine learning applications and because it is typically unbalanced hence better testing models' performances. In fact, the imbalance of the target class is a trait that machine learning practicioners will have to deal with when working with data, especially in more impacting fields such as that of medical diagnosis or customer churn detection.

After identifying a selection of datasets, one about chronic kidney diseases has been chosen, fulfilling all the requirements of medical field, binary output and imbalance in the output class distribution. At the time of the publication, the dataset used for the present analysis can be found in the UCI Machine Learning Repository via this link. If the provided connection does not lead to a valid resource, the data can still be retrieved from the Github repository where the code for this project is stored.
In detail, in its raw form the dataset consists of properly anonimised data of 400 patients analysed in 2015 thanks to the combined work of the Apollo Hospital and Alagappa University. As found in the repository, the studied features are 24 plus the target class which is the presence or absence of a chronic kidney disease. The attributes are described in detail providing also a brief medical context for the features. Such explanations are not extremely relevant for the purpose of this analysis but are useful to familiarize with the dataset and eventually have a better control over the results. More specifically, the attributes are:

- Age (*age*) - numerical

Age of the patient in years

- Blood Pressure (*bp*) - numerical
  Blood pressure of the patient recorded in millimeters of mercury (mm/Hg). Usually it is recorded with two values (systolic and diastolic blood pressure). Here only one number is provided and, given the range, I expect it to be the diastolic value (lower number).

- Specific gravity (*sg*) - nominal
  Specific gravity indicates the state of hydration of the patient with higher values indicating an alarming dehydration (ie. greater concentration of solutes). Here it is provided as a categorical variable with five possible values being: 1.005, 1.010, 1.015, 1.020, 1.025.

- Albumin (*al*) - nominal
  Albumin is a protein found in the blood that should not pass into urine. Anything above 30 mg/g may indicate the presence of a kidney disease. Here the recorded value is not the actual number but a categorical level with five possible options being: 0,1,2,3,4,5.

- Sugar (*su*) - nominal
  The presence of glucose in urine in an amount greater to 0.8 millimoles per liter (mmol/L) could be a sign of a health problem. Similarly to the albumin case, the actual numerical value is not provided, the information has been discretized in five possible levels being: 0,1,2,3,4,5.

- Red Blood Cells (*rbc*) - nominal
  The datasets treats this variable as categorical distinguishing between a "normal" or "abnormal" presence of the red blood cells.

- Pus Cells (*pc*) - nominal
  Pus cells are essentially an excess of white blood cells and their presence indicates an elevated amount of bacteria. Again, the actual numerical value is not provided and the variable is treated as categorical with two possible options being "normal" or "abnormal" presence of white blood cells.

- Pus Cells Clumps (*pcc*) - nominal
  Similarly to the previous variable, this feature indicates the presence of bacteria. What differs is that when clumps are formed it means that an infection has been lasted for a while indicating a more severe medical condition. The variable is treated as categorical recording the clumps as "present" or "notpresent".

- Bacteria (*ba*) - nominal
  Bacterial colonization represents a sign of infection. This variable is treated as categorical recording the bacteria as "present" or "notpresent".

- Blood Glucose Random (*bgr*) - numerical
  Together with the categorical variable (*sugar*), the level of sugar is recorded also in the blood providing the value in milligrams per deciliter (mgs/dl).

- Blood urea (*bu*) - numerical
  The actual numerical value of the amount of blood is recorded in milligrams per deciliter (mgs/dl).

- Serum creatinine (*sc*) - numerical
  Creatinine is a waste product resulting from the normal wear and tear of the muscles. Higher levels of creatinine may indicate that the kidneys are not well performing in filtering waste from the blood. The level of serum creatinine in the urine is recorded in milliequivalent per litre (mEq/L).

- Sodium (*sod*) - numerical
  A normal value of sodium falls between 40 mEq/L and 220 mEq/L, levels outside this range can be an indicator of kidney damages.

- Potassium (*pot*) - numerical
  Potassium plays a crucial role in cell metabolism, and it's important in maintaining the balance of fluids and electrolytes in the body. Having high or low levels of potassium can be an indicator for kidney damage. Here the level of potassium in urine is provided in milliequivalent per litre (mEq/L).

- Hemoglobin (*hemo*) - numerical
  A normal level of hemoglobin is between 135 and 175 grams and a tad smaller for women, ranging from 120 and 155. If the level of hemoglobin in the blood rises too high, then hemoglobin begins to appear in the urine indicating a kidney condition that requires attention.

- Packed Cell Volume (*pcv*) - numerical
  The Packed Cell Volume (or hematocrit) is a measurement of the proportion of blood that is made up of cells (percentage of cells in blood). A rise in PCV indicates a state of dehydration which is particularly harmful for the kidneys.

- White Blood Cell Count (*wc*) - numerical
  This numerical variable is another element of the complete blood count and indicates the value value of white cells in the blood recorded as cells per cubic millimeter (cells/cmm).

- Red Blood Cell Count (*rbcc*) - numerical
  Red blood cells carry oxygen throughout the body so having low or high levels of red blood cells can be an indicator for diseases. This variable is treated as numerical and recorded in millions per cubic millimeter (cells/cmm).

- Hypertension (*htn*) - nominal
  A blood pressure that is higher than normal can be a sign of chronic diseases. Here it is recorded as a categorical variable with two values indicating the presence (*yes*) or absence (*no*) of hypertension in the patient.

- Diabetes mellitus (*dm*) - nominal
  Diabetes mellitus is a disease that affects how the body uses blood sugar leading to high levels of sugar in the blood which is a major risk factor for kidney disease. Here it is treated as a categorical variable with two values indicating if the patient is affected (*yes*) or not (*no*) by this medical condition.

- Coronary Artery Disease (*cad*) - nominal
  Coronary artery disease consists in a condition in which the heart is unable to pumpt the blood in the right way. It is the leading cause of morbidity and mortality in patients with chronic kidney disease hence this information is provided in the dataset as a categorical variable indicating whether or not the patient suffers from this illness.

- Appetite (*appet*) - nominal
  A poor appetite is a common condition of patients with several health diseases and particularly relates to the pedal edema and hypertension. Here it is recorded as a categorical variable quantifying the appetite of the patient with only two levels being these "good" or "poor".

- Pedal edema (*pe*) - nominal
  Pedal edema consists in the accumulation of fluid in the lower legs. This is related to declining levels of albumin and a retention of sodium that causes swelling in the feet. In the dataset it is treated as a categorical variable with two values indicating the fact that the patient does present ("yes") or does not present ("no") edema to the feet.

- Anemia (*ane*) - nominal
  Anemia is a condition in which a lower level of red blood cells causes the inadequate transportation of oxygen to the tissues. Here it is provided as a categorical variable with only two values indicating the fact that the patient is or is not affected by this condition.

- Class (*class*) - nominal
  This is the binary target class indicating the presence (*'ckd'*) or absence of the disease (*'notckd'*)

To see the attributes in a more concise form head to Table A.1 in the Appendix.
Additional information about the dataset are that there are missing values and that it is imbalanced. In fact, there is a different number of instances for the presence (*ckd*: 250 instances) and absence (*notckd*: 150) of the chronic kidney disease.

Operatively, the data has been downloaded as a rar file and decompressed obtaining three different files: the description of the dataset (*chronic_kidney_disease.info.txt*), a partial set of records (*chronic_kidney_disease.arff*) and the whole dataset (*chronic_kidney_disease_full.arff*). The full dataset has been chosen as input data for the analysis being the one with the greatest amount of information. The dataset is an ARFF(Attribute-Relation File Format) file so an ASCII text file that describes a list of instances sharing a set of attributes. This extension is compatible with the Weka[4] machine learning software which is an open source software for machine learning. In order to work on the dataset with Pandas the dataset has been converted into a csv file with a python script[5] and loaded into a jupyter notebook and PyCharm.

3.3.2 Data exploration and preparation

The starting data is raw data without any kind of manipulation, exception made for anonimisation procedures that lead to have only the features and the target class without any reference to the patients. Being the dataset loaded in its raw form, there is the need to preprocess it to bring it into a proper format understandable by the algorithms. To this end, the data preparation phase has proceeded with the correction of several errors, the analysis and imputation of missing data, the exploration of the whole dataset and the engineering of the features with the aim of feeding the dataset to binary classification algorithms. The reasoning behind each of these steps is explained in depth in the following paragraph before moving on to the next section which is the fitting of the models.

Cleaning obvious errors

The first step after loading the dataset was to have a feeling of the content and prepare it to feed them to the classification models. The process has started by removing the quotes from the headers of the variables which would have prevented from using certain syntax. Moreover, obvious errors has been cleaned such as an unexpected number of classes (having it chosen with the binary classification in mind, the first thing one could notice is that there were 3 classes in the beginning). The same issue has occurred with other categorical variables for which the number of levels was known to begin with. This was due to the fact that missing values have been denoted with '?' leading the syntax to consider it as an additional level of the variable (further evaluations in subsection 3.3.2).

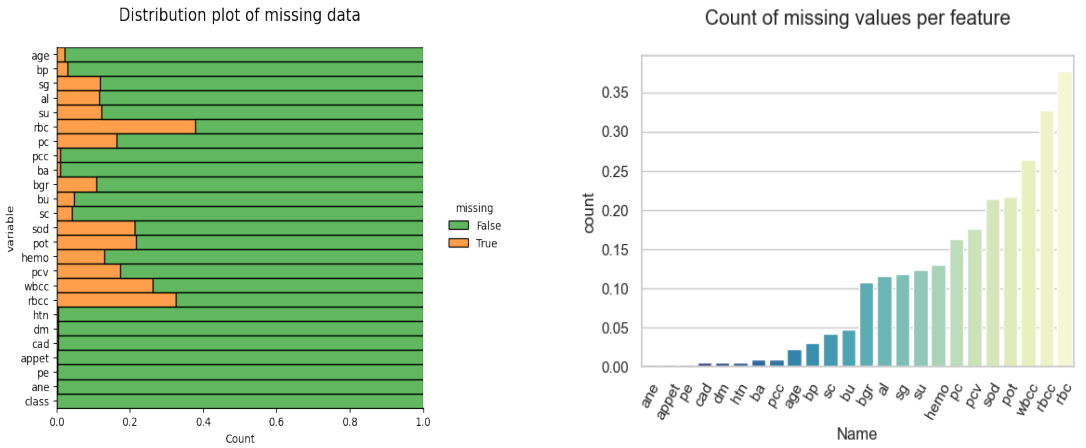
Missing data: analysis and imputation

The real-world data often has a lot of missing values being this generally due to either corruption or failure to record data. The handling of missing data is very important during the preprocessing of the dataset as many machine learning algorithms do not support missing values so in order to best deal with such issue, first the distribution of missing

[4] `https://www.cs.waikato.ac.nz/~ml/`

[5] `https://github.com/haloboy777/arfftocsv`

values has been analyzed and then it has been proceeded with their imputation.
As for the analysis of the missing values, this stage has required to substitute the question mark notation for the missing value with actual numpy NaN so to separate the study of numerical and categorical variables and work more seamlessly with *isnull()* function. After doing so, the distribution plot of missing values for each variable (*see* Figure 3.3a) have been plotted from which one is able to appreciate that there are attributes with a small presence of missing values, such as anemia (*ane*) or pedal edema (*pe*). On the other hand, there are also attributes with a relatively considerable number of missing values such as the various types of blood cells counts (*rbc*,*rbcc*,*wbcc*), level of potassium (*pot*), or sodium (*sod*). Hence, to have a more accurate view, the percentage of missing variables for each feature (*see* Figure 3.3b) has been plotted. The greatest presence of missing values is in the Red Blood Cell Count (*rbc*) variable where more than 35% of the records do not provide this value, followed by other types of blood cells counts. The reasoning behind these missing values is not likely to be due to a corruption in the data but is probably due to the fact that not all patients have had their blood tested. To have a more thorough understanding, the



(a) Distribution plot of the missing values (b) Barplot of the missing values per attribute

Figure 3.3: Analysis of the missing values

heatmap of missing values has been plotted and can be seen in Figure 3.4. Here one can see how the values that are not provided are all attributes that can be sourced back to a blood test results. Under the assumption that data has been collected chronologically, one can hypothetize that the creator of the dataset has considered to add blood samples during the development of the study. Then, to have a better understanding of the differences between the two levels of the binary class, the countplot of missing values distinguished per target class has been plotted (*see* Figure 3.4. Overall, there are generally fewer missing values for those without the chronic disease. For healthy patients (*orange*), less than 10% of the values are missing for each attribute but there are missing values for all variables. Instead, among patients with chronic kidney disease (*blue*), the missing values are less spreaded, less present in all the attributes but when present the number is generally higher (eg. *rbc,*

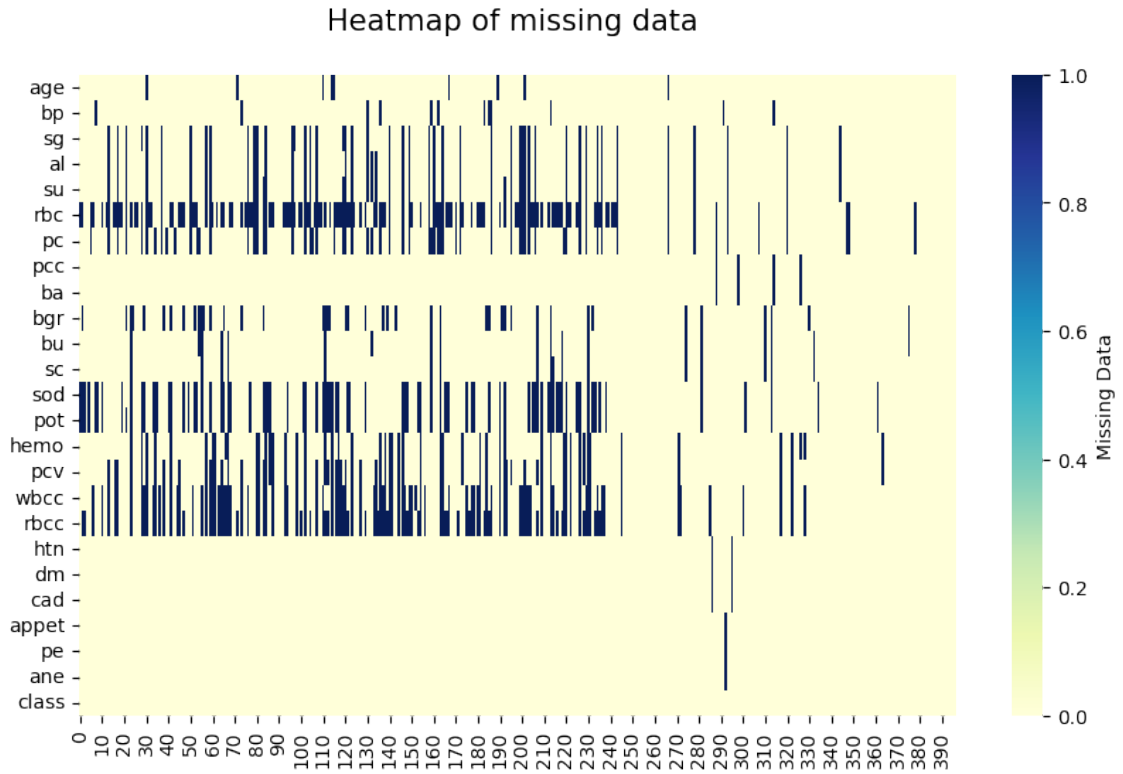


Figure 3.4: Heatmap of the missing values per attribute

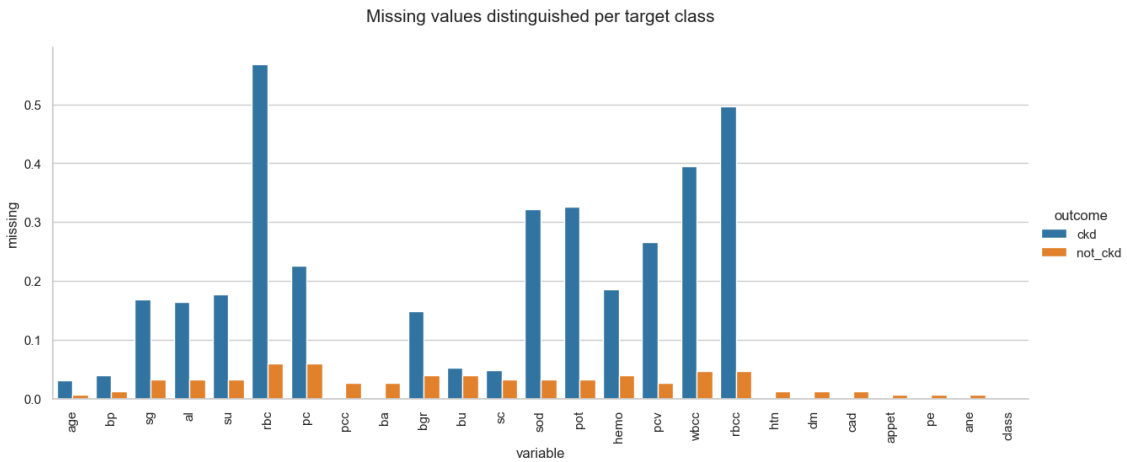


Figure 3.5: Missing values per attribute distinguished per class

rbcc, wbcc, pot, sod). On the other hand, still among those affected by the disease, there are variables that are completely free from missing values (those on the right eg.*cad, pe, ane, etc*). Lastly, one important aspect to notice is that there are no missing values in the target class, this is provided for all records.

