UNIVERSITY OF CAGLIARI

MASTER DEGREE IN COMPUTER ENGINEERING, CYBERSECURITY AND ARTIFICIAL INTELLIGENCE

Spam Mail Detection Project

COURSE
Artificial Intelligence

STUDENTS
Christian Scano (70/90/00346)
Laura Porcu (70/90/00326)

A.Y. 2023/2024

Contents

Chapter 1

Introduction

The rapid growth of digital communications has raised a crucial challenge in email management: the widespread spam problem in emails. Using machine learning techniques, our goal is to provide a robust and efficient solution that can automatically distinguish between spam and legitimate messages.

The dataset provided presented different examples of emails correctly classified as legitimate (HAM) and not legitimate (SPAM). In order to verify the performance of our system and ensure accurate classification of the dataset, we compared the performance of two different classifiers.

The first one is the **Decision Tree**, which uses a tree structure to make classification decisions. The second one is the **Logistic Regression**, a linear classifier that models the probability of class membership based on independent variables.

The results reported after the training phase, show that our approach provides high accuracy and robustness in performing classification. The final part also includes limitations related to the implementation of our project.

Chapter 2

Theoretical aspects

2.1 Decision Tree algorithm

A decision tree is a non-parametric supervised learning algorithm used for classification and regression. It is composed of a hierarchical tree structure, which consists of a root node, branches, internal nodes, and leaf nodes.

A decision tree starts with a root node, which has no incoming branches. Outgoing branches from the root node feed internal nodes, also known as decision nodes. Based on the available functionality, both types of nodes conduct evaluations to form homogeneous subsets, which are represented by leaf nodes or terminal nodes.

This subdivision process is then repeated recursively from top to bottom until all or most of the records have been classified into specific class labels.

The top-down induction process of a Decision Tree is an example of a Greedy algorithm. It is a heuristic approach to problem-solving that makes local optimal choices at each step. During the construction of the tree, the best available choice is made at each pass based on the feature that provides the most meaningful split in the data. This choice is made locally, without considering the overall impact on

the entire tree structure.

2.2 Logistic Regression

Logistic regression is a supervised ML algorithm used primarily for binary classification tasks. Unlike linear regression, which predicts continuous values, logistic regression predicts the probability that an input belongs to a particular category or class. It accomplishes this by applying a logistic (or sigmoid) function to a linear combination of input features, mapping the result to the range [0, 1]. This enables the model to output probabilities, making it interpretable and suitable for decision-making scenarios. During training, logistic regression adjusts its parameters to minimize a cost function, typically the log loss, optimizing its ability to discriminate between classes. Despite its simplicity, logistic regression remains a powerful tool in various domains due to its robustness, ease of interpretation, and efficiency, particularly with small to moderately-sized datasets. However, it's essential to recognize its assumptions, such as linearity in feature relationships, and its limitation to binary classification tasks, although it can be extended for multiclass problems with appropriate modifications.

Sigmoid Function

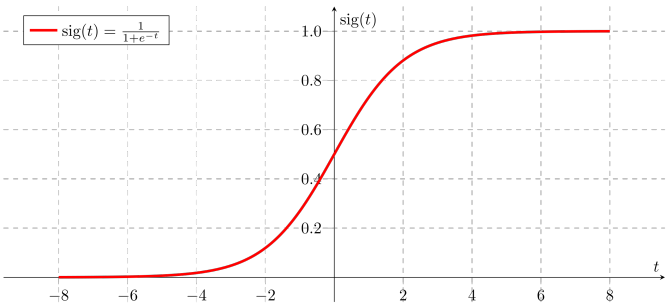

Figure 2.1: Sigmoid function

Chapter 3

Experimental Setup

Our project focused on developing a mail spam detector using two different approaches - Decision Tree and Logistic Regression. While the primary objective was to implement a functional spam detection system, we also aimed to compare the performance of the two models in terms of misclassification rate.

This comparison provided us with valuable insights into the strengths and weaknesses of each approach and helped us understand which model is more effective in identifying spam emails.

3.1 Dataset

The dataset we have used is the "spam-mails-dataset" composed of 5170 samples. It is divided into three columns:

- **label**: Indicates whether emails are classified as Spam or Ham (non-spam);
- **text**: Contains the actual content of the emails;
- **label_num**: A numerical representation where 1 indicates Spam and 0 indicates Ham.

Globally the label 'ham' accounts for 71% of the dataset, while 'spam' makes up the remaining 29%. During the k-fold cross-validation, the training fold was composed of 4136 samples and the testing fold was composed of the remaining 1034 samples.

3.2 Data processing

Before feeding data into the classifier, a **preprocessing step** was necessary. A dedicated preprocessing function was developed to refine the text using regular expressions. This function included several essential steps.

