21 novembre 2022

Auteur :
SÁNCHEZ LAGUARDIA Manuel
CABEZA GALLUCCI Manuel
CHEKER BURIHAN André

Tuteurs :

Compte Rendu Projet GIT

TAF MCE - UE A : Intro ML

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Table des matières

1 Introduction

1.1 Project Presentation

In this lab, the main objectives presented are : to develop good programming practices, learn to use standard development tools, get used to collaborative work ; everything while working on Machine-Leaning Classification problems using two different datasets.

The link to the GitHub used for version control and working collectively is the following :
https ://github.com/manuelcgallucci/BinaryClassificationWorkflow

2 Data

The data sets used for this experiment were the two following :

1. Banknote Authentication Dataset : https ://archive.ics. uci.edu/ml/datasets/banknote+authentication
2. Chronic Kidney Disease Dataset : https ://www.kaggle.com/mansoordaku/ckdisease

The goal was to have a unique code that processed both datasets just by changing the input dataset. The input dataset was defined by the data and multiple parameters such as the name of features, the size of the cross validation, the test set size and the model to train the data. This is done with a dictionary as it is shown in the following figure.

```
BankNotes = {
    "name": "BankNotes",
    "data_path": "./data/data_banknote_authentication.txt",
    "ftr_names": ["variance of WTI", "skewness of WTI", "curtosis of WTI", "entropy"],
    "out_path": "./output/bank_notes/",
    "model": RandomForestClassifier(n_estimators=10, max_depth=6),
    "cross_validation": 10,
    "test_size": 0.3,
    "true_label": None,
    "dummy_cols": None,
}
```

FIGURE 1 – Standard dataset definition.

2.1 Data processing

To process the data, it was imported from a csv to a dataFrame using Pandas. We asume that the dataset is for binary classification and thus only 2 labels exist. After importing the following steps were made to clean the data :

1. Import the data.
2. Drop labels column and separate it from the rest of the dataset. If they are strings (such as "yes" or "no"), turn them to integers 0 or 1.
3. Remove rows with no labels.
4. Treat dummy columns (columns with strings) :
   (a) If any rows have an empty value, they are replaced with the most common string by using the function $pd.mode$.
   (b) Then $pd.get\_dummies$ is used to change its values to integers : 0 and 1.
5. Dealing with missing values : Fill missing values replaced by the mean or the median of the column.
6. Normalize data : Can be done by subtracting the mean and dividing by the standard deviation, or by mapping it to the interval [0,1].
7. Shuffle the data.

3 Models proposed

Supervised learning methods were used and compared on both datasets. All models from the "sklearn" library that have a fit() and predict() method can be used on the workflow. For the bank notes dataset a 10-fold cross validation was used and for the kidney disease a 5-fold one. In all cases the validation set proportion was 30%. [1]

Model	Precision	Recall	F1 Score
Logistic Regression	1	0,97	0,955
SVM 'linear'	1	0,99	0,985
SVM 'poly'	1	0,97	0,955
Decision Tree	0,98	0,98	0,941
Random Forest	0,98	0,99	0,956

TABLE 1 – BankNote Dataset results.

Model	Precision	Recall	F1 Score
Logistic Regression	0,99	1	0,985
SVM 'linear'	1	0,99	0,985
SVM 'poly'	1	0,59	0,469
Decision Tree	1	1	1
Random Forest	1	1	1

TABLE 2 – Chronic Kidney Disease Dataset results.

It can be stated that for both datasets, even simple models, such as Logistic Regression, perform very good, with F1 scores close to 1 (which is the ideal value for a perfect classifier). Nevertheless, to achieve better results other more complex models were used, such as random forest or linear support vector machines. These methods have the highest mean f1 score over both datasets ; as seen in Table 1 and Table 2.

The same model (Random Forest) was evaluated with different parameters to see how these parameters impacted performance. The results are shown in the following Tables 3 and 4.

Random Forest : (n_estimators, max_depth)	Precision	Recall	F1 Score
(2,6)	0,96	0,98	0,913
(2,10)	0,92	0,99	0,870
(2,15)	0,93	0,99	0,884
(10,6)	0,98	0,99	0,956
(10,10)	0,99	0,99	0,970
(10,15)	0,98	0,99	0,956

TABLE 3 – Random Forest for the bank notes dataset.