After getting a sense of the missing value, the adopted strategy has been to delete elements for which the number of missing values is excessive. In this case, there were no features for which more than half of the records have missing values so all features have been kept. If this analysis were to be applied to another dataset, it is advisable to keep this issue into consideration. Instead, there were a few records for which the values of more than 10 features (among the 24) were missing. Such records have been dropped, which appears to be a reasonable course of action given that this meant to eliminate a negligible (only 11) number of records.
To deal with the imputation step in the resulting dataset, the strategy has been that of distinguishing among numerical and categorical variables and then proceed with the attribution. Because of the question mark notation of the missing values, Pandas loaded everything as object so the distinction of the type for the features has required some refinement. After doing this, the attribution of the values has followed. There is no rule of thumb to handle missing data that guarantees a robust model with good performance. There are several sophisticated techniques for the imputation of missing values to prevent the loss of information while at the same time not bias the analysis and hence create a robust model (*e.g.* imputation using Deep Learning Library Datawig [6] or the imputation of missing values with scikitlearn [7] which takes into account also the covariance between the missing value column and other columns). However, since the purpose of this work is more on the evaluation stage (than the processing), here the imputation is performed in a more straightforward way by substituting missing values with:

- the mean or the median depending on the distribution in the case of *numerical* variables;
- the mode in the case of *categorical* variables.

As for numerical variables, in order to decide whether the mean or the median were to be attributed, the boxplot for each variable has been plotted (*see* Figure 3.6). Here one can see how there are several outliers but all are reasonable meaning that there are no obvious errors (*e.g.* negative ages) so everything is kept as it is so to reproduce real world qualities as much as possible. Overall, most features are relatively skewed so the missing values get substituted with the median. On the other hand, for more homogenous distributions (*i.e. age, rbcc, pcv, hemo*) missing values have been substituted with the mean.

[6] https://datawig.readthedocs.io/en/latest/
[7] https://scikit-learn.org/stable/modules/impute.html

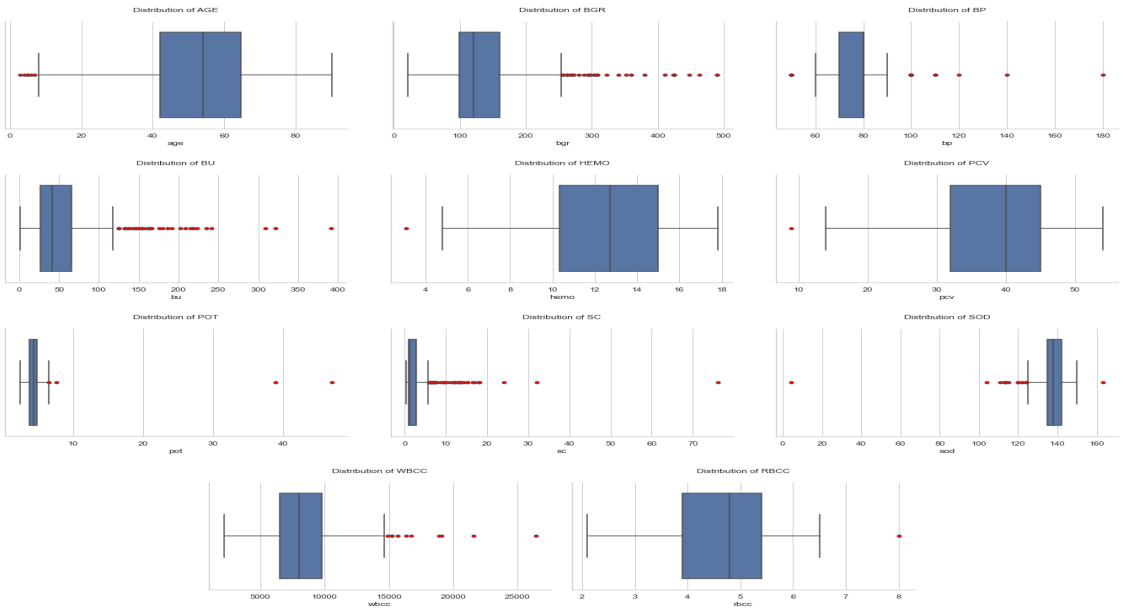


Figure 3.6: Boxplots of the numerical features

As for categorical variables, the procedure has been more effortless and simply attributed the most frequent level to the missing value.

Exploration of the dataset

Once obtained the complete dataset, two main visualisations have been plotted to have a better understanding of the dataset characterization and to suggest intuitions about the relationship between the features and the target class.
As for the numerical features, the boxplot for all variables has been plotted making a distinction between the two different levels of the target class (*see* Figure 3.7). The boxplots show how the values of the numerical features vary depending on the class (presence or absence of the kidney disease). If the plots are similar regardless the class then it is expected for the feeature not to influence greatly on the prediction of the class. Instead, if there is a contrast among the distribution for the two labels, than that feature will play a greater role. For example, Blood Glucose Random (*bgr*) has a staggering difference between patients affected by the chronic kidney disease and the healthy ones. For those without the disease the blood glucose random tends to be around the same value, whereas for those with the disease there is a greater range of values). Hence, one should expect these features to be more relevant in the prediction of the class. This is true also for other features such as *age*, *rbcc* and all those with a different distribution.

A similar exploration is performed also for categorical variables with a grouped bar chart that shows how each categorical value weigh in determining the class (*see* Figure 3.8). Similarly to what has been done for the numerical features, if there are distinctly distributions of target values between the classes it is likely that the variable will contribute more in the

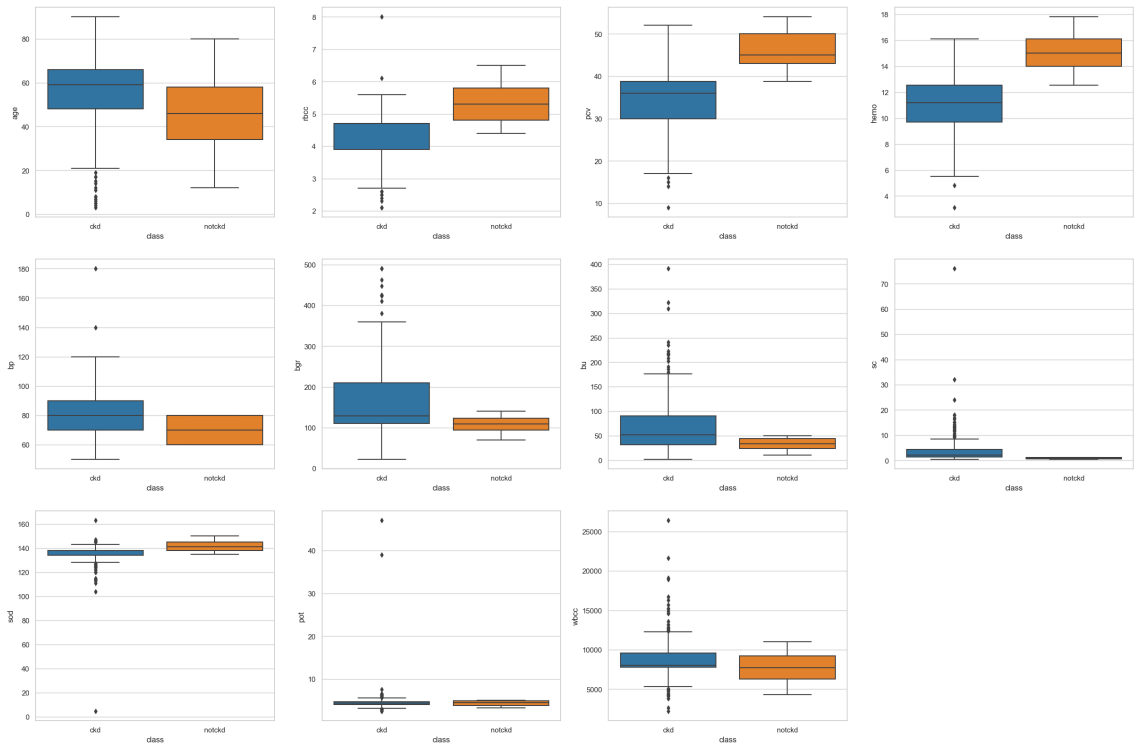


Figure 3.7: Boxplot of all numerical features vs target class

prediction of the label. For instance, this is the case of the presence of hypertension (*htn*) among patients which target value distinctly distributes between classes hinting at a greater role of this variable in the prediction. Similarly, there is a different distribution of the disease according to the presence or absence of Coronary Artery Disease (*cad*) since no patients with the coronary disease are free from the chronic kidney disease. Overall, all the displayed categorical features appears to have different distribution of the class according to the categorical value so one can reasonably expect all categorical features to be relevant in the prediction of the class.

Data preparation

To bring the dataset into a form that is properly readable by the classification algorithms the following steps have been undertaken:

- **Normalization**
  When features have different ranges, data normalization is necessary for machine learning to bring the values to a common scale, without distorting differences due to differences in the ranges of values. Otherwise, a feature with a larger value might influence the results making it artificially appear as more important than other predictors. In this dataset, one can notice (*see* Figure 3.9) that there is a varying scale so numerical variables are normalized.

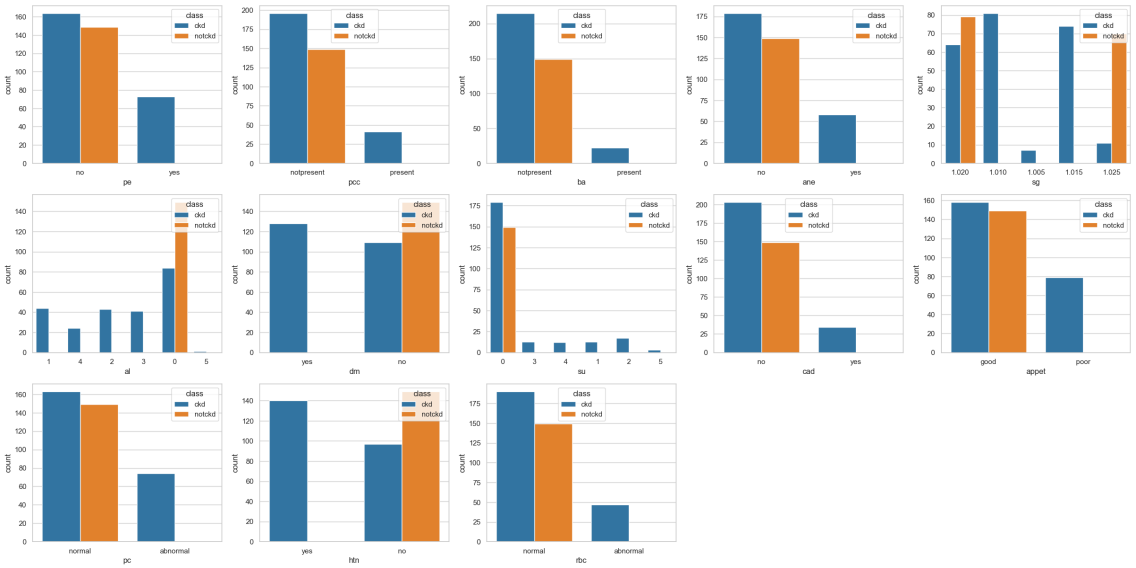


Figure 3.8: Grouped bar chart of categorical vs target variable

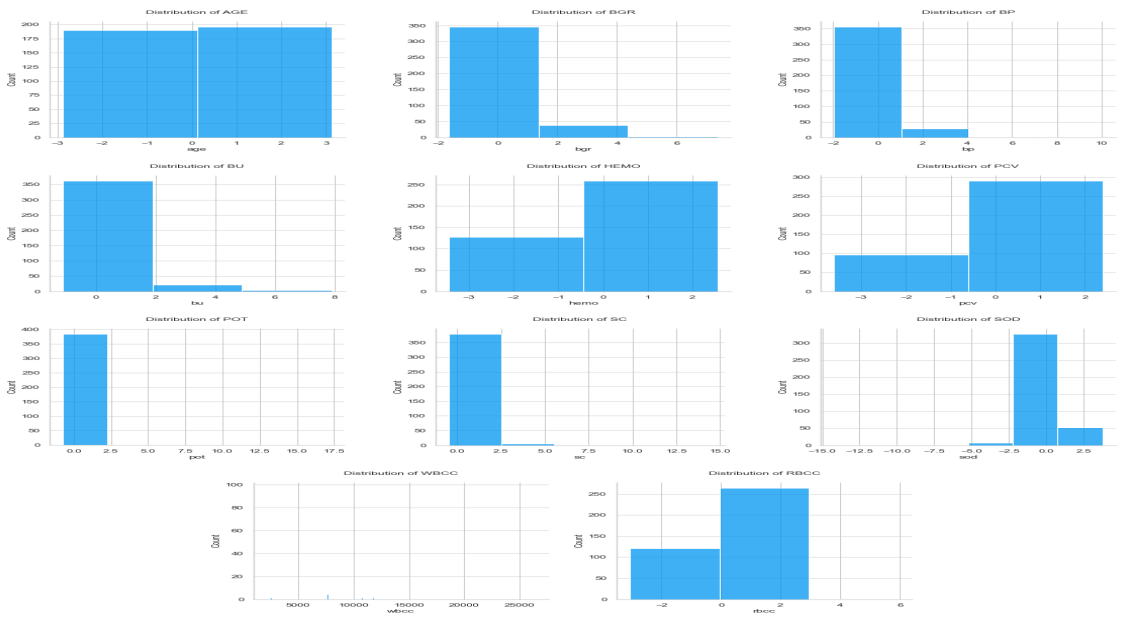


Figure 3.9: Need for normalization of the numerical variables

- **Encoding**

  To enable the learning algorithms to process the data using mathematical equations, the levels of the categorical variables need to be encoded. Here it is possible to proceed with the one-hot encoding offered by *Pandasget_dummies* [8] (no particular need to save the exploded categories with *OneHotEncoder* [9] from sklearn preprocessing given that all data has been processed already and that the number of categories is the same between the train and test set).

- **Feature modification**

  This specific dataset comes with a manageable number of features so no need to remove (*featureselection*) or combine (*featureengineering*) any of them. However, if such analyses were to be reproduced on other datasets, I would advise to monitor also the risk of high-dimensionality.

  In this case I consider to remove features only if there is multicollinearity (even though it should not severely affect the results in machine learning applications [10]). To quantify the severity of multicollinearity, the Variance Inflation Factor (VIF) is computed. This measures the ratio between the variance for a given regression coefficient with only that variable in the model versus the variance for a given regression coefficient with all variables in the model. All VIF values are around 1 meaning that the predictor is not correlated with the others. Higher and hence more correlated values are the *pcv* and *hemo* due to the fact that they depend on other features included in the blood test analysis. These are still kept because still the value is relatively low so there will be no associated issues.

- **Merging and balance check**

  The final dataset has been created by merging the numerical and categorical processed data and the encoded target class with 1 associated to the presence of the chronic kidney disease (*ckd*) and 0 to its absence (*notckd*). Since a relevant perk of the proposed tetrahedron, is the fact that it helps in better depicting the performance with imbalanced dataset. the distribution of the class is plotted to check that the dataset presents a different proportion of target labels (*see* Figure 3.10).