Initially, it involved removing the *Subject* term from the beginning of each e-mail, which was considered redundant for classification purposes. Next, *numerical values*, *punctuation marks*, and *hyperlinks* were removed from the text to mitigate irrelevant information. In addition, another step involved the removal of stopwords such as *this* and *is*, which had no discriminatory value in distinguishing between legitimate and spam emails.

Then, for each word tokenized from the text, *stemming* was employed to reduce the words to their root, thus facilitating analysis.

Pre-processing ensured that the textual data were standardized and optimized, improving the effectiveness of the classification.

3.3 Hyperparameters

We used the two implementations proposed in Scikit-Learn library:

- DecisionTreeClassifier
- LogisticRegression

The hyperparameters of the *DecisionTreeClassifier* are mainly two:

- The **criterion of splitting**, that was set to *log loss*. This parameter measures the quality of a split;
- The **max depth**, that was set to 70. The motivation for this tuning is that we saw better performances in terms of the accuracy of classification.

Regards to the *LogisticRegression*, we used the default hyperparameters set by the Scikit-Learn library:

- The **penalty**, that was set to L2 (Ridge). It penalizes the squared magnitudes of the coefficients, encouraging smaller coefficient values without forcing them to zero;
- The **maximum number of iterations** for the solver to converge. It was set to 100;
- The **regularization parameter** (C), that was set to 1.0. This parameter helps to prevent overfitting and promote smoother functions.

3.4 Evaluation criteria

For evaluating these two models we have taken into account the misclassification rates. We have computed the misclassification rates as:

$$\text{Misclassication rate} = \frac{\text{Number of misclassified samples}}{\text{Total number of samples}}$$

The number of misclassified samples is computed as the sum of the false positive and false negative extracted from the confusion matrix computed by the apposite function proposed by the Scikit-Learn library.

To evaluate the models we have also plotted the ROC curve: a ROC (Receiver Operating Characteristic) curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The TPR is defined as:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

where TP is the number of true positive cases and FN is the number of false negative cases.

The FPR is defined as:

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}}$$

where FP is the number of false positive cases and TN is the number of true negative cases.

Chapter 4

Experimental results

4.1 Misclassification rates and ROCs

We used roughly 1034 test samples to see the classification performance of the used models. As far as the Decision Tree is concerned, we can see that the misclassification rate does not show a notable decrease as the number of features increases compared to the Logistic Regression. Instead, seeing at the Logistic Regression performances we can see that is more accurate than the Decision Tree at classifying for every values of features taken into consideration (N=100, 500, 1000).

The figures below shown the expressed misclassification rate as a percentage for both the Decision Tree (4.1) and the Logistic Regression (4.2):
In addition to the misclassification rate curves, we also plotted the ROC curves for each value of features considered for both models. We decided to plot the ROC curves to get a more complete overview of the performance of the models taken into consideration terms of sensitivity and specificity.

The figures below are the ROC Curves for each number of features taken into account and for the Decision Tree (4.3) and Logistic Regression (4.4)