Random Forest : (n_estimators, max_depth)	Precision	Recall	F1 Score
(2,6)	0,94	0,97	0,871
(2,10)	0,95	1	0,926
(2,15)	0,93	1	0,897
(10,6)	1	1	1
(10,10)	1	0,99	0,985
(10,15)	1	1	1

TABLE 4 – Random Forest for the Chronic Kidney Disease dataset.

1. If no parameters are indicated in the model name then the defaults where used.

Increasing the depth significantly increases performance, but up to a certain point. After this threshold is reached, it is a better idea to increase the number of estimators, in order to achieve the best results.

4 Results

The following results are the output of the program when training with the parameters described in Section 3 and using a Random Forest model with 10 estimators and a maximum depth of 10.

4.1 Banknote Authentication Dataset

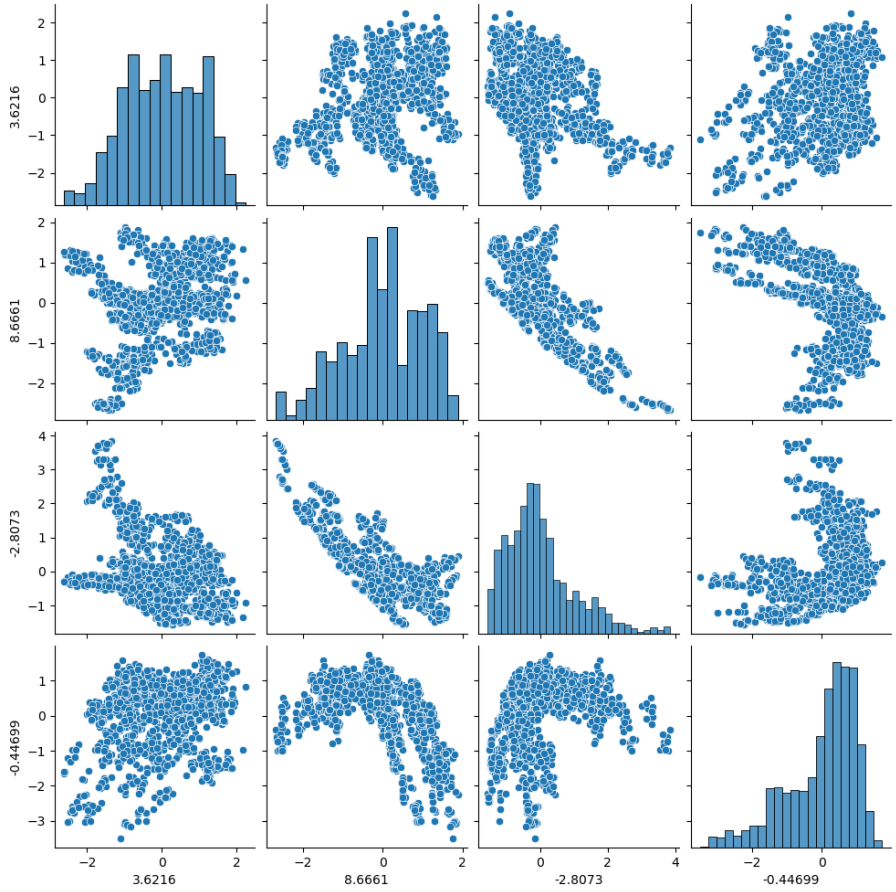

FIGURE 2 – Pairplot for the features of the dataset.

The Pairplot is done when the number of features is small. This allows to see the correlation between every pair of features as well as the distribution of each one. For example, in Figure 2, features 2 and 3 have a negative correlation and feature 1 has values that are more or less uniform in the interval $(-1.5, 1.5)$.

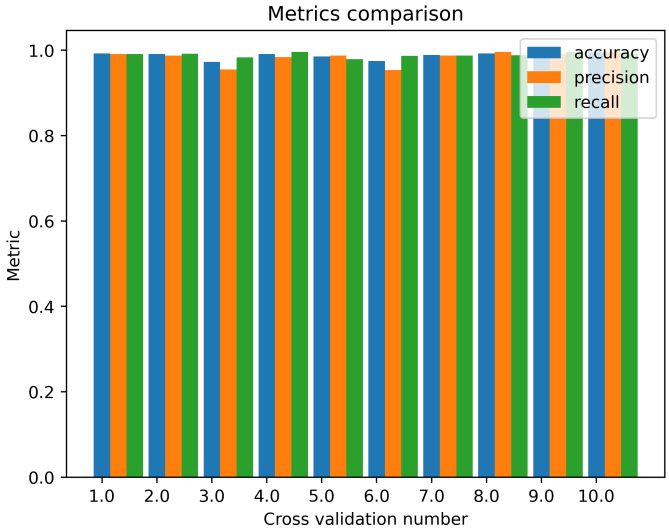

FIGURE 3 – Metrics for all cross validations using Random Forest.