- **Splitting of the dataset**

  In order to have a portion of the data that can be used for the evaluation of the model, the final dataset is split into training and testing subsets. By doing so, a chosen portion of the data is not incorporated in the learning of the model minimizing the potential bias in the evaluation process. Using the dedicated method in scikitlearn [11] the cleaned dataset has been split setting the size of the test subset at 0.33 (*i.e.* 33%

[8] `https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html`

[9] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

[10] Stack exchange discussion

[11] link to the documentation, random_state=42

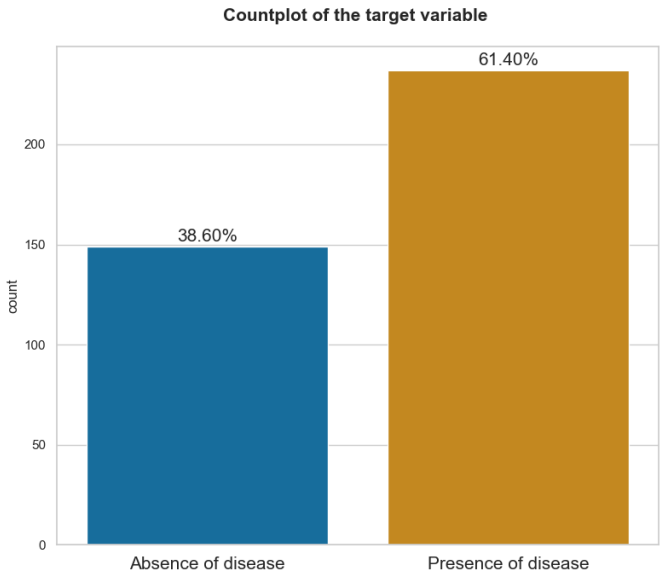


Figure 3.10: Countplot of the target variable in the final dataset

of the starting dataset is notused for the fitting of the model) which appears to be reasonable considering the size of the dataset.

3.3.3 Fitting the models and standard evaluation

Once the cleaned dataset has been obtained, several different algorithms have been tested so to obtain a consistent number of models on which to perform the evaluation procedure illustrating the difference between the standard and the novel approach. To have a more broad view of the possible results, the adopted algorithms range from linear and non linear methods to tree based ones. In particular, the algorithms included in the analysis and their associated results are:

- **Logistic regression (LR)**
  LR is the oldest and most common algorithm to solve a classification problem. It consists of a linear statistical method that uses a logistic (*logit*) function to model a categorical binary dependent variable predicting $P(Y = 1)$ as a function of the attributes X. The confusion matrix resulting from the LR model is reported in Table 3.1 its baseline version and in Figure 3.11 in the classical visualisation approach.

Table 3.1: LR - confusion matrix

		Predicted	
		Yes	No
Actual	Yes	48	1
	No	0	79

The standard evaluation approach for the binary confusion matrix establishes that

a number of performance metrics (*see* Chapter 2) is computed taking as input the matrix entries. Here, all models have been judged on a subset of these metrics consisting of *accuracy*, *precision*, *recall*, F_1 and MCC. The resulting indicators for LR can be seen in Table 3.2.

Table 3.2: LR - performance metrics

performance metric	**value**
accuracy	0.9922
precision	0.9875
recall	1.0000
F_1	0.9937
MCC	0.9835

To evaluate such results the only graphical effort is usually the plotting of the heatmap reporting a subset of the performance metrics (*see* Figure 3.11) which adds only a relatively small value for the interpretation of the results.

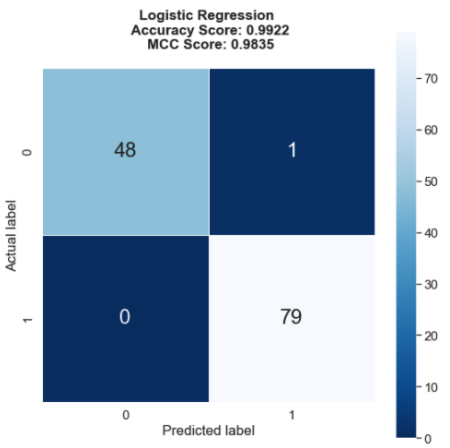


Figure 3.11: LR - classical visualisation approach

- **Naive Bayes (NB)**
  The NB method is another supervised learning algorithm based on applying the Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. The confusion matrix resulting from the NB model is reported in Table 3.3. The evaluation procedure adopted for this and the subsequent models follows the same procedure of the logistic regression case. Hence, the resulting indicators for NB can be seen in Table 3.4 and the visualisation of the results in Figure 3.12

Table 3.3: NB - confusion matrix

		Predicted	
		Yes	No
Actual	Yes	49	0
	No	2	77

Table 3.4: NB - performance metrics

performance metric	**value**
accuracy	0.9844
precision	1.0000
recall	0.9747
F_1	0.9872
MCC	0.9677

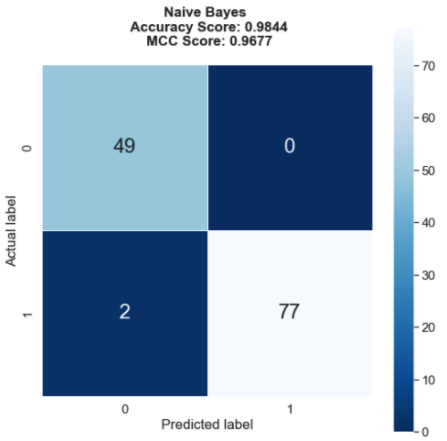


Figure 3.12: NB - classical visualisation approach

- **Support Vector Machines (SVM)**
  The SVM classifier adopts an iterative approach to find the Maximum Marginal Hyperplane (MMH) in the multidimensional space created by the features so to separate data belonging to different classes. The confusion matrix resulting from the SVM model is reported in Table 3.5, the resulting indicators for SVM can be seen in Table 3.6 and the visualisation of the results in Figure 3.13

Table 3.5: SVM - confusion matrix

		Predicted	
		Yes	No
Actual	Yes	47	2
	No	0	79

Table 3.6: SVM - performance metrics

performance metric	**value**
accuracy	0.9844
precision	0.9753
recall	1.0000
F_1	0.9875
MCC	0.9672

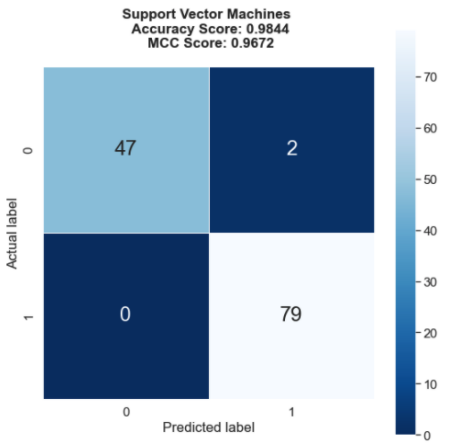


Figure 3.13: SVM - classical visualisation approach

- **K-Nearest Neighbor (KNN)**
  The KNN is a non parametric learning method that is based on the assumption that similar things exist in close proximity. Hence, it produces a prediction for the record assigning it to the class that is most common among its k nearest neighbors.
  In the case of this dataset, the number of K has been set by evaluating the testing accuracy resulting from different values of the K (*see* Figure 3.14) After setting $K = 3$,

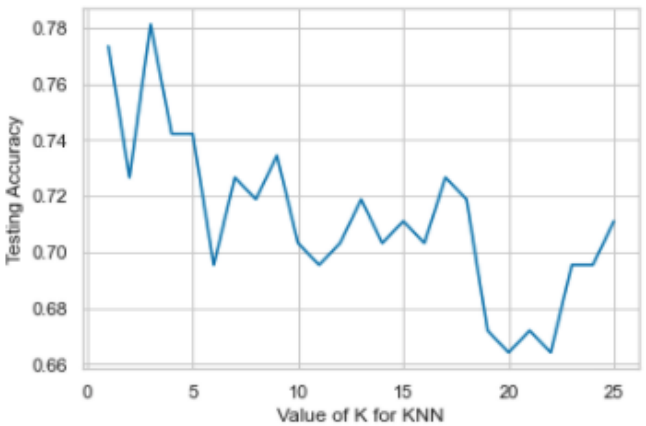


Figure 3.14: Choice of the K for KNN

the confusion matrix obtained is reported in Table 3.7. The resulting indicators for KNN can be seen in Table 3.8 and the visualisation of the results in Figure 3.15.

Table 3.7: KNN - confusion matrix

		Predicted	
		Yes	No
Actual	Yes	41	8
	No	20	59

Table 3.8: KNN - performance metrics

performance metric	value
accuracy	0.7812
precision	0.8806
recall	0.7468
F_1	0.8082
MCC	0.5679

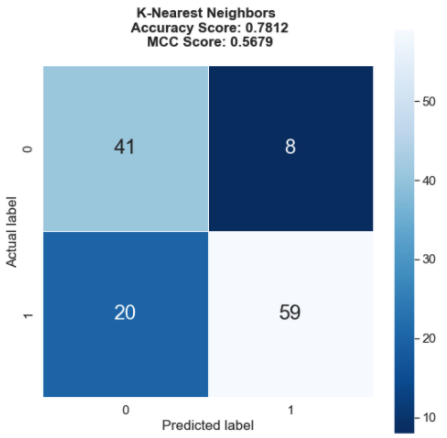


Figure 3.15: KNN - classical visualisation approach

- **Decision Trees (DT)**
  The DT algorithm continuously splits the data trying to define a condition on the features so to separate all the classes contained in the dataset to the fullest purity. In the implementation the Gini index has been used to measure of purity of a split (default for sklearn DecisionTreeClassifier()). The confusion matrix resulting from the DT model is reported in Table 3.9, the resulting indicators for DT can be seen in Table 3.10 and the visualisation of the results in Figure 3.16.

Table 3.9: DT - confusion matrix

		Predicted	
		Yes	No
Actual	Yes	47	2
	No	3	76

Table 3.10: DT - performance metrics

performance metric	value
accuracy	0.9609
precision	0.9744
recall	0.9620
F_1	0.9682
MCC	0.9178

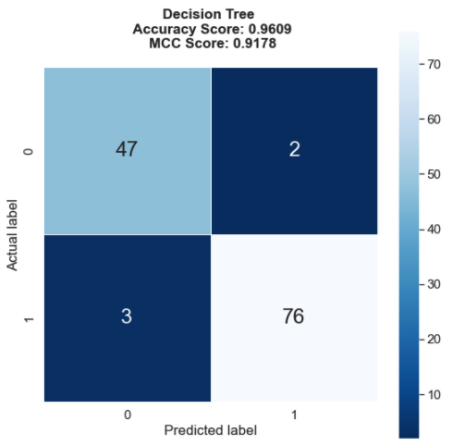


Figure 3.16: DT - classical visualisation approach

Several ensemble learnings methods (*i.e.* Bagging, Boosting and Random Forest (RF)) have been applied for the binary classification task but, given that the single DT already provided a relatively satisfactory performance, the outcomes are not extremely relevant and hence are not reported in the results

Table 3.11: Summary of the performance metrics

model	**accuracy**	**precision**	**recall**	F_1	MCC
LR	0.9922	0.9875	1	0.9937	0.9835
NB	0.9844	1	0.9747	0.9872	0.9677
SVM	0.9844	0.9753	1	0.9875	0.9672
KNN	0.7812	0.8806	0.7468	0.8082	0.5679
DT	0.9531	0.9740	0.9494	0.9615	0.9020

Overall, the outcomes (*see* Table 3.11) indicate that the models that appear to perform better are the linear ones and in particular the LR. Tree based methods do not particularly lend themselves to this specific use case as exemplified by the lower values relatively to the tested models. The MCC is particularly severe in the judgement of the DT allowing to spot the overfitting issue that characterizes this type of models. Instead, the KNN shows particularly substandard results, with an MCC performing just slightly better than the random case.

Such evaluations stem from the comparison of the performance metrics computed for each model. The results for this implementation are relatively straightforward so interpreting the results through a standard contingency table does not result as challenging as in more controversial cases.

Nonetheless, even if the interpretation is feasible, there is still the drawback of the information loss and the absence of a global view over the performance. In fact, the results are flattened to few numerical indicators without any visualisation over the geometrical environment in which the model learns or the global behaviour of the metric.

This is the dimension where the Confusion Tetrahedron demonstrates its full added value and the reason behind the proposal of a new approach to the evaluation of the results.

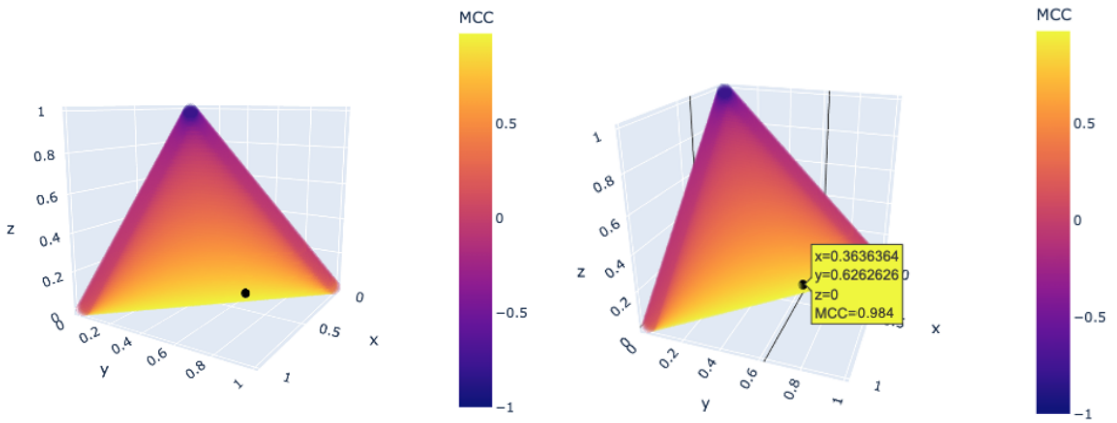
3.3.4 New approach to the evaluation

The novel proposal to the models' evaluation procedure integrates the assessment of the numerical indicators with the Confusion Tetrahedron. As anticipated, this visual tool provides a full 3D outlook on the behavior of the performance metric providing an global view over the results of the binary classification model enriched with information previously not observable.
The gradient of the tetrahedron results from the computation of the chosen performance metric for all possible equivalence classes of confusion matrices normalized by the cardinality of the dataset. In this implementation the MCC has been set as the default performance metric due to its robustness to imbalanced dataset and its greater reliability (see Chapter 2). However, it is possible to tailor the geometrical view substituting the MCC with the preferred performance indicator as explained in Section 4.1.

Here in the tetrahedron only the result for the logistic regression are reported, being this the model that appears as the preferable one from a first assessment with the numerical indicators subsection 3.3.3. Visualising the model under evaluation in the global environment of the binary classifier requires first to plot the tetrahedron and then add an additional visual element that takes as input the tested model. This is done in Python adopting Plotly [12] as graphing library because of its great interactivity, extended flexibility and lean aesthetic.
The result is a graphical tool that presents a full snapshot of the global behaviour of the perfomance metric and depicts how the single model under evaluation stands with respect to all possible outcomes. In the specific implementation displayed in Figure 3.17) the LR model (*i.e.* the black dot) is further from the violet regions representative of a lower MCC and stands in a yellow area which indicates a higher MCC. The symbol mark for the model's performance is extremely close to the perfect classification line which is the best outcome for any binary classification task. This placement visually communicates the great performance achieved by the model and hence is immediately intuitable by a broad range of stakeholders.

In the interactive visualization it is possible to observe simultaneously more than one model so to appreciate also the different positioning of one with respect to the other unlocking the possibility to formulate even more comparative insights.
To effectively explain how a different model would look, in Figure 3.18 the case of a particularly poor result is plotted together with the LR model to exemplify the different position of two opposite performances.
In Figure 3.18a the visual result of the poor performance stems from a fictitious confusion

[12] Plotly documentation



(a) Frontal view (b) User interaction

Figure 3.17: Novel approach to the evaluation of the tested model with the Confusion Tetrahedron. The three axes represent TP, FP, TN and the visualisation can be moved according to the preffered perspective

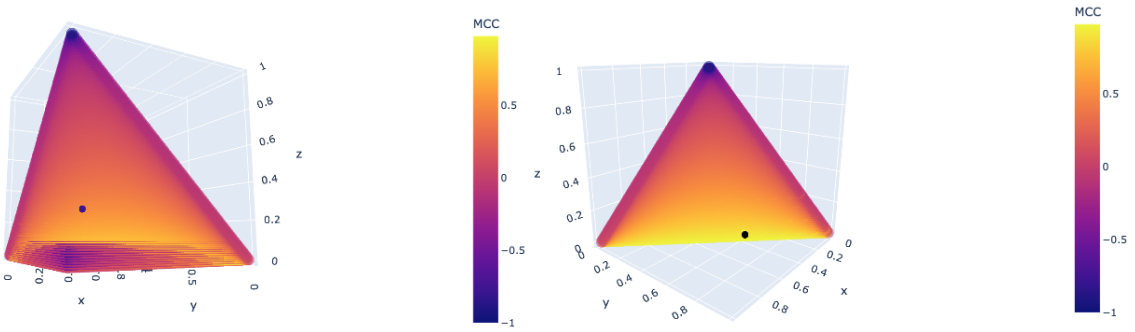
matrix formulated in such a way that it would lead to low MCC results (*see* Table 3.12).