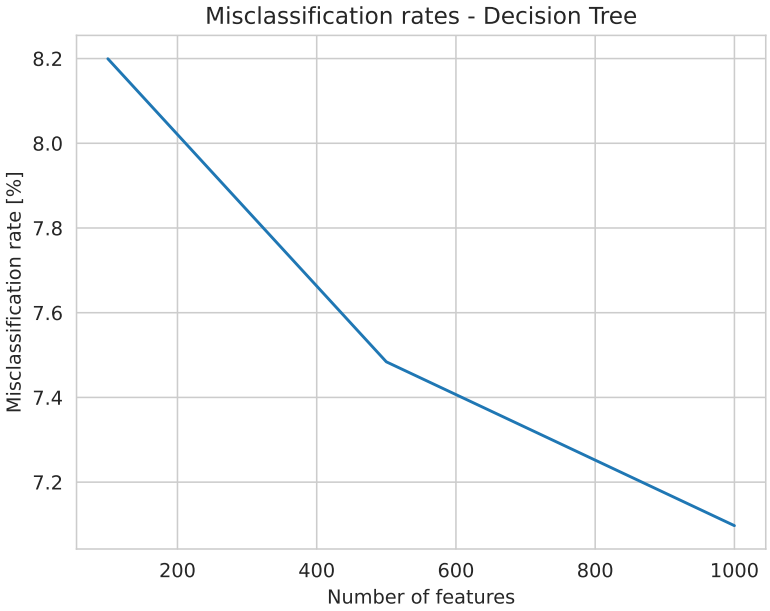

Figure 4.1: Misclassification rate for the Decision Tree

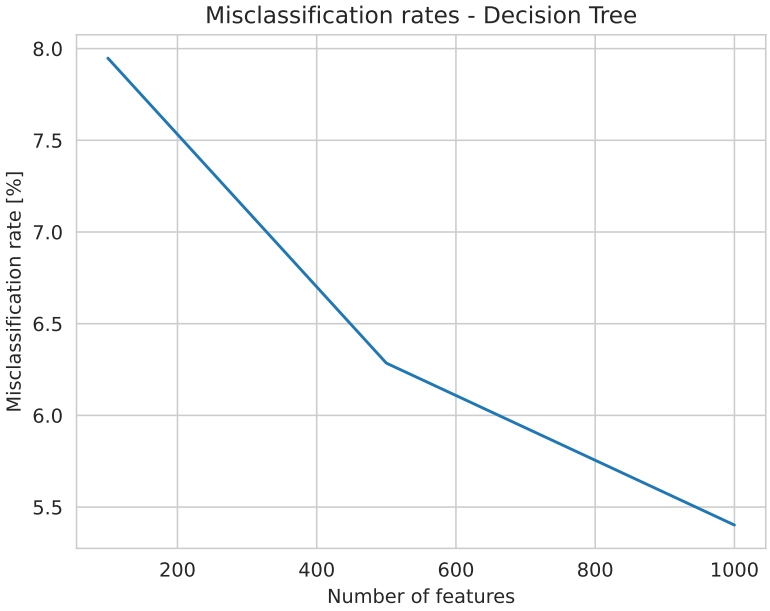

Figure 4.2: Misclassification rate for the Logistic Regression

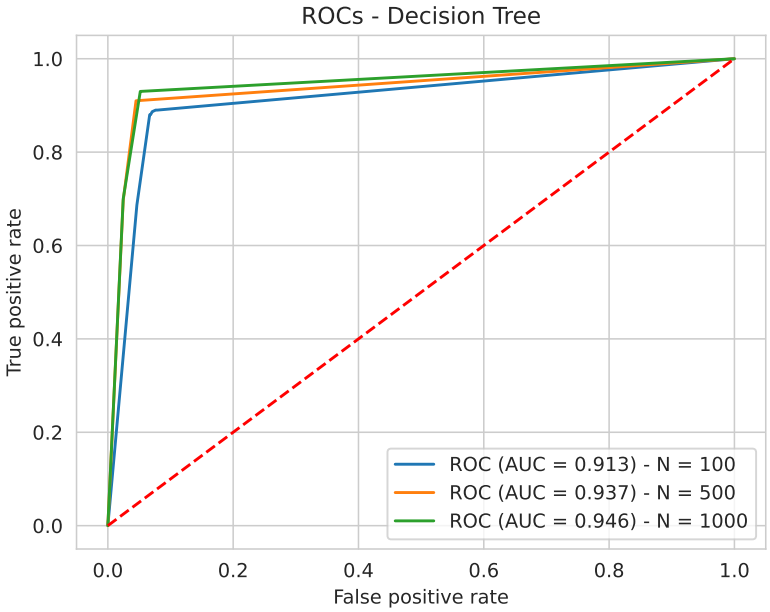

Figure 4.3: ROC curvers for the Decision Tree

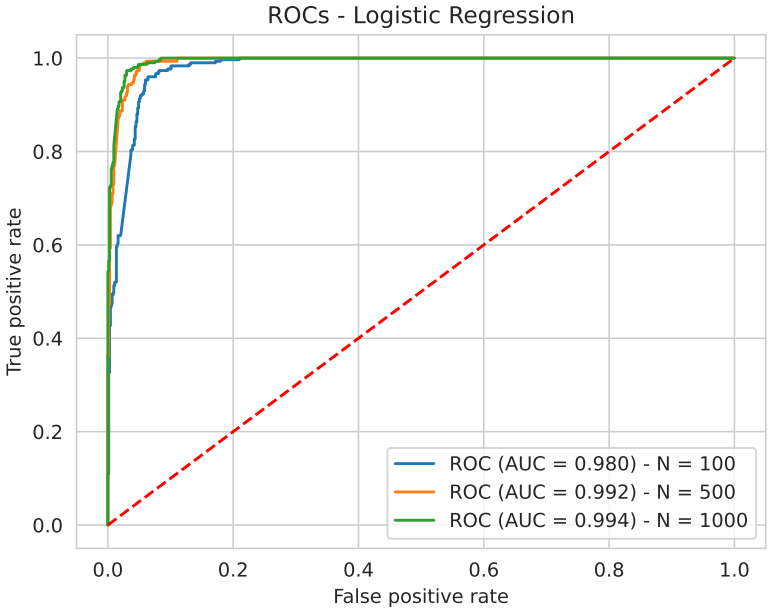

Figure 4.4: ROC curves for the Logistic Regression

4.2 Limitation

One limitation of the project is the relatively small dataset, comprising approximately 5000 samples. This could lead to overfitting of the classification model, as it might memorize specific features of the training data rather than generalize underlying patterns. However, in future works, employing a larger dataset could potentially improve performance.