Khalid Moro, Martina Pastorino, Meritxell Riera i Marín, Gustavo Rodrigues Dos Reis

Development project in Machine Learning

TAF MCE

December 2019

Contents

1. Introduction

Machine Learning (ML) is a field of artificial intelligence (AI) that provides the systems with the ability to automatically learn and improve from experience without being explicitly programmed. ML computer programs focus on a way to access data and use them to learn for themselves. The process of learning begins with observing data, in order to look for patterns and make better decisions in the future based on the examples provided. The main goal is to allow computers to learn automatically without human intervention or assistance. Learning algorithms are generally based on the optimization of a performance criterion that measures how well the selected "machine" matches the available data [1].

The objective of this project is to study different machine learning algorithms dedicated to the *classification* of two different datasets: *Banknote Authentication* and *Chronic Kidney Disease*. However, the main goal is not the programming part itself, but the analysis of their behaviour in each dataset and the extraction of different conclusions on their performances.

Since both datasets are examples of *supervised learning*, where the *instances* are given with known *labels* (i.e., the corresponding correct outputs), the *clustering algorithm*, which is used for *unsupervised learning*, was discarded. The goal of *supervised learning* is to build a concise model of the distribution of class labels in terms of predictor features. The resulting classifier is then used to assign class labels to the testing instances where the values of the predictor features are known, but the value of the class label is unknown [2]. In particular, this work is concerned with *classification* problems, in which the output of instances admits only *discrete* values. It was decided to study *Principal Component Analysis (PCA)*, *Support Vector Machine (SVM)*, *Neural Networks* and *Decision Trees*. These four different algorithms represent the most popular ways of classifying data in supervised machine learning [3].

As stated before, in this project two datasets need to be analysed. The *Banknote Authentication* dataset consists of 1372 instances: 762 are *non authentic notes* and 610 are *authentic*. The features are five and represents the *variance*, the *skewness*, the *curtosis*, the *entropy* and the *class* (either *authentic notes* or *not*) [4]. In Fig. 1.1, the *boxplots* of these parameters are reported: only for the case of *curtosis* and *entropy* there are some outliers. This fact can be observed as well in Fig. 1.2, which shows the *distributions* of the considered parameters.

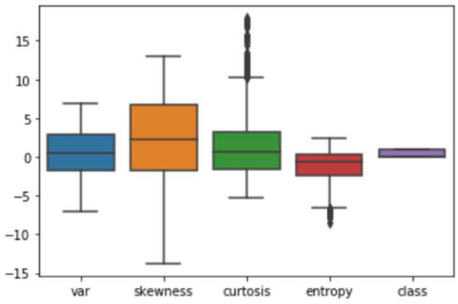

Fig. 1.1 - Boxplot of *banknote authentication* data.

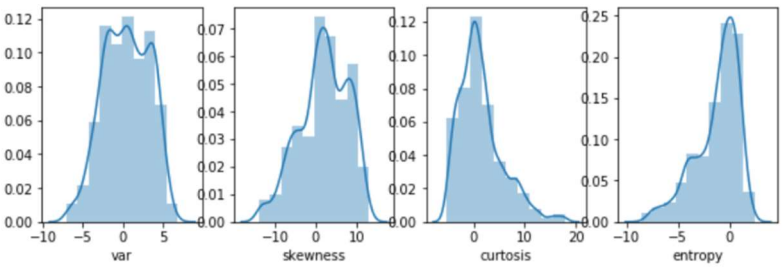

Fig. 1.2 - Distribution plot of *banknote authentication* dataset.

The *Chronic Kidney Disease* dataset consists of 400 instances and 25 attributes. It is a problem of *binary classification* which consists in detecting either the *presence* or the *absence* of kidney disease [5]. In this dataset, the *categorical variables* will be ignored and only the *numerical variables* will be used for classification. In Fig. 1.4, the boxplots of the dataset are reported, showing the presence of several outliers (mostly for what it concerns *"blood glucose"* and *"blood urea"*), highlighting the variability of the data.