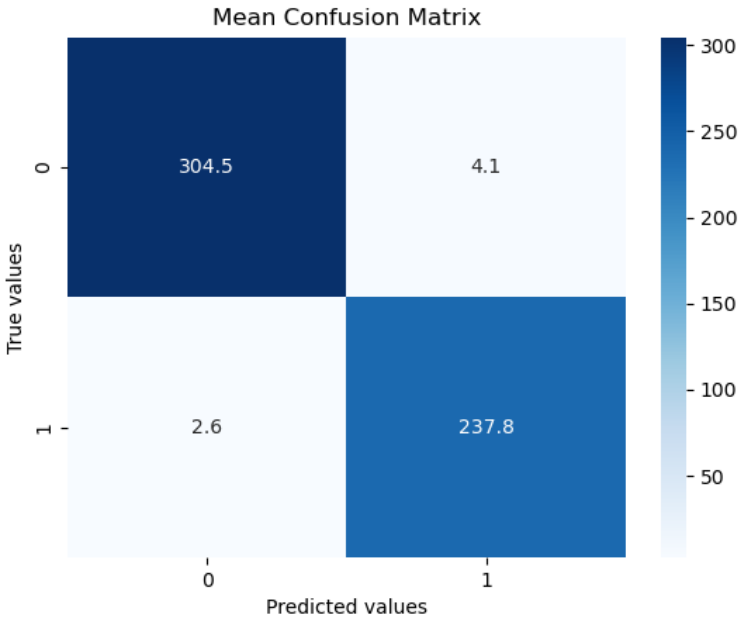

FIGURE 4 – Average confusion matrix over the validation data.

It is clear that all cross validation instances for the random forest model in this dataset perform really well. In the mean confusion matrix the results are very good, there are less than 3 false negatives and 4 false positives for the 549 entries of the dataset.

4.2 Chronic Kidney Disease Dataset

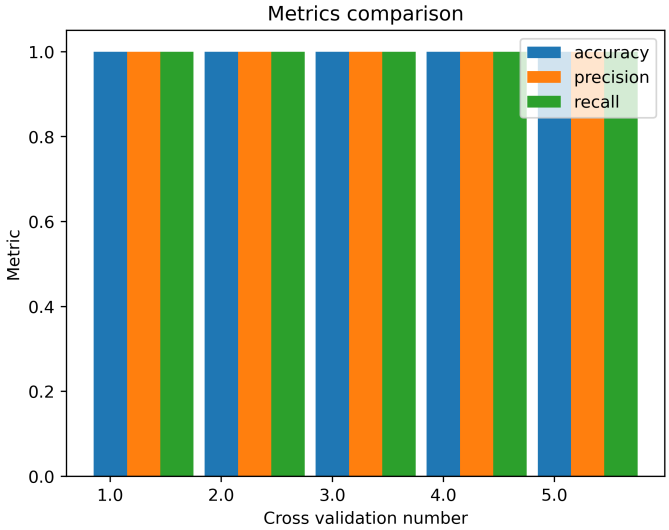

FIGURE 5 – Metrics for all cross validations using Random Forest.

Similarly to the previous dataset, all cross validation instances for the random forest model in this dataset perform really well.