Table 3.12: CT: Fictitious confusion matrix to show result of poor performance

		Predicted	
		Yes	No
Actual	Yes	10	30
	No	55	5

Hence, from Figure 3.18 it is possible to appreciate the relative positioning of different models within the global landscape of classifiers. The closer to the yellow area the better the performance of the model, whereas the more violet the region in which the dot stands, the poorer the result. Clearly, the extension of the areas associated to well and poor performance will differ depending on the performance metric adopted (*see* Section 4.1). However, regardless the chosen metric, it will always be true that the more the model symbol is closer to the diagonal of the x and y plane the better the performance. Similarly, the more this symbol shifts towards the z axis the inferior the performance.

The conclusions for this specific use case are relatively straightforward. However, the results already exemplify the added value provided by the Confusion Tetrahedron. Visualizing the model within the geometrical environment of all possible binary confusion matrices permits to more intuitively understand the results through the assessment of the model's placement within the environment. As a result, a researcher can evaluate the performance at a glance experiencing a more easier understanding of the information that is available with the current evaluation approaches; Moreover, the tetrahedron offers the possibility to view information that would not be observable without the tetrahedral structure such as where the model stands compared to all other possible outcomes. Hence, not only the



(a) Poor performance - fictitious CM (b) Good performance - LR CM

Figure 3.18: CT - comparison of models' performances

Indeed, the best approach to the evaluation of binary classification tasks would be the combination of the two approaches in the standard workflow. By putting the tetrahedron at the service of the computational approach, the interpretation and understanding of the numerical indicators would be facilitated. However, the benefit of an integration of the CT in common evaluation practices, would allow not only a more intuitive interpretation, but more importantly would provide additional information over the global learning environment that would not be possible to observe in the bidimensional space.

Notice, that this is the standard baseline visualisation of the Confusion Tetrahedron and it is consistent in the first application. Then, as we enter additional temporal dimensions, as in Section 4.3, the look is re-adapted to accommodate for a leaner visualization of the results maintaining the tetrahedron as the learning environment but coloring the dot to achieve a more leaner aesthetic Section 4.3. Overall, the design is subject to modification depending on the specific application scenario. The visualization portrayed here serve as a Proof of Concept (POC) demonstrating that the proposed design concept is feasible and provides additional insights with respect to current evaluation approaches

Further evolutions of this project aim to provide for more interactivity and personalisation. Current actions are oriented towards the:

- implementation of a Python package ready to be pip-installed in one own's environment;
- deployment of an interactive web application allowing researchers to tailor the results according to the selection of a set of widgets.

3.3.5 Technical specifications

The visualization software for the Confusion tetrahedron has been developed in the Python language using several libraries and modules that are detailed here along with their specific purpose within the analysis.

Among the totality of the libraries, Numpy and Pandas have been used to operate, process and generate the input data for the visualization. The input data for the different implementations has taken the form of a Pandas dataframe comprising of the three CT coordinates (*i.e.* the three normalized matrix entries Normalized True Positives (nTP), Normalized True Negatives (nTN), Normalized False Positives (nFP)). These three columns constitutes the baseline structure for the input data then, depending on the specific application, additional columns are added to this core format.
In particular, to assess the global behaviour of performance metrics (Section 4.1) a column reporting the computation of the metric of interest (*i.e.* MCC, accuracy, F_1 score) In the second application, given that the objective is to compare 2D and 3D visualization of the performance as the threshold varies, the coordinates for each of these geometrical environment have been generated. Hence, the baseline structure of the input data has been integrated first with the column vector of all the possible values for the threshold generated with numpy arange discretizing the value of the probability at 0.001. Then, for each level of the threshold, the associated coordinates for the 2D plane of sensitivity and specificity have been calculated and added to the input data.
As for the third application, the baseline input data structure with the CT coordinates has been generated extracting the confusion matrices of each of the 100 epochs with a callback function during the building process of the neural network. To link each confusion matrix to the associated epoch and eventually allow the slider animation according to the epoch, a column reporting the epoch number has been added to the baseline structure.
The fourth application follows a similar reasoning for the generation of the input data. For each split of the resampling the code keeps track of the confusion matrix obtained on the fold, creates a dataframe with the CT coordinates as well as the consequent normalized performance metric (*i.e.* MCC). This procedure is undertaken both for the stratified and non stratified resampling so the input data structure with the performance integrates also a column with a categorical variable indicating whether the set of coordinates stems from a split of randomized or stratified resampling. Having the type of resampling strategy encoded in a column of the input data allows the interactive personalization of the information displayed with Plotly.

Such input dataframes are then visualized with Plotly, an Open Source Graphing Library offering great interactivity, extended flexibility and a lean aesthetic. Within this library the most adopted modules have been Plotly express and Graph objects. Overall, the visualisation workflow consisted in plotting the geometrical environment of the CT with a 3D Mesh which, provided the vertices, draws a 3D set of triangles. Then, depending on the informatin to be delivered in the specific application either a line plot or scatterplot has been added within this structure.
In particular, for the visualization of the global behaviour of the performance metric, the volume delimited by the 3D Mesh of the CT has been discretized by initializing a numpy

linspace[13] from 0 to 1 with 100 values for the three axes (*i.e.* x,$y.z$). Following the same reasoning of Section 3.2, their combination generates all points of the cube that is stored first in a modification of a numpy meshgrid[14] and eventually turned into a dataframe. Eventually, only the datapoints below the plane defined in Section 3.2 (*i.e.* $z < 1 - x - y$) are selected and their performance metric displayed within the geometrical structure.
As for the second application (*i.e* Section 4.2), the input data structure has been used to plot with a line the coordinates of the curve and the trajectory, respectively for the 2D and 3D geometrical environment.
A similar procedure has been undertaken for the first plot (*i.e.*Figure 4.13) of the third application (*see* Section 4.3). Instead, the subsequent dynamic visualisation of the performance throughout the epochs has required a different approach with the plotting of a scatter 3d plot with the assignment of the animation frame[15] and hover name[16] to the epochs and the color to the performance metric (*i.e* MCC).
Eventually, the fourth application follows a similar reasoning with the added possibility to interact with the result by selecting only the CT projections associated to one specific resampling strategy through a widget.

Other adopted libraries have been Matplotlib and Seaborn mainly for the plotting of the ROC curve and other minor visualizations.

[13]linspace:evenly spaced sequence in a specified interval
[14]meshgrid: returns coordinate matrices from coordinates vectors
[15]to define and add the slider according to the epochs
[16]to interactively display the label of the epoch associated to the point

4 Applications

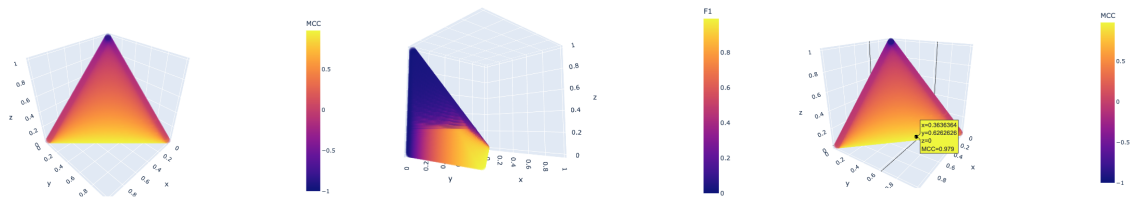
To illustrate the value offered by the Confusion Tetrahedron, the use of this new visual approach has been tested in several applications. This section explains the relevance of the geometrical view through its use in the evaluation of the learned models compared to the standard assessment approach in four main scenarios.

The first application exemplifies the added value offered by the three-dimensionality in order to have a representation of the global landscape for the classifier. The second application illustrates the trajectory of the confusion matrices resulting from the variation of the probability threshold in a 3D space. It compares the obtained visualization with that of the Receiver Operating Characteristic (ROC) curve and argues that the former provides a more complete outlook on the behavior of the confusion matrices with respect to what is observable in the bi dimensional (2D) plane defined by Sensitivity (SNS) and SPEC. More specifically, two different CT trajectories can result in the same 2D representation hampering the possibility to effectively discriminate the behaviour. Then, with the third application, the CT is adopted for a temporal analysis of the performance of a neural network. Here, the scenario illustrates the advantages of the tetrahedron when assessing Neural Networks (NN) models in order to appreciate the evolution of the performance throughout the different epochs of the learning process. Lastly, the fourth application briefly presents the added value provided by the tetrahedron when assessing imbalanced datasets. Here, the CT permits to visually compare the different distribution of confusion matrices according to the type of resampling strategy. All the aforementioned applications aim at explaining the differences in the evaluation of the models between the standard assessment with performance metrics alone as opposed to the proposed strategy of adopting a more visual approach in a 3D environment.

4.1 Global landscape for classifiers

The assessment of a binary classification model is generally done by several performance metrics all presenting different perks and drawbacks, as detailed in Chapter 2. It has already been extensively motivated why there is not one single metric that fits for all use cases and how a pool of combined indicators should always be examined for the evaluation. However, even when pondering over more than just one metric, there is an underlying pitfall in the numerical approach alone. In fact, by assessing only numerical metrics, the information contained in the model is flattened to few numbers. These alone can be misleading and difficult to interpret, especially for non technical stakeholders, but more importantly they tend to overlook the environment in which the model learns. Integrating the evaluation of the performance metrics with a more graphical tool not only it would be helpful in clarifying the numbers but also the interpretation would be enriched with all the information that gets left out from the computation of one single metric. In this perspective, the objective of this application is to employ the Confusion Tetrahedron as a visual tool to represent the global landscape of the binary classifier aiding the numerical evaluation of the performance metric.

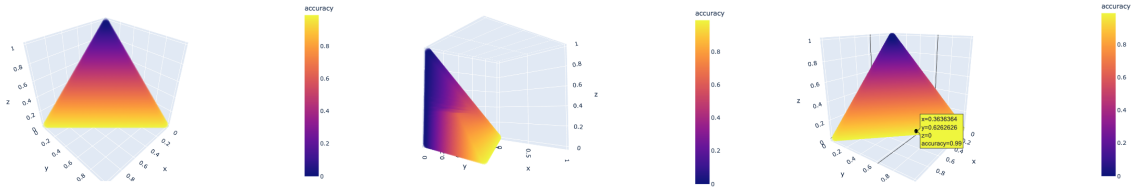
To this end, the Confusion Tetrahedron as obtained in Chapter 3 offers the opportunity to tailor the performance metric associated with each of the confusion matrices composing the tetrahedron. In fact, originally the CT has been plotted with the MCC (*see* Figure 4.1) due to the perks analyzed in Chapter 2. Here, it is shown how the tetrahedron and hence the global classifier performance would differ if other metrics were to be chosen. In particular, to exemplify the results, *accuracy* (*see* Figure 4.2) and the F_1 *score* (*see* Figure 4.3) have been reported here.



(a) Frontal view (b) Side view (c) User interaction

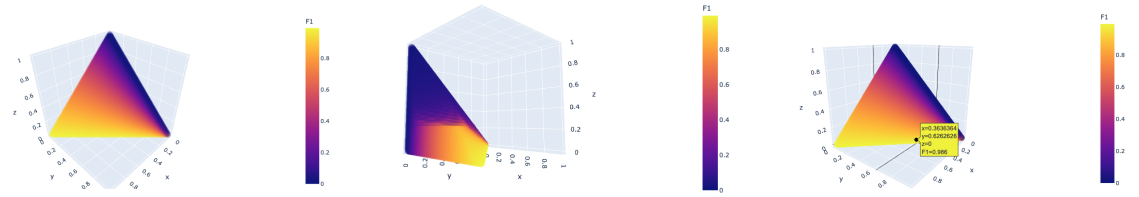
Figure 4.1: Confusion Tetrahedron coloured according to the chosen performance metric, MCC. The three axes represent TP, FP, TN

The visualisation is interactive so in order to display the results, three perspectives have been chosen and reported in Figure 4.4 for each metric. The first is the frontal view (a) of the tetrahedron which is the default result from the execution of the code. Here it is already possible to appreciate differences in the distribution of the colour. In fact, if the frontal view of the MCC suggests a more gradual transition from better to worst models (*i.e.* from yellow to violet), this transition appears more discretized in the case of accuracy.



(a) Frontal view (b) Side view (c) User interaction

Figure 4.2: Confusion Tetrahedron coloured according to the chosen performance metric, Accuracy. The three axes represent TP, FP, TN



(a) Frontal view (b) Side view (c) User interaction

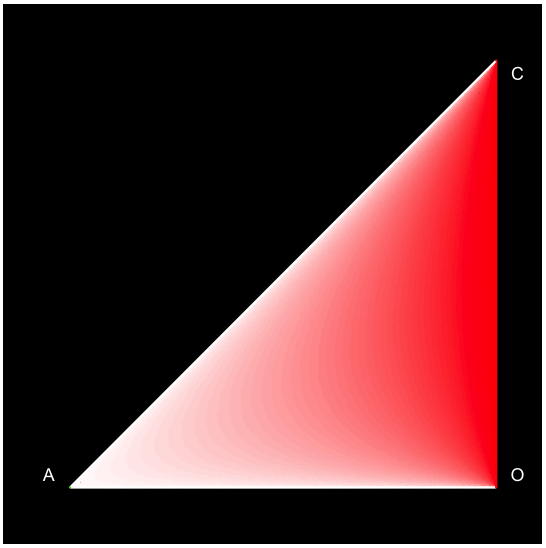
Figure 4.3: F_1 score

Figure 4.4: Confusion Tetrahedron coloured according to the chosen performance metric, F_1. The three axes represent TP, FP, TN

As for the F_1 score, being this metric insensitive to TN, the confusion matrices along the y axis appears to be worst according to the metric.

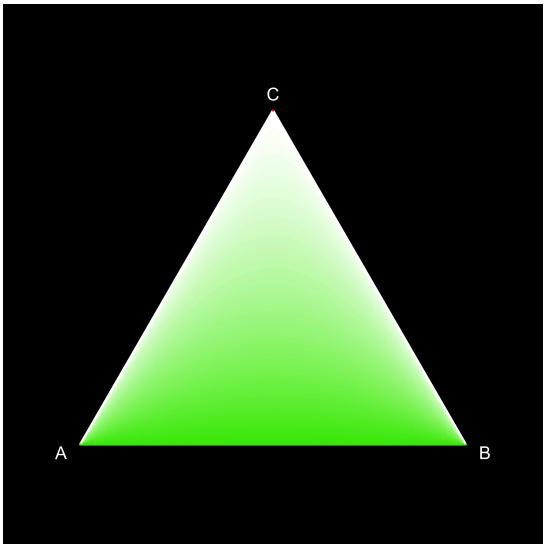
The second perspective (*b*) is given by the side view of the tetrahedron obtained by sliding the three-dimensional interactive visualisation. These illustrate the different behavior of the classification environment depending on the metric. In particular, notice how with accuracy the yellow area is greater than that for the two other metrics. This expresses the pitfall of accuracy which tends to overestimate the performance of binary classification models, especially in unbalanced datasets as in this case. On the other hand, for F_1 it is interesting to notice how the areas are more polarized with less regions colored with intermediate colours.

Lastly (*c*), the third view incorporates the point for the model being evaluated and exemplifies the user interaction with the tetrahedron. The experience here is the same for all metrics and consists in the visualisation of the coordinates as well as the actual values for the chosen metric through the yellow label. To explain the reasoning behind the enriched information provided by the Confusion Tetrahedron, here the behaviour of the metric can be observed also in the bi dimensional plane (*see* Figure 4.5). It can be observed how the holistic view over the behaviour of the metric offered by the 3D environment gets flattened in the 2D plane with a consequent depletion of information with respect to the whole geometrical environment. This stems from the fact that the reduction of one variable forces trajectories that can be perceived as different in 3D to have the same bidimensional aspect. This is detrimental to our objective set of observing the full behaviour.