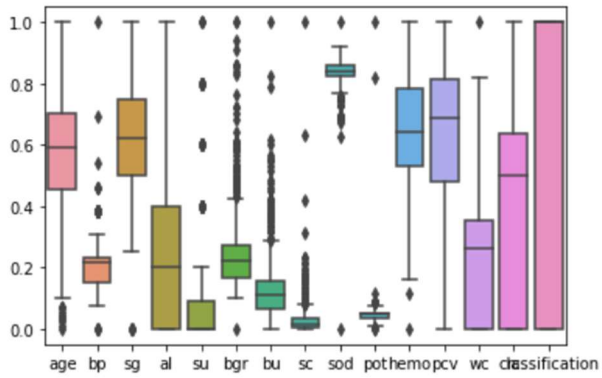

Fig. 1.4 – Boxplot visualization of *chronic kidney disease* data.

This dataset is really meaningful because early detection and characterization are considered to be critical factors in the management and control of *chronic kidney disease.* Consequently, the use of efficient *data mining techniques* is shown to reveal and extract hidden information from clinical and laboratory patient data, which can be helpful to assist physicians in maximizing accuracy for the identification of the disease severity stage [6].

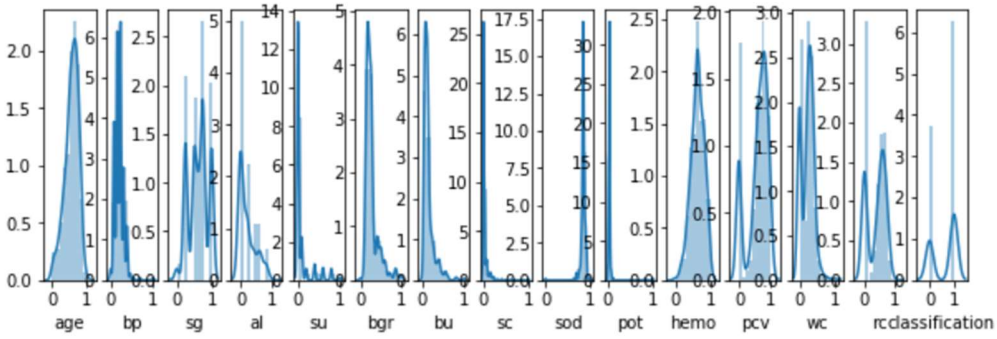

Fig. 1.5 - Distribution plot of *chronic kidney disease* dataset.

Fig. 8.1 in the Appendix, which reports the *pairplots* of the various features, is useful to observe the correlation between the variables of the chronic kidney disease dataset. For example, a high correlation between *hemo* (*hemoglobin*) and *pcv* (*packed cell volume*) can be noticed. The same happens also for *pcv* and *rc* (red blood cell count), as well as for *hemo* and *rc* (which intuitively makes sense).

Good programming practices are essential when a task should be performed collectively, because they allow a certain uniformity of treatment between the various parts. This can be very useful whenever it is necessary to debug a code, or even to understand what has been done.

A first good practice is to use a *proper naming convention*. For instance, one can start with lowercase letters when declaring variables and uppercase when declaring functions, to help in the readability of code. Following a verb-object structure for naming methods is another very important practice for the same purpose.

Moreover, writing a *good documentation* and avoiding the repetition of obvious information in the comments of the code are essential for the "maintainability" of a project and for later use.

Another good practice is to use a *constant indentation* throughout the code.

Not *repeating yourself* (known popularly as DRY principle), writing several similar lines, to adapt frequent routines in new methods is another important point to be considered while trying to maintain conciseness and simplicity. Finally, the last general programming practice that implies all the others is *code refactoring* (i.e., modifying existing code without changing its external behaviour, in order to improve the readability and reduce the complexity).