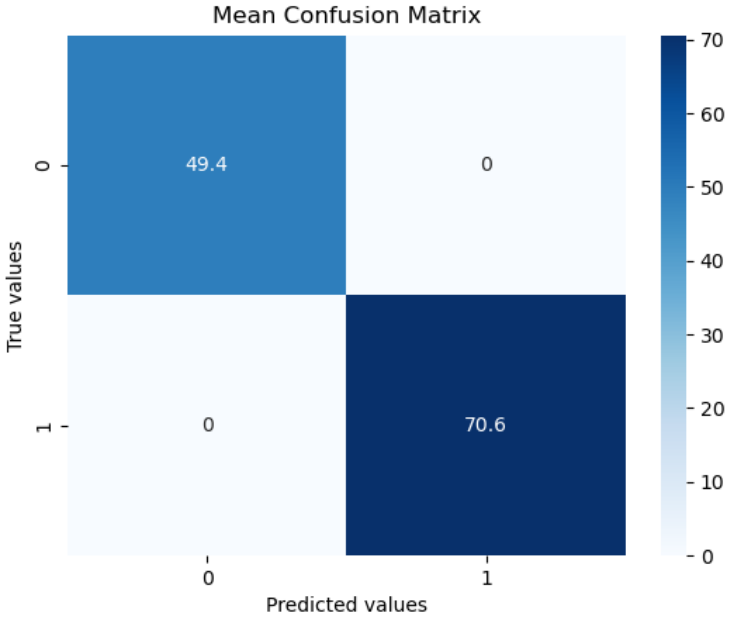

FIGURE 6 – Average confusion matrix over the validation data.

In this case, the results are even better, there are no false positives or false negatives in any cross validation run. This means that the mean confusion matrix has made no mistaken predictions for the whole datasets, 5-fold times.

5 Unit testing

An unitary test was made for the 'import_data()' function which specifically tests for the handling missing values (NaN) in the input data.

It takes the possible values for the 'replace_nan' argument and tests the behavior of the function, the expected result is read from a separate CSV file and compered to the output.

Unit testing 'import_data()' assured that all edge cases in the result when an element is missing are taken care of. This are :

1. If no label exists, then the row is deleted
2. If a value is missing from a numeric column it is replaced by the mean, median or deleted. (According to user input)
3. If a value is missing from a column with string objects inside then it is replaced by the mode or deleted. (According to user input)

6 Good Programming Techniques

In the following section, various programming techniques will be assessed. The list is not exhaustive, but an order of importance was taken into account to choose the most relevant techniques (in our opinion).

6.1 Documentation

A well explained documentation might be the most important technique to consider. Commenting your code is the best way to keep track of changes and to explain why you chose to write something in a specific way. This can be very useful at the time of debugging or finding weaknesses in our code. It not only helps to update and upgrade the code in the future, but also is key when rereading an old code.

6.2 Peer review

As the saying goes : "Two heads think better than one". This is very true when programming. Having someone else review our code can be of paramount importance to find errors or improvements. Sometimes, when we work for a very long time on a specific code, we tend forget to take into account edge cases, this is why having someone else with a fresh new perspective can be extremely helpful to tackle this kind of problems. This goes in hand with the next point.

6.3 Share your code and data openly

If you are working on a project that will end up being published, the best thing you can do is show others how you did it and what data you used to do it. This will allow for better peer reviewing, which is fundamental in the scientific community. It will also be beneficial for others that would like to delve deeper into your publication and maybe propose a new approach, perspective, or simply replicate your experiments, thus making the scientific community grow.

Of course, if what you are working on is confidential and cannot be shared openly, it is better to keep it to yourself, but try to share as much information as possible.

6.4 Version Control and Testing

Another fundamental aspect regarding programming techniques is version control. In this specific project, GIT versioning was used to keep track of changes made by the group individually and collectively. This allows for easy sharing and deploying, while being able to rewind to the exact point in time where an action was taken.

Also, it's important to keep track the code that is being added to the codebase is good and works as it is supposed to. That is why unit testing is another aspect to consider when writing proper code, this way we not only verify that the code compiles (when working in other languages, as python code doesn't compile) but also that it works as it should. Taking into account edge cases is also easier when doing unit testing as a single test can be used

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

IMT Atlantique Bretagne - Pays de la Loire - www.imt-atlantique.fr

Campus de Brest
Technopôle Brest-Iroise
CS 83818
29238 Brest Cedex 03
T +33 (0)2 29 00 11 11
F +33 (0)2 29 00 10 00

Campus de Nantes
4, rue Alfred Kastler - La Chantrerie
CS 20722
44307 Nantes Cedex 03
T +33 (0)2 51 85 81 00
F +33 (0)2 51 85 81 99

Campus de Rennes
2, rue de la Châtaigneraie
CS 17607
35576 Cesson Sévigné Cedex
T +33 (0)2 99 12 70 00
F +33 (0)2 99 12 70 08