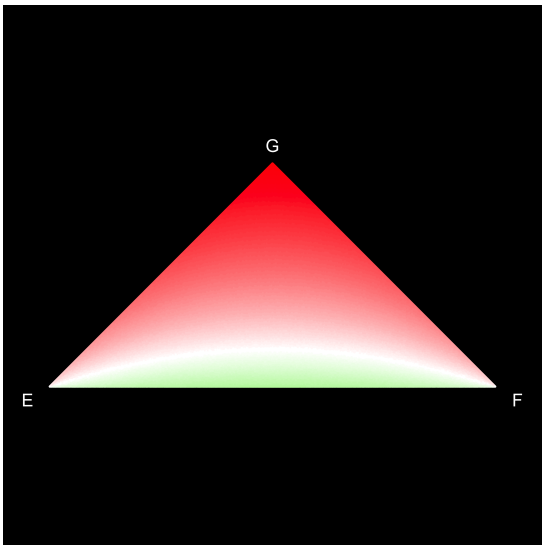
△

(a) AOC: 2D section of the plane defined by the equation $TN = 0$ ($y = 0$) that identifies the vertices $A = (1, 0, 0)$, $O = (0, 0, 0)$, $C = (0, 0, 1)$



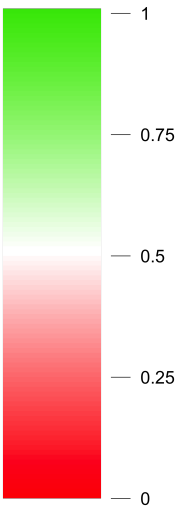
△

(b) ABC: 2D section of the plane defined by the equation $FN = 0$ ($1 - x - y - z = 0$) that identifies the vertices $A = (1, 0, 0)$, $B = (0, 1, 0)$, $C = (0, 0, 1)$



△

(c) EFG: 2D section of the plane defined by the equation $FP = 0.5$ that identifies the vertices $E = (\frac{1}{2}, 0, \frac{1}{2})$, $E = (\frac{1}{2}, 0, \frac{1}{2})$, $E = (\frac{1}{2}, 0, \frac{1}{2})$



(d) Gradient associated to the value of the performance metric: color palette generated for the display of the 2D sections through the use of color generator and R

Figure 4.5: The 2D Cartesian plane representation of sections of the Confusion Tetrahedron identified by different splitting equations. Points in planes not orthogonal to the axes were projected using the barycentric coordinates. The barycentric coordinates of a point can be interpreted as masses placed at the vertices of the simplex, such that the point is the center of mass (or barycenter) of these masses.

It results that the confusion tetrahedron allows to assess the whole behavior of the chosen performance metric and see where the fitted model stands with respect to the whole environment (*i.e.* all the potential models that could have been fit). Thus, it successfully aids in the interpretation and understanding of the numerical indicators as our objectives set required.

4.2 Trajectories and ROC in Confusion tetrahedron

After building a classification model, possible strategies for the evaluation of ML models involve numerical indicators with the computation of performance metrics but might as well include more visual tools. A common graphical approach to the evaluation of performance metrics consists in the ROC curve. This is a bidimensional plot showing the performance of a classification model for all values of the probability threshold selected in the formulation of the actual prediction. In particular, the curve is a probability curve that plots two parameters:

- the TPR [1], also called Recall or Sensitivity, on the y-axis
- the FPR [2] $= 1 -$ Specificity on the x-axis

These values are visualised for all the possible values of the threshold, which is the minimum probability requested to classify the element as positive

$$P(Y = 1) \geq \text{threshold} \qquad \text{with } t \in [0, 1] \tag{4.1}$$

In fact, a ML classification model can be used to predict the actual class of the data point *directly* or predict its *probability of belonging to different classes*. The latter tends to give more control over the model as one can tailor the threshold according to the specific requirements of its application setting. For different levels of the threshold, different confusion matrices are produced. Hence, relevant information is stored in the variation of such metrics as a consequence of threshold setting. This is precisely what the ROC curve offers in the bidimensional plot. A change in the level of the threshold modifies the classification of the positive class which gets reflected in a variation of sensitivity and specificity and hence a different point in the plane identified by the ROC curve.

Alternatively, the same information can be plotted in a 3D plot producing a trajectory instead of a curve. In fact, despite the ROC curve represents a step towards the objective of the present work of incorporating more visual tools in the evaluation of the classification algorithms, the bidimensional representation might produce a potentially flawed visualisation of the performances of alternative models. In fact, two ROC curves that display a similar

[1] $\frac{TP}{TP+FN}$
[2] $\frac{FP}{FP+TN}$

line on a 2D geometrical environment, might hide a profoundly different behaviour that can only be discovered by bringing the same information on a three dimensional environment.

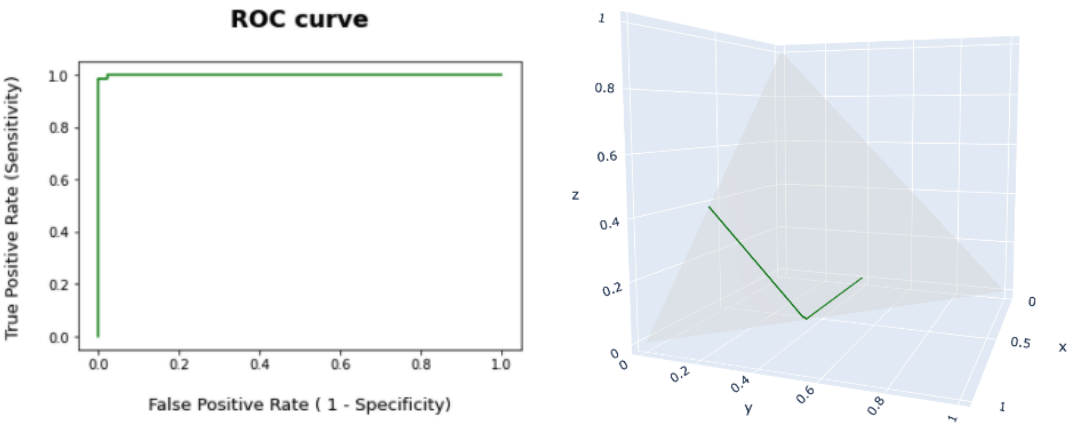
To this end, the trajectory of the confusion matrices for all levels of the threshold has been plotted in the CT. From the results, it is clear how seeing a trajectory in a 3D geometrical environment as the Confusion Tetrahedron enriches the information provided by the same set of input. Moreover, it allows the discrimination of particularly critical situations such as those for which the 3D trajectory would be flattened to the same 2D curve.

To exemplify the visualisation results and the range of possible trajectories that one can obtain from different kind of models, three different scenarios are included in the present analysis.

The first is the case of a good performance (*see* Figure 4.6) where almost all observations are correctly predicted. The input model for this ROC curve can be deemed as a good one because as the threshold increases (*i.e.* it is required for the single observation to have a higher probability of belonging to the positive class), the TP are maximized while little change can be observed in the rate of FP. Similarly, in the CT as the threshold increases

To display the range of motion, it has been plotted also an intermediate performance (*see* Figure 4.7) and the worst performance (*see* Figure 4.8), that is the case of a random classification.

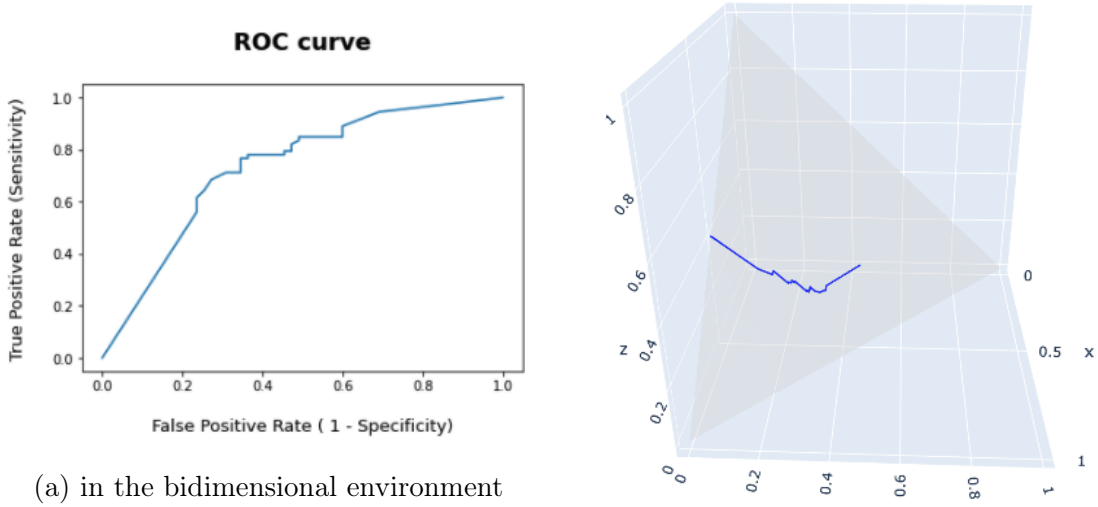
Overall, a common pattern can be observed between the ROC curve and the CT trajectory but the information is far more enriched in the 3D environment with the possibility to detect differences in the behavior of models that would be considered similar otherwise. Moreover, in the interactive plot there is also the additional advantage of the dynamics of the visualisation. In fact, as changes occur in the value of the threshold, the visualisation immediately reproduces the point on the plot facilitating the understanding of the link between the change in the threshold and that in the confusion matrix.



(a) in the bidimensional environment

(b) in the three dimensional environment

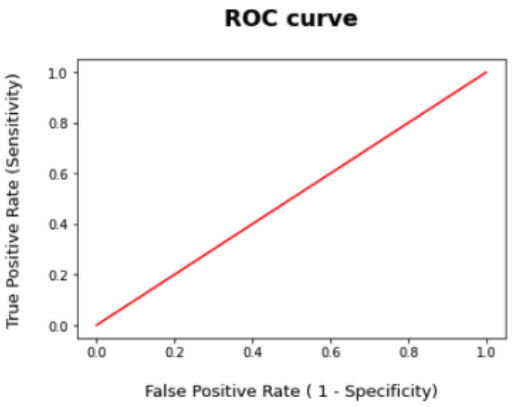
Figure 4.6: Behaviour of a model for all levels of the thresholds. Case of a good performance: comparison among the picture provide by the bi dimensional plane with the ROC curve and the three dimensional environment of the CT. The three axes represent TP, FP, TN



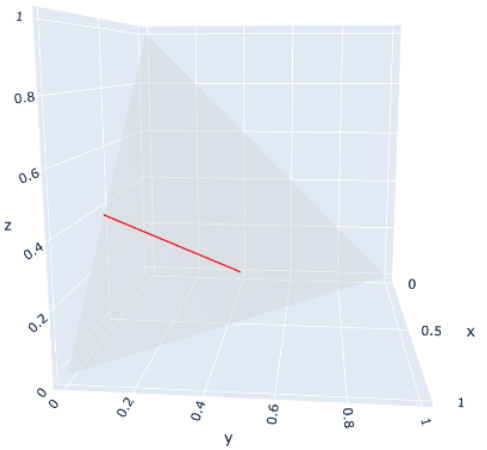
(a) in the bidimensional environment

(b) in the three dimensional environment

Figure 4.7: Behaviour of a model for all levels of the thresholds. Case of an intermediate performance: comparison among the picture provide by the bi dimensional plane with the ROC curve and the three dimensional environment of the CT. The three axes represent TP, FP, TN

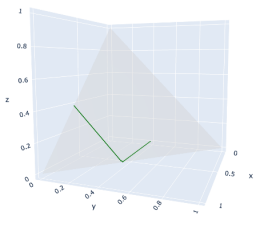


(a) in the bidimensional environment

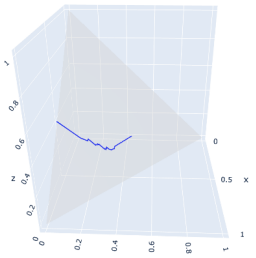


(b) in the three dimensional environment

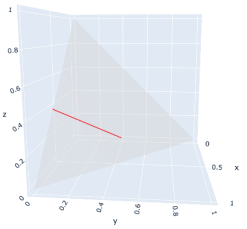
Figure 4.8: Behaviour of a model for all levels of the thresholds. Case of a poor performance: comparison among the picture provide by the bi dimensional plane with the ROC curve and the three dimensional environment of the CT. The three axes represent TP, FP, TN



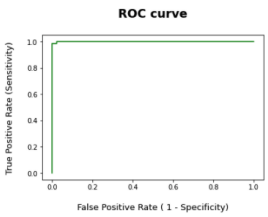
(a) Good CT trajectory



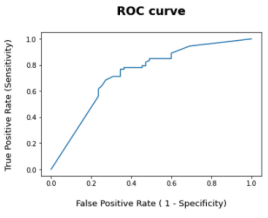
(b) Intermediate CT trajectory



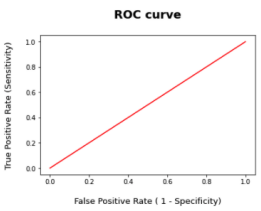
(c) Poor CT trajectory



(d) Good ROC curve



(e) Intermediate ROC curve



(f) Random ROC curve

Figure 4.9: Summary of the possible views. The three axes in (a),(b),(c) represent TP, FP, TN

4.3 Trajectories in training epochs

The objective of this application is to extend the visual evaluation approach to the temporal analysis of NN. Such deep learning models have achieved state-of-the-art accuracy for many prediction tasks but understanding them still remains a challenge as little work has been done to monitor the fitting process and analyse their performance throughout the epochs. The literature expresses a growing interest in the adoption of visual tools in the emerging field of deep learning due to their positive impact on the interpretability and further understanding of complex models [48] [49] [50] [51] [52]. A first step towards the uncovering of the learning dynamics of a neural network has been made with the development of visual tools such as AcutiVis [53] and ConfusionFlow [54] but still the field presents flourishing research opportunities that the present work intends to develop.

To illustrate what is meant for *temporal analysis* a brief explanation of the inner working of NN is needed. Neural networks are a subset of ML within the Deep Learning (DL) realm that are designed to recognize patterns in the data mimicking the way that biological neurons signal to one another. These computational models are comprised of node layers, containing an input layer, one or more hidden layers and an output layer (*see* Figure 4.10). The nodes (*i.e.* neurons) connect to each other to transfer the data according to specific

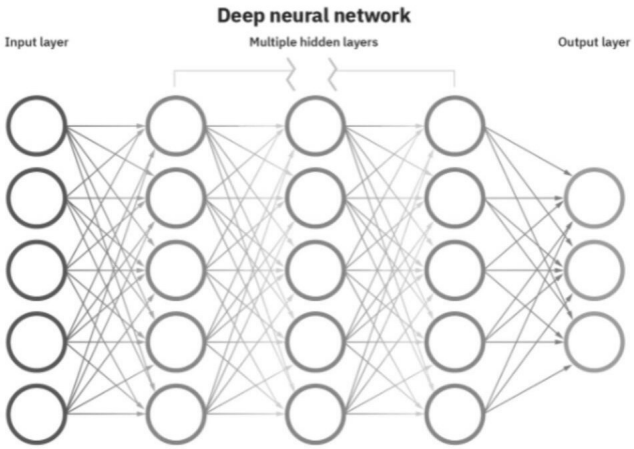


Figure 4.10: Representation of the learning process of a Neural Network

rules. Each of the nodes has an associated weight and threshold since only if the output of any individual node is above this specified threshold value, the node activates and sends data to the next layer of the network. If the value does not surpass the threshold no data is passed along to the next layer of the network. This is what happens at the most granular level when training a neural network and it serves the ultimate goal of gradually updating the weights associated with the nodes. This process is repeated several times collecting a number of *intermediate models* of the NN. In order to eventually obtain one final model, the network parameters are optimized iteratively over all these intermediate results using a model selection procedure.

The number of intermediate results is defined by setting a certain value for the *epochs* which consists in the number of times that the data is fed to the NN for its training. This is a hyperparameter of the model that depends on several variables such as the way in which the data is input in the network (i.e. one by one, mini-batches, entire batch), the network architecture and the chosen loss function. However, its tuning goes beyond the purposes of this analysis which focuses on emphasising the great opportunity for better performance evaluation and interpretation through the leverage of the intermediate results produced by each epoch.

In this perspective, the Confusion Tetrahedron constitutes a useful tool for a more informed evaluation process of NN on binary classification tasks. In fact, currently to monitor the learning process and track model quality, the training is accompanied by accuracy and loss curves throughout the epochs. Eventually, the performance of only the final trained model is summarized using the confusion matrix. However, by doing so the final model alone is assessed and visualised through the CM, disregarding the information contained in all intermediate results of the learning process. Condensing each of the intermediate performances that the model progressively achieves in a standard accuracy and loss curve brings two main drawbacks that affect the implementation and adoption of NN. The first is suffered by a more technical public and consists in the loss of the information stored in the fitting process which hampers the amount and quality of insights that the researcher could acquire from the visualisation of the learning flow . The second pertains to a broader public of non technical stakeholders who, due to the growing adoption of automated applications in business settings, might need to at least have an understanding of the process and the results. Such problems are particularly severe when the model does not perform satisfactorily because having no tool that visualizes the intermediate performances hampers the interpretability and understanding of the process. Ultimately, this is detrimental to an informed and reasoned adoption of the models. In order to overcome these issues, it would be advisable to enable a temporal analysis of the classifier confusion throughout the whole epochs within the learning environment. An effort to perform this analysis has been done here with the plotting of each epochs' result within the geometrical environment of the Confusion Tetrahedron.

To produce the visualisation of the intermediate performances, a neural network has been trained on the cleaned dataset as resulting from the subsection 3.3.2. This has been split into training and testing subset with the latter being 20% of the whole records. Using Keras [3], a neural network performing binary classification on the presence or absence of chronic kidney disease has been built (*see* Figure 4.11).

More in detail, the implemented neural network has been obtained by:

1. instantiating a *Sequential* (*i.e.* a model that is created sequentially where the output

[3]high-level framework based on tensorflow

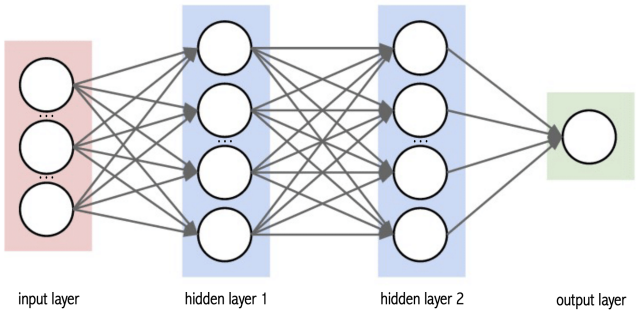


Figure 4.11: Architecture of the Neural Network built for the classification of the chronic kidney disease. The architecture consists of an input layer, 2 hidden layers and one output layer.

of each added layer is the input of the next layer) and *Dense* (*i.e.* fully connected) model;

2. feeding the initial data to the NN through an *input layer* (red in Figure 4.11). This has been defined to have 128 neurons, an input dimension of 48 (*i.e.* number of attributes) and the Rectified Linear Unit (ReLU) (Equation 4.2) has been chosen as activation function .

$$f(x) = max(0, x) \tag{4.2}$$

3. according to certain criteria, the data is then forwarded to the *hidden layers* (blue in Figure 4.11). These are the intermediate layers between input and output and the core of the computation for the updating of the weights. In the developed implementation, the network has two hidden layers, respectively with 24 and 32 neurons and both with the ReLU as activation function;

4. eventually in order to obtain the final prediction, the data enters the *output layer* (green in Figure 4.11). This is the step in which a prediction for the record is obtained through a different activation function that is the *sigmoid function* (Equation 4.3)

$$S(x) = \frac{1}{1 + e^{-}x} = \frac{e^x}{e^x + 1} \tag{4.3}$$

5. for the *fitting* of the final the model the specified loss function is the Binary Cross Entropy and the optimize is the Adam. For each intermediate model, the *accuracy* and the MCC[4] has been stored as performance metric of the model.

   The training of the NN has been performed with a 100 epochs and setting a relatively batch size of 10 records due to the low numbers of records in the dataset.

[4]the MCC has been calculated from the normalized values of TP, TN, FP extracted from each epoch with a callbackfunction

Eventually, this process leads to the building of the neural network and the obtaining of the predictions. Despite this being the final product of the NN, the effort is just at the beginning. In fact, then it comes the time for the evaluation of the model.
The standard evaluation process of NN consists in plotting the evolution of the training and validation accuracy for the model (*see* Figure 4.12).

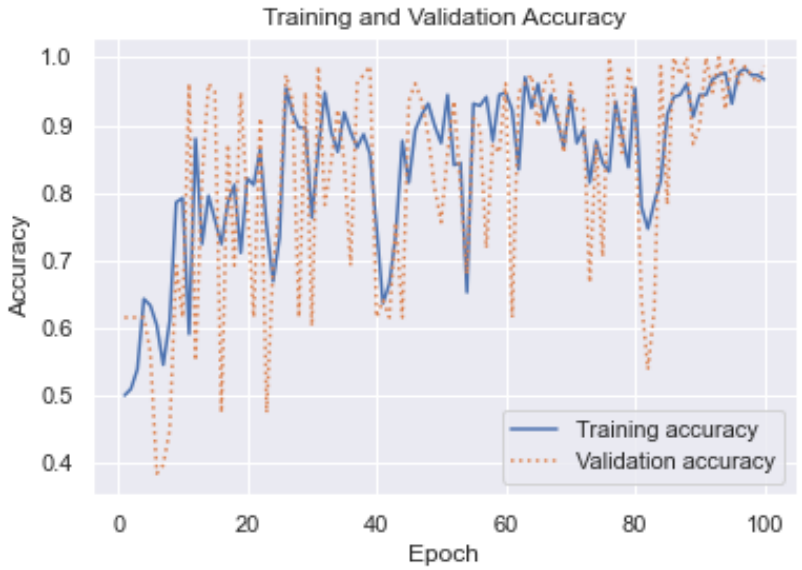


Figure 4.12: Neural network: accuracy throughout the epochs

This allows us to appreciate how the untrained network gradually grows from an almost random classifier progressively into better performing (at least according to accuracy) classifiers.
However, this visualisation is static and does not provide any additional information on the environment in which the learning dynamics occur.Here, the Confusion Tetrahedron offers an added value in the interpretation and understanding of the inner flow of the NN.

Here (*see* Figure 4.13) the orange points represent the projection in the tetrahedron of the confusion matrix extracted from each epoch. Instead, the lines connecting each point represent the trajectory followed in the learning flow so they are dashed as they do not represent actual values but conveys the movement.

A similar evolution to Figure 4.12 can be appreciated in the following results. However, by plotting dynamically the results for each epoch within the three dimensional tetrahedron environment, researchers can have a full snapshot of the dynamics of the algorithm.

Starting from the first few epochs (Figure 4.14) where the NN appears to perform poorly according to the MCC as depicted by the violet colour of the dot.

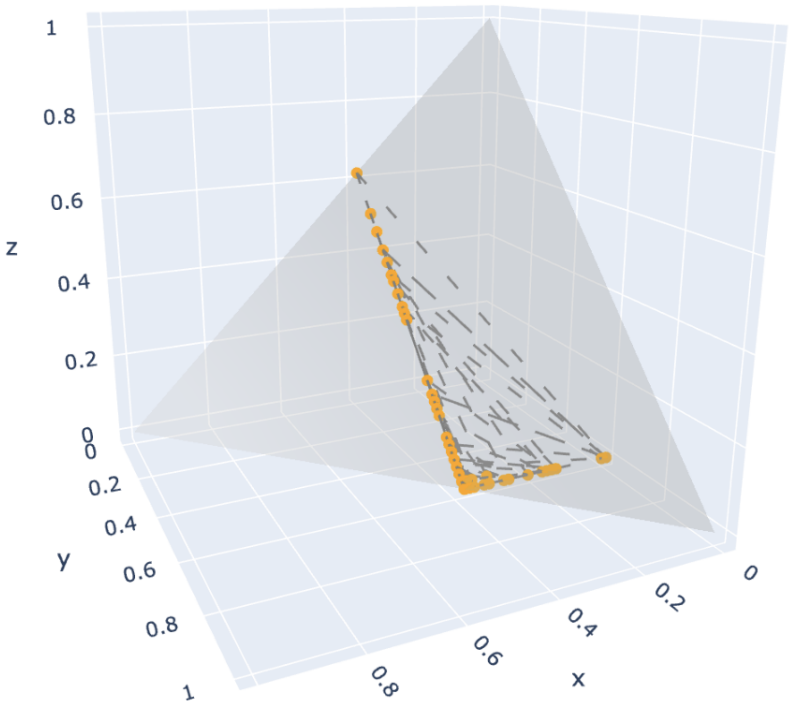


Figure 4.13: Neural network - Results for each epoch (orange points) and the associated trajectory (dashed grey line). The three axes represent TP, FP, TN.

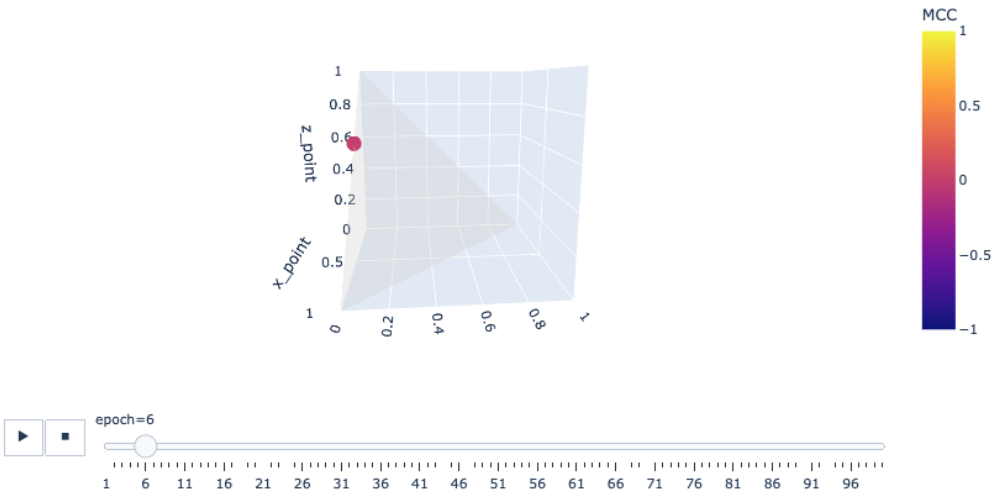


Figure 4.14: Neural network: Confusion Tetrahedron result in epoch = 5. The three axes represent TP, FP, TN.

Throughout the iterations, the weights of the nodes are updated and the performance progressively improves as illustrated by the gradually turning of the colour from violet to lighter shades (Figure 4.15)

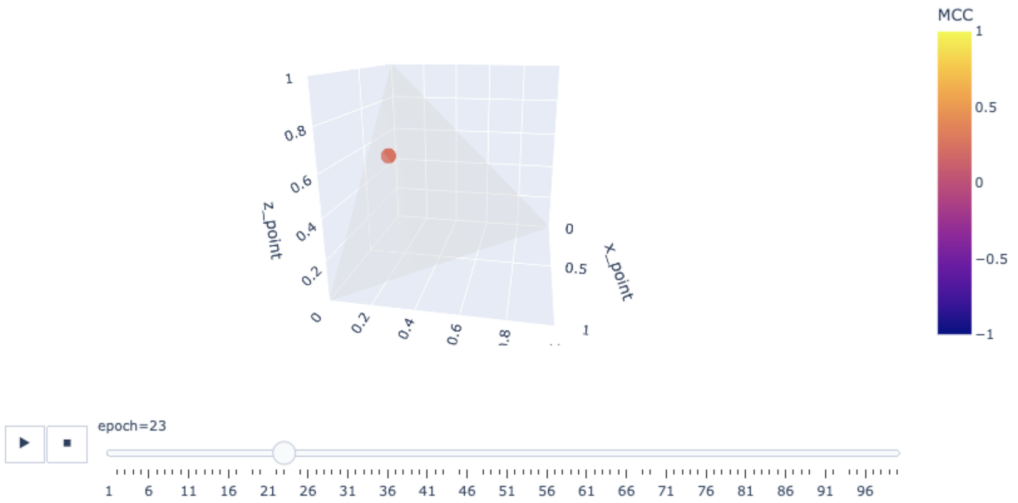


Figure 4.15: Neural network - Confusion Tetrahedron result in epoch=23. The three axes represent TP, FP, TN.

When the model appears to start shifting toward better performances, occasionally it comes back to areas associated with worst performances and throughout the epochs tends to scatter around the learning environment (Figure 4.16).

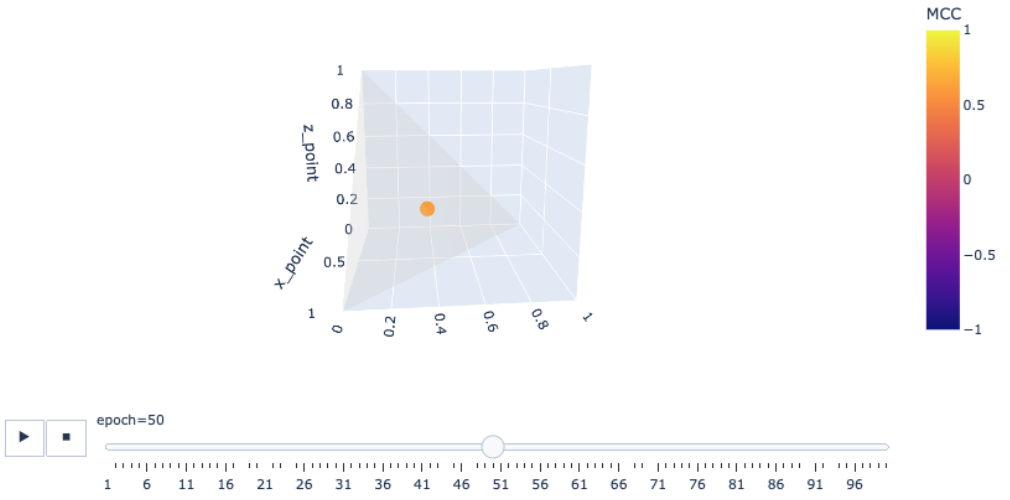


Figure 4.16: Neural network - Confusion Tetrahedron result in epoch=50.The three axes represent TP, FP, TN.

As the training continues, the performance improves and the resulting dot turns into more yellow shades communicating higher MCC (Figure 4.17).

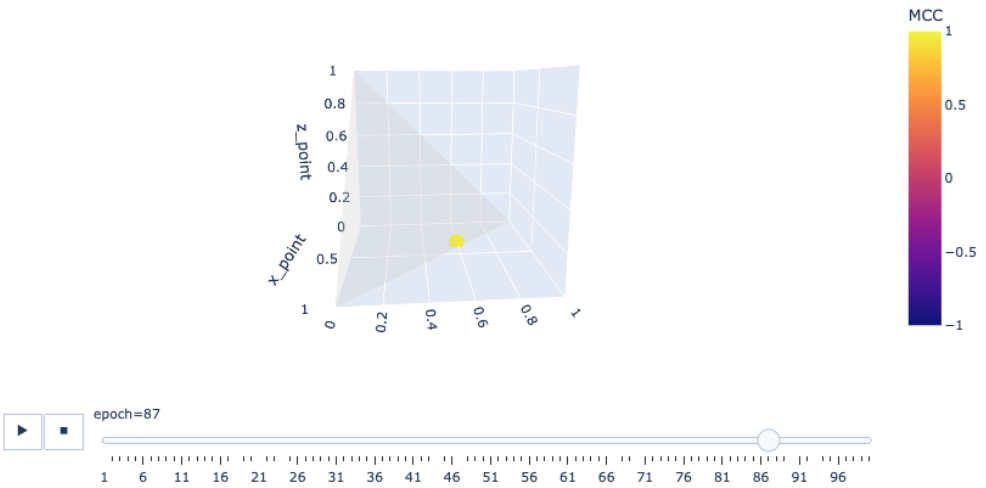


Figure 4.17: Neural network - Confusion Tetrahedron result in epoch = 87.The three axes represent TP, FP, TN.

Eventually (*see* Figure 4.18) the NN reaches the area of the best classification performance as illustrated by the yellow colour of the dot delivering the information of a high MCC.

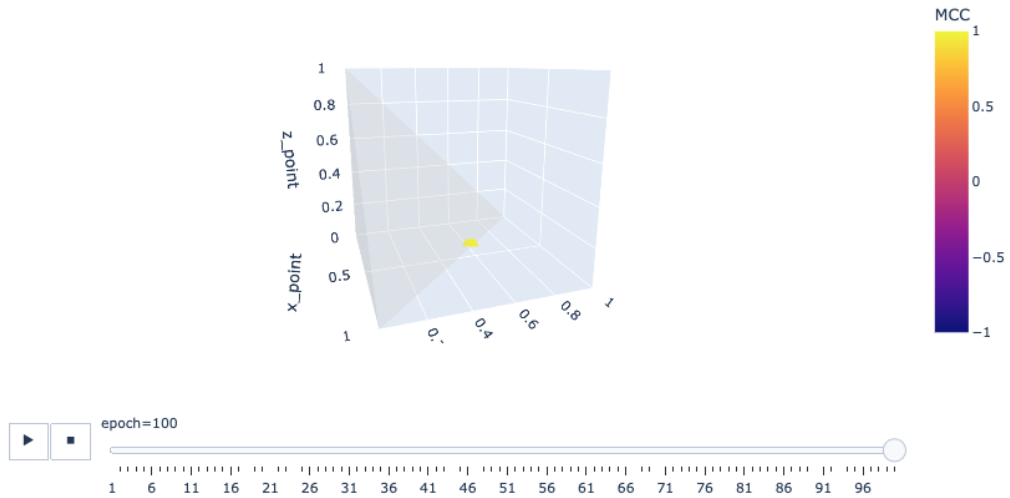


Figure 4.18: Neural network - Confusion Tetrahedron result in epoch=100.The three axes represent TP, FP, TN.

As a result, it provides an interactive and comparative visualisation of the model confusion over the whole epochs within the learning environment (*see* Figure 4.19).

This result is relatively remarkable considering that the possibility to track and visualise the model confusion across epochs is currently lacking. Through the plotting of the results of each epoch within the Confusion Tetrahedron it is possible to tap into the dynamics of the learning flow and visualize the behavior of the algorithm. This represents an advancement with the respect to the current evaluation approach of neural networks in classification

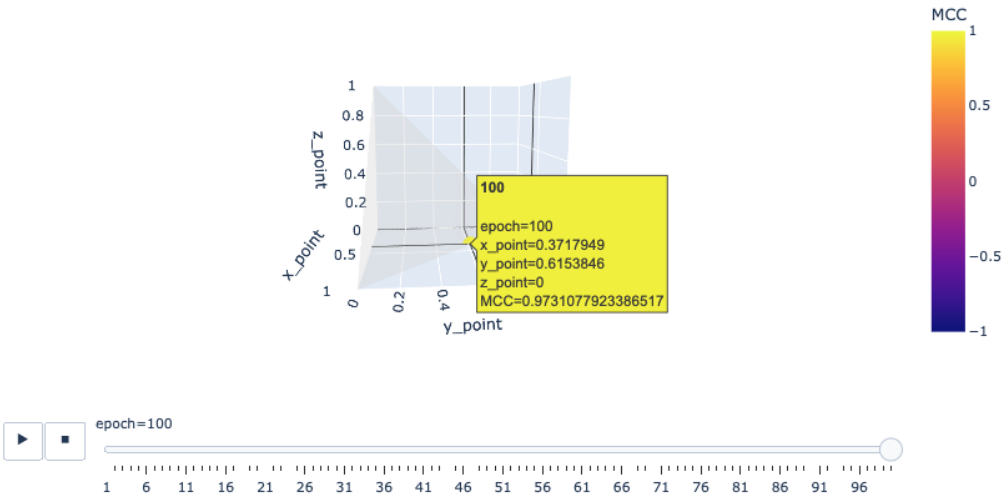


Figure 4.19: User interaction

tasks and a first step towards a more critical design process for network architectures.

4.4 Stratified vs non stratified resampling

Since most algorithms are designed to maximize accuracy and reduce errors, machine learning algorithms work best when the number of samples in each class are similar. However, class imbalance is a common feature that appears in many domains ranging from fraud detection, spam filtering, disease screening or advertising click-through. Having great difference in the number of samples belonging to one class poses significant problems in the evaluation of classification algorithms because one can obtain relatively high accuracy on the test set that falsely depicts the actual circumstances. This represents an issue that is caused by the accuracy paradox for which the performance of the model, generally assessed through accuracy, appears to be excellent while it is only reflecting the underlying distribution.

This is clearly detrimental to the evaluation of the models so there are various actions to tackle this issue such as adopting alternative performance metrics (*see* Section 2.3) or implementing Cross Validation (CV) techniques. Resampling techniques are used to separate training and test set by permutating the elements in the dataset used to build and test. These can be *stratified* or *non stratified*. The former technique returns stratified randomized folds made by *preserving the percentage of samples for each class*. Instead, with the latter, the distribution of samples among each class is not kept in the generated randomized folds.

Due to the different generation process, the performance of the fitted models will differ according to the adopted strategy. In particular, the stratified resampling technique achieves better performance and the more imbalanced the dataset, the greater the gap between the results of the two resampling strategies.

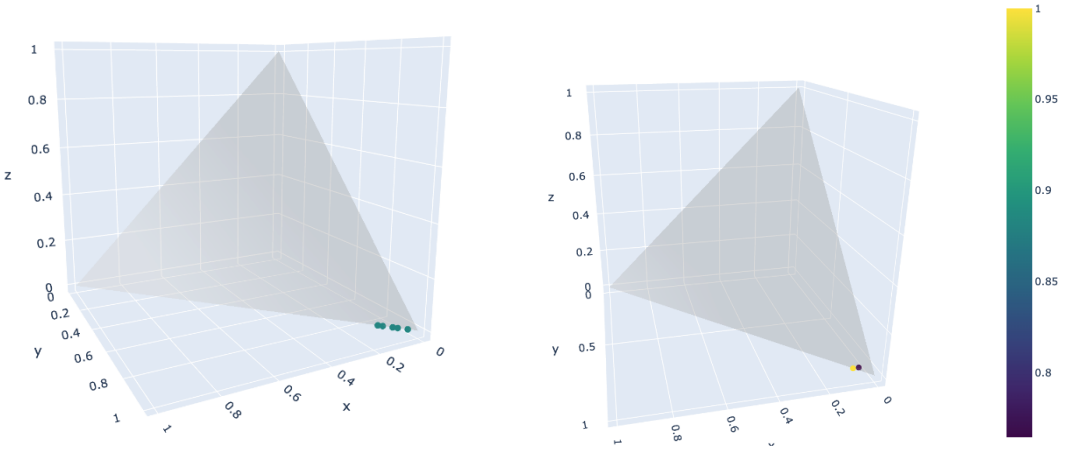
In this perspective, the CT can effectively aid in the visualization of the results for each resampling strategy. This geometrical environment permits to appreciate how all models generated from stratified randomized folds will concentrate together generally closer to the perfect classification line. On the other hand, the results for the models generated by the simply randomized folds tend to be more distributed and report a worst performance.

In order to represent the result, starting from an unbalanced dataset D, with about 95% samples of class 0 and 5% samples of class 1, the two alternative resampling strategies have been applied. In particular, two sets M_1 and M_2 of Montecarlo CV sets have been built,

- each consisting of 10 random splits (or a higher number) of the original dataset D $M_1 = \{(T_i^1, V_i^1) : 1 \leq i \leq 100\}$ and $M_2 = \{(T_i^2, V_i^2) : 1 \leq i \leq j\}$
- such that, for each i, T_i^k includes a random 75% (or a different percentage) of D and V_i^k the remaining 25%,
- with the difference that, in M_1 the split is completely random (not stratified), while

in M_2 the split is stratified, thus in both T_i^2 and V_i^2 the ratio between the samples of class 0 and the samples of class 1 is the same as in the original dataset D.

Afterwards, a SVM model has been trained and it has been applied on V_i^k, computing the resulting performance (e.g., by MCC) on V_i^k. Thus the obtained set of performances for each kind of technique (*i.e* 10 values of MCC for the splits M_1 and 10 values of MCC for the splits M_2) can be displayed within the CT successfully visualizing the different performance achieved by the two techniques.



(a) Results for the non stratified resampling strategy: CT coordinates and MCC for each of the 10 fitted models

(b) Results for the stratified resampling strategy: CT coordinates and MCC for each of the 10 fitted models.

Figure 4.20: Comparison of the results for the two alternative resampling strategies

Overall (*see* Figure 4.20), all models tend to stand on the right of the environment (*i.e.* high TN) and closer to the basis of the Confusion Tetrahedron (*i.e.* low FP). This comes at no surprise given that ML models will generally be correct in predicting a negative class in the case of datasets with a great imbalance towards the negative class. However, what the CT helps in visualizing is the different distribution of the models according to the alternative resmapling strategy. In particular, the set of models from the stratified resampling strategy (*see* Figure 4.20b) tend to cluster together more than the set of models resulting from the non stratified resampling strategy (*see* Figure 4.20a)

This result can be further analyzed and appreciated by zooming on the resulting set of models in Confusion Tetrahedron (*see* Figure 4.21).

(a) Results for the randomized resampling strategy: zoom on the CT coordinates to better observe the distribution.

(b) Results for the stratified resampling strategy: zoom on the CT coordinates to better observe the distribution.

Figure 4.21: Comparison of the results for the two alternative resampling strategies: zoom on the resulting models and distribution

5 Conclusions

Data visualization is a fast growing area of data science with a great potential impact for supporting ML research, and, in particular, for model's evaluation and understanding. However, little work has been done to research alternative visualisations for the results of machine learning models. Here, data visualization is proposed as a pivotal tool to support and enhance ML operations by supplying practitioners with greater guidance on how to decide upon the adoption and deployment of ML algorithms. The present work aims at showing an effective case study demonstrating the helpful support that an actionable data visualization tool can provide to a machine learning pipeline, extending this research area by focusing on the visualisation of confusion matrices towards the model interpretation and evaluation stage.

In particular, after a comprehensive excursus of the performance metrics adopted for the evaluation of the confusion matrix, we discuss to what extent the evaluation should occur not with the metric alone but also with more visual approaches. Hence, in Chapter 2 an extensive review is shown of the available classical visualisation designs for the confusion matrix, being this the basis the computation of every metric. All these measures share one common feature that is the bidimensionality of the visualisation which comes at a cost in terms of quality and quantity of information provided and hampered complete understanding of the geometrical environment in which the model learns . As a result, in Chapter 3 a contribution to the advancement of the field has been made by proposing a novel geometrical 3D visualization called the *Confusion Tetrahedron*. This environment is built on three out of the four normalized entries for the confusion matrix that are TP, TN and FP and, leveraging the facts that the fourth entry is dependent on the other three for a given number of samples and most of the measures are only relying on the mutual ratios of the entries, it contains all equivalence classes of confusion matrix that can result from a learning model.

As a result, the 3D view of all possible confusion matrices – or, better, of their equivalence classes w.r.t. the dataset size – provides a more complete perspective over the learning process and better depicts the geometrical environment in which the model learns. This is beneficial not only for a more intuitive and interactive visualization of the geometry of the problem but also as it provides additional information that would not be observable otherwise. In particular, this 3D view drives the realization of a more effective comparison of the performance metrics, to obtain an holistic view of the behaviour of the models and

their trajectories and to have a full snapshot of the dynamics of the algorithm for ML applications.

To illustrate these advantages, a full implementation on a classification task has been performed and four applications have been developed. The first in Section 4.1 depicts the added values provided by the possibility to have an holistic view over the whole geometrical environment. The second in Section 4.2 illustrates the behaviour of the confusion matrix as the classifier threshold value varies, thus highlighting how the Confusion Tetrahedron allows the discrimination behavior that would not be possible to distinguish in the 2D plot of the ROC curve. The third use case in Section 4.3 provides a dynamic interactive visualization of the inner learning flow of a neural network by plotting the projection in the tetrahedron environment of the confusion matrix produced in each epoch. Eventually, the last application in Section 4.4 tackles the evaluation of imbalanced datasets and more specifically exemplifies how the Confusion Tetrahedron allows the visual comparison of the different distribution of confusion matrices stemming from alternative types of resampling strategies.

The results of this work needs to be read in light of some considerations. First of all, the fact that here the starting point for the evaluation of a machine learning model has been identified in the confusion matrix and in its projection in a 3D environment. However, the assessment of a model should not disregard all the aspects that comes before the generation of the predictions. In fact, several errors might have occurred in the data collection and processing (*i.e.* Extract, Transform, Load (ETL) pipeline) stages that would not be reflected by any metric nor visualization over an environment for the principle that flawed input data produce non sense output (*i.e* Garbage In, Garbage Out (GIGO))).

Nonetheless, as with the majority of the evaluation methodologies, the proposed approach is subject to some limitations that may be addressed in the future. The primary limitation to this evaluation approach is that it triggers from the confusion matrix so it always adopts only four entries of a matrix (three according to our representation) and summarizes the performance coloring the environment with a metric. However, this present an intrinsic issue that taking (certain transformations of) just few numbers as a proxy for model quality can be deemed as too much of a shortcut. An approach that would be more balanced and reflective of real circumstances should also take into account a cost function for the error [55][56]. That is, especially for practical deployment settings, the assessment of a model should not categorize predictions just as correct or wrong. Instead, it should associate a weight to correctness of a model according to the real world consequences of different predictions so to better capture the value delivered by the model in practical scenarios.

Moreover, the added value provided by the three dimensional visualization is fully realized only as long as the visualisation is interactive and dynamic. In fact, the proposed visualization and the illustrated application scenarios all serve the ultimate purpose of actively

engaging the human in the loop being this a researcher who can harness the additional information provided by the 3D view or the average non technical stakeholder who can benefit from a more intuitive visualization of the results.
As a result, this visualisation approach to the evaluation of the models would unleash its full potential with the development of an interactive tool allowing the insertion of one's own confusion matrix and display the results.
Hence, the next steps for the present work will consists in the creation of a library ready to be pip installed followed by a web application to enable interactivity and personalisation.

Overall, the field of data visualization in machine learning offers growing potential for interesting research opportunities such as *what-if analysis* for outliers detection in the data preprocessing step which would aid in the tackling of the first consideration exposed. Moreover, particularly valuable would be to follow up with the attempt of combining data visualization with data debugging.

To sum up, transforming abstract mathematical data into physical visions is a powerful means to represent the environment and the inner working of machine learning models. Data visualization tackles the untapped potential behind data offering great value both for technical and non technical stakeholders in a perspective of AI democratization. This principle has been exemplified with the case of the Confusion Tetrahedron which not only facilitates the interpretation of the result but also offers an holistic and dynamic view over the behavior of a model providing information that would not be observable otherwise. Together with the theoretical support for the construction of the Confusion Tetrahedron, the present work provides ML practicioners with an off the shelf solution readily available for the use and test of the proposed visualization. The code has been implemented in Python and R to serve both communities of users.

Bibliography

[1] F. Viègas and M. Wattenberg, "NeurIPS 2018 Tutorial Session on Visualization for Machine Learning," 2018. Tutorial at NeurIPS 2018.

[2] M. Brunato and R. Battiti, *The LION way. Machine Learning plus Intelligent Optimization*. LIONlab, University of Trento, Italy, 2017.

[3] C. Ferri, J. Hernández-Orallo, and R. Modroiu, "An experimental comparison of performance measures for classification," *Pattern Recognition Letters*, vol. 30, no. 1, pp. 27–38, 2009.

[4] R. Caruana and A. Niculescu-Mizil, "Data Mining in Metric Space: An Empirical Analysis of Supervised Learning Performance Criteria," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, p. 69–78, Association for Computing Machinery, 2004.

[5] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *Journal of Machine Learning Research*, vol. 7, p. 1–30, 2006.

[6] D. G. Kleinbaum and M. Klein, *Logistic Regression: A Self-Learning Text*. Springer, 2011.

[7] I. Rish, "An Empirical Study of the Naïve Bayes Classifier," *IJCAI 2001 Work Empir Methods Artif Intell*, vol. 3, 01 2001.

[8] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[9] M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18–28, 1998.

[10] L. Breiman, "Random forest," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[11] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," *CoRR*, vol. abs/1603.02754, 2016.

[12] R. Kohavi and F. Provost, "Glossary of terms. Special issue of applications of machine learning and the knowledge discovery process," *Mach. Learn.*, vol. 30, 01 1998.

[13] L. Oneto, *Model selection and Error Estimation in a Nutshell. Modelling and Optimization in science and technologies*, vol. 15. Springer, 2019.

[14] S. Raschka, "An Overview of General Performance Metrics of Binary Classifier Systems," *CoRR*, vol. abs/1410.5330, 2014.

[15] M. Kohl, "Performance Measures in Binary Classification," *International Journal of Statistics in Medical Research*, vol. 1, pp. 79–81, 01 2012.

[16] M. Hossin and S. M.N, "A Review on Evaluation Metrics for Data Classification Evaluations," *International Journal of Data Mining & Knowledge Management Process*, vol. 5, pp. 01–11, 03 2015.

[17] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[18] C. Parker, "An Analysis of Performance Measures for Binary Classifiers," *2011 IEEE 11th International Conference on Data Mining*, pp. 517–526, 2011.

[19] J. Yerushalmy, "Statistical Problems in Assessing Methods of Medical Diagnosis, with Special Reference to X-Ray Techniques," *Public Health Reports (1896-1970)*, vol. 62, no. 40, pp. 1432–1449, 1947.

[20] H. Brenner, O. Gefeller, F. R. Bender, G. Freitag, and H. J. Rampisch, "Variation of sensitivity, specificity, likelihood ratios and predictive values with disease prevalence," *Stat. Med*, pp. 981–991, 1997.

[21] B. J., "Prevalence threshold (ϕe) and the geometry of screening curves," *PLOS ONE*, vol. 15, no. 10, pp. 1–12, 2020.

[22] J. T. Schaefer, "The critical success index as an indicator of Warning skill," *Weather and Forecasting*, vol. 5, pp. 570–575, 1990.

[23] C. X. Ling, J. Huang, and H. Zhang, "AUC: A Statistically Consistent and More Discriminating Measure than Accuracy," *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, p. 519–524, 2003.

[24] A. Ben-David, "About the relationship between ROC curves and Cohen's kappa," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 6, pp. 874–882, 2008.

[25] F. J. Provost and T. Fawcett, "Robust Classification for Imprecise Environments," *CoRR*, vol. cs.LG/0009007, 2000.

[26] M. Bekkar, H. Djema, and T. Alitouche, "Evaluation measures for models assessment over imbalanced data sets," *Journal of Information Engineering and Applications*, vol. 3, pp. 27–38, 01 2013.

[27] Q. Gu, L. Zhu, and Z. Cai, "Evaluation Measures of the Classification Performance of Imbalanced Data Sets," in *Computational Intelligence and Intelligent Systems* (Z. Cai, Z. Li, Z. Kang, and Y. Liu, eds.), (Berlin, Heidelberg), pp. 461–471, Springer Berlin Heidelberg, 2009.

[28] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, "The Balanced Accuracy and Its Posterior Distribution," in *2010 20th International Conference on Pattern Recognition*, pp. 3121–3124, 2010.

[29] D. M. W. Powers, "What the F-measure doesn't measure: Features, Flaws, Fallacies and Fixes," *CoRR*, vol. abs/1503.06410, 2015.

[30] D. M. W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *CoRR*, vol. abs/2010.16061, 2020.

[31] A. Tharwat, "Classification assessment methods," *Applied Computing and Informatics*, 2020.

[32] A. S. Glas, J. G. Lijmer, M. H. Prins, G. J. Bonsel, and P. M. M. Bossuyt, "The diagnostic odds ratio: a single indicator of test performance," *Journal of Clinical Epidemiology*, vol. 56 11, pp. 1129–35, 2003.

[33] G. Jurman, S. Riccadonna, and C. Furlanello, "A Comparison of MCC and CEN Error Measures in Multi-Class Prediction," *PLOS ONE*, vol. 7, pp. 1–8, 2012.

[34] D. Chicco, "Ten quick tips for machine learning in computational biology," *BioData Mining*, vol. 10, p. 35, 12 2017.

[35] D. Walther, "Using confusion matrices to estimate mutual information between two categorical measurements," *Proceedings - 2013 3rd International Workshop on Pattern Recognition in Neuroimaging, PRNI 2013*, pp. 220–224, 06 2013.

[36] M. McHugh, "Interrater reliability: The kappa statistic," *Biochemia medica : časopis Hrvatskoga društva medicinskih biokemičara / HDMB*, vol. 22, pp. 276–82, 10 2012.

[37] D. Hand, "Measuring Classifier Performance: A Coherent Alternative to the Area Under the ROC Curve," *Machine Learning*, vol. 77, pp. 103–123, 10 2009.

[38] F. Sebastiani, "An Axiomatically Derived Measure for the Evaluation of Classification Algorithms," in *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*, ICTIR '15, p. 11–20, Association for Computing Machinery, 2015.

[39] P. Baldi, S. Brunak, Y. Chauvin, C. Andersen, and H. Nielsen, "Assessing the accuracy of prediction algorithms for classification: An overview," *Bioinformatics (Oxford, England)*, vol. 16, pp. 412–24, 06 2000.

[40] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, p. 6, 2020.

[41] D. Chicco, V. Starovoitov, and G. Jurman, "The Benefits of the Matthews Correlation Coefficient (MCC) Over the Diagnostic Odds Ratio (DOR) in Binary Classification Assessment," *IEEE Access*, vol. 9, pp. 47112–47124, 2021.

[42] D. Chicco, N. Tötsch, and G. Jurman, "The Matthews correlation coefficient (MCC) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two-class confusion matrix evaluation," *BioData Mining*, vol. 14, no. 1, pp. 1–22, 2021.

[43] T. Dennis, "Confusion matrix visualization." https://medium.com/@dtuk81/confusion-matrix-visualization-fc31e3f30fea, 2019.

[44] SalRite, "Demistifying "confusion matrix" confusion." https://towardsdatascience.com/demystifying-confusion-matrix-confusion-9e82201592fd, 2018.

[45] S. Laursen, "The confusion matrix visualized." https://towardsdatascience.com/the-confusion-matrix-visualized-e778584c8834, 2020.

[46] E. Beauxis-Aussalet and L. Hardman, "Simplifying the Visualization of Confusion Matrix," *In Proceedings of the IEEE Symposium on Information Visualization (IEEE InfoVis), in press*, 11 2014.

[47] S. Mazzanti, "Interactive visualization of the outcome of any binary classifier." Interactive Visualization of the Outcome of Any Binary Classifierhttps://towardsdatascience.com/interactive-visualization-of-binary-classification-in-5-lines-of-python-9c1f627ded8, 2020.

[48] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau, "Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 8, pp. 2674–2693, 2019.

[49] S. Chung, S. Suh, C. Park, K. Kang, J. Choo, and B. C. Kwon, "ReVACNN:Steering convolutional neural network via real-time visual analytics," *Future of Interactive Learning Machines Workshop at the 30th Annual Conference on Neural Information Processing Systems (NIPS)*, 2016.

[50] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards Better Analysis of Deep Convolutional Neural Networks," *CoRR*, vol. abs/1604.07043, 2016.

[51] D. Smilkov, N. Thorat, C. Nicholson, E. Reif, F. B. Viégas, and M. Wattenberg, "Embedding Projector: Interactive Visualization and Interpretation of Embeddings," *ArXiv*, vol. abs/1611.05469, 2016.

[52] J. Yosinski, J. Clune, A. M. Nguyen, T. J. Fuchs, and H. Lipson, "Understanding Neural Networks Through Deep Visualization," *CoRR*, vol. abs/1506.06579, 2015.

[53] M. Kahng, P. Y. Andrews, A. Kalro, and D. H. Chau, "Activis: Visual exploration of industry-scale deep neural network models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 88–97, 2018.

[54] M. Ennemoser, J. Kepler, P. Ruch, H. Stitz, H. Strobelt, and M. Streit, "Confusion-Flow: Visualizing Neural Network Confusion Across Epochs," *IEEE Transactions on visualization and computer graphics*, 2018.

[55] F. Casati, P.-A. Noël, and J. Yang, "On the Value of ML Models," *Workshop on Human and Machine Decisions at NeurIPS 2021 (WHMD 2021)*, 12 2021.

[56] B. Sayin, J. Yang, A. Passerini, and F. Casati, "The Science of Rejection: A Research Area for Human Computation," *Proceedings of The 9th AAAI Conference on Human Computation and Crowdsourcing (HCOMP 2021)*, 11 2021.

List of Figures

1.1 Context: opportunities for visualizations in ML 6

2.1 First approach: heatmap of the baseline confusion matrix 26
2.2 First approach: addition of labels and percentages 26
2.3 First approach: normalization 27
2.4 First approach: compact visualization 27
2.5 Second approach: composition with a number of decorators 28
2.6 Second approach - First graphical element: bars for the pairwise ratio and the consequent visualisation of the performance metrics 29
2.7 Second approach - Second element: outside bars placement to display the prevalence in the among the actual records as well as the predictions 30

2.8 Second approach - Third element: squares for performance metrics computed with more than two entries . . . 30
2.9 Second approach - Third element: further explanation of the squares linking the interpretation with the graphical choices . . . 31
2.10 Second approach - Third element: added value of displaying both metrics . 31
2.11 Second approach - Fourth element: DOR analysis . . . 32
2.12 Third approach: overall design . . . 34
2.13 Third approach: composition of one single stacked bar chart . . . 34
2.14 Third approach: breakdown of the graphical composition . . . 35
2.15 Fourth approach: single graphical and interactive visualisation . . . 36
2.16 Actually positive elements . . . 37
2.17 Fourth approach: predicted positive elements . . . 38
2.18 Fourth approach: true positives (TP) . . . 38
2.19 Fourth approach: false positives (FP) . . . 39
2.20 Fourth approach: false negatives (FN) . . . 39
2.21 Fourth approach: true negatives (TN) . . . 40
2.22 Fourth approach: legend and performance metrics (TN) . . . 40
2.23 Fourth approach: different choices of thresholds synchronously modify the plot 41

3.1 The Confusion Tetrahedron . . . 47
3.2 Implementation: workflow diagram . . . 50
3.3 Analysis of the missing values . . . 56
3.4 Heatmap of the missing values per attribute . . . 57
3.5 Missing values per attribute distinguished per class . . . 57
3.6 Boxplots of the numerical features . . . 59
3.7 Boxplot of all numerical features vs target class . . . 60
3.8 Grouped bar chart of categorical vs target variable . . . 61
3.9 Need for normalization of the numerical variables . . . 61
3.10 Countplot of the target variable in the final dataset . . . 63
3.11 LR - classical visualisation approach . . . 64
3.12 NB - classical visualisation approach . . . 65
3.13 SVM - classical visualisation approach . . . 66
3.14 Choice of the K for KNN . . . 66
3.15 KNN - classical visualisation approach . . . 67
3.16 DT - classical visualisation approach . . . 68
3.17 Novel approach to the evaluation of the tested model with the Confusion Tetrahedron. The three axes represent TP, FP, TN and the visualisation can be moved according to the preffered perspective . . . 70
3.18 CT - comparison of models' performances . . . 71

4.1 Confusion Tetrahedron coloured according to the chosen performance metric, MCC. The three axes represent TP, FP, TN 76
4.2 Confusion Tetrahedron coloured according to the chosen performance metric, Accuracy. The three axes represent TP, FP, TN 77
4.3 F_1 score 77
4.4 Confusion Tetrahedron coloured according to the chosen performance metric, F_1. The three axes represent TP, FP, TN 77
4.5 The 2D Cartesian plane representation of sections of the Confusion Tetrahedron identified by different splitting equations. Points in planes not orthogonal to the axes were projected using the barycentric coordinates. The barycentric coordinates of a point can be interpreted as masses placed at the vertices of the simplex, such that the point is the center of mass (or barycenter) of these masses. 78
4.6 Behaviour of a model for all levels of the thresholds. Case of a good performance: comparison among the picture provide by the bi dimensional plane with the ROC curve and the three dimensional environment of the CT. The three axes represent TP, FP, TN 81
4.7 Behaviour of a model for all levels of the thresholds. Case of an intermediate performance: comparison among the picture provide by the bi dimensional plane with the ROC curve and the three dimensional environment of the CT. The three axes represent TP, FP, TN 81
4.8 Behaviour of a model for all levels of the thresholds. Case of a poor performance: comparison among the picture provide by the bi dimensional plane with the ROC curve and the three dimensional environment of the CT. The three axes represent TP, FP, TN 82
4.9 Summary of the possible views. The three axes in (a),(b),(c) represent TP, FP, TN 82
4.10 Representation of the learning process of a Neural Network 83
4.11 Architecture of the Neural Network built for the classification of the chronic kidney disease. The architecture consists of an input layer, 2 hidden layers and one output layer. 85
4.12 Neural network: accuracy throughout the epochs 86
4.13 Neural network - Results for each epoch (orange points) and the associated trajectory (dashed grey line). The three axes represent TP, FP, TN. 87
4.14 Neural network: Confusion Tetrahedron result in epoch = 5. The three axes represent TP, FP, TN. 87
4.15 Neural network - Confusion Tetrahedron result in epoch=23. The three axes represent TP, FP, TN. 88

4.16 Neural network - Confusion Tetrahedron result in epoch=50.The three axes represent TP, FP, TN. 88
4.17 Neural network - Confusion Tetrahedron result in epoch = 87.The three axes represent TP, FP, TN. 89
4.18 Neural network - Confusion Tetrahedron result in epoch=100.The three axes represent TP, FP, TN. 89
4.19 User interaction 90
4.20 Comparison of the results for the two alternative resampling strategies . . . 92
4.21 Comparison of the results for the two alternative resampling strategies: zoom on the resulting models and distribution 93

List of Tables

3.1 LR - confusion matrix 63
3.2 LR - performance metrics 64
3.3 NB - confusion matrix 64
3.4 NB - performance metrics 65
3.5 SVM - confusion matrix 65
3.6 SVM - performance metrics 65
3.7 KNN - confusion matrix 66
3.8 KNN - performance metrics 67
3.9 DT - confusion matrix 67
3.10 DT - performance metrics 67
3.11 Summary of the performance metrics 68
3.12 CT: Fictitious confusion matrix to show result of poor performance 70

A.1 List of the attributes in the dataset with the associated label and type . . . 114

Acronyms

DS	Data Science
ML	Machine Learning
BI	Business Intelligence
ROC	Receiver Operating Characteristic
MCC	Matthew Correlation Coefficient
NN	Neural Networks
DL	Deep Learning
CM	Confusion Matrix
nCM	Normalized Confusion Matrix
CT	Confusion Tetrahedron
CPU	Central Processing Unit
VIF	Variance Inflation Factor
ETL	Extract, Transform, Load
GIGO	Garbage In, Garbage Out
ReLU	Rectified Linear Unit
CV	Cross Validation
POC	Proof of Concept
LR	Logistic Regression
NB	Naive Bayes
SVM	Support Vector Machines

RF Random Forest

DT Decision Tree

KNN K-Nearest Neighbors

MMH Maximum Marginal Hyperplane

3D three dimensional

2D bi dimensional

TP True Positives

nTP Normalized True Positives

FP False Positives

nFP Normalized False Positives

FN False Negatives

TN True Negatives

nTN Normalized True Negatives

TPR True Positive Rate

FNR False Negative Rate

TNR True Negative Rate

PPV Positive Predictive Value

PCR Precision

PT Prevalence Threshold

FDR False Discovery Rate

FPR Flse Positive Rate

NPV Negative predictive value

FOR False Omission Rate

FNR False Negative Rate

MR Miss Rate

REC Recall

SENS Sensitivity

SPEC Specificity

HR Hit Rate

CSI Critical Success Index

TS Threat Score

BA Balanced Accuracy

BM Bookmaker Informedness

MK Markedness

DOR Diagnostic Odd Ratio

MCC Matthew's Correlation Coefficient

ACC Accuracy

SNS Sensitivity

SPEC Specificity

GS Gilbert Skill Score

HR Hit Rate

Appendices

Attachment A Materials

The *raw data* used for the implementation and the first three applications can be found in the UCI repository at the following link. For the fourth application a more unbalanced dataset has been used to better target the problem tackled by the visualization. The adopted dataset is a modification (*see* comments in 4.4 of the github repositorygithub repository for details) of the Credit Card Dataset from Kaggle.

The list of necessary *libraries and packages* to be pip-installed in the environment can be found on the github repository in the requirement.txt file.

What follows is a table comprising all features of the dataset that have been used in the classification models.

List of attributes			
Attribute	Label	Type	Explanation
Age	*age*	num	Age of the patients in years
Blood Pressure	*bp*	num	Blood pressure of the patients record in mm/Hg
Specific gravity	*sg*	nom	Categorical variable with five possible values being: 1.005, 1.010, 1.015, 1.020, 1.025
Albumin	*al*	nom	Categorical variable with five possible values being: 0,1,2,3,4,5
Sugar	*su*	nom	Categorical variable with five possible values being: 0,1,2,3,4,5
Red Blood Cells	*rbc*	nom	Categorical variable with two possible levels being normal or abnormal appearance of the red blood cells
Pus cells	*pc*	nom	Categorical variable with two possible levels being normal or abnormal pus cells
Pus cell clumps	*pcc*	nom	Categorical variable indicating the presence or absence of pus cells indicating that an infection has been present for a while
Bacteria	*ba*	nom	Categorical variable recording the presence or absence of bacteria in the urine
Blood Glucose Random	*bgr*	num	mgs/dl
Blood urea	*bu*	num	mgs/dl
Serum creatinine	*sc*	num	mEq/L
Sodium	*sod*	num	mEq/L
Potassium	*pot*	num	mEq/L
Hemoglobin	*hemo*	num	gms
Packed Cell Volume	*check*	num	check
White Blood Cell Count	*wc*	num	cells/cmm
Red Blood Cell Count	*rbcc*	num	millions/cmm
Hypertension	*htn*	nom	Categorical variable with two values indicating the presence (*yes*) or absence (*no*) of hypertension in the patient
Diabetes mellitus	*dm*	nom	Categorical variable with two values indicating if the patient is affected (*yes*) or not (*no*) by diabetes of type mellitus
Coronary Artery Disease	*cad*	nom	yes/nor
Appetite	*appet*)	nom	Categorical variable quantifying the appetite of the patient discretizing the result in two levels being good or poor
Pedal edema	*pe*	nom	Categorical variable with only two values indicating the fact that the patient does present or does not predent edema to the feet
Anemia	*ane*	nom	Categorical variable with only two values indicating the fact that the patient is or is not affected by anemia
Class	*class*	nom	Binary output indicating the presence (*'ckd'*)or absence of the disease (*'notckd*)

Table A.1: List of the attributes in the dataset with the associated label and type

Attachment B Code

The *code* is not reported in the present work but it can found in the public github repository available at this link. The whole analysis has been split in different stages: the data cleaning and preparation, the implementation of the models with the first application, the second application on the ROC curves, the third application on the temporal analysis of training epochs and the fourth for the visualization of the results of stratified vs random resampling. The repository is comprehensive of the *data folder* containing the dataset as they result from the data preparation and preprocessing stage. All the installed libraries and packages necessary to execute the analysis are contained in the file *requirements.txt* that can be pip-installed in one's own environment.