Data Science and *Machine Learning* oriented good practices: it should be possible to generalize the code in order to have compatibility with several datasets; work with *Version Control*, which allows comparison between the performances of different versions of the code and protects the work integrity; use *pre-existing libraries* whenever possible since they are well-documented and implemented and it is unnecessary to invent a new way to do some task if it is not crucial for the project.

2. Principal Components

Principal Component Analysis (PCA) is a dimensionality reduction technique used in the field of machine learning. The main goal of PCA is to find patterns in data, by means of the correlation between variables [7]. The algorithm uses orthogonal transformations to convert a set of possibly correlated variables into a set of linearly uncorrelated variables called *Principal Components*.

The transformation is defined in such a way that the *First Principal Component (PC1)* has the largest possible variance (meaning that it accounts for as much data variability as possible). Each succeeding component has the highest possible variance, under the constraint that it is orthogonal to the preceding components. PCA is sensitive to the scaling of the original variables.

2.1 PCA over *Banknote Authentication*

The first step consists in the standardization of the data, by centering and dividing them for the standard deviation.

$$x_{ik} = \frac{x_{ik} - \bar{x}}{\sigma_x}$$

Once having computed the correlation matrix and its eigenvectors and eigenvalues, it is possible to decide how many components will be necessary to have a meaningful representation of the data. Indeed, the eigenvectors with the highest eigenvalues are the ones that bear the most information about the distribution of the data.

From the *pairplots* in Fig. 8.1 it is possible to see that there is no significant correlation between the features in the dataset, except for skewness and kurtosis.

From the results of PCA analysis, it was observed that, considering the first two eigenvectors, the retained variance corresponds to 86.8% of the variance of the original dataset. This could mean that it may be sufficient to train the classifiers with just these two components [8]. The data distribution in the two-dimensional feature subspace is shown in Fig. 2.1.

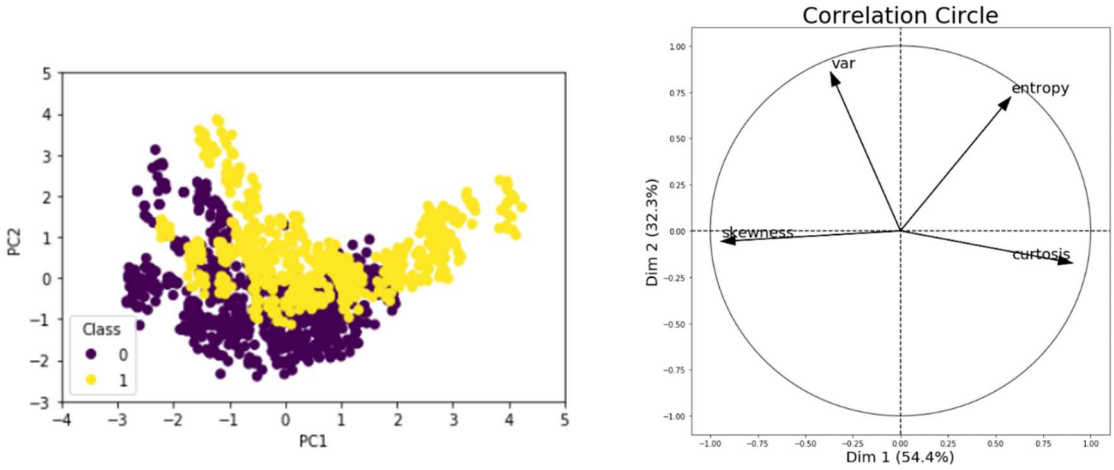

Fig. 2.1 – Plot of two *Principal Components* (*left*), *Correlation Circle* (*right*).

From the correlation circle (Fig. 2.1), it is possible to see that the *curtosis* has a strong positive correlation with the first component, while for the *skewness*, the correlation is negative (the cosine is almost equal to -1).

2.2 PCA over *Chronic Kidney Disease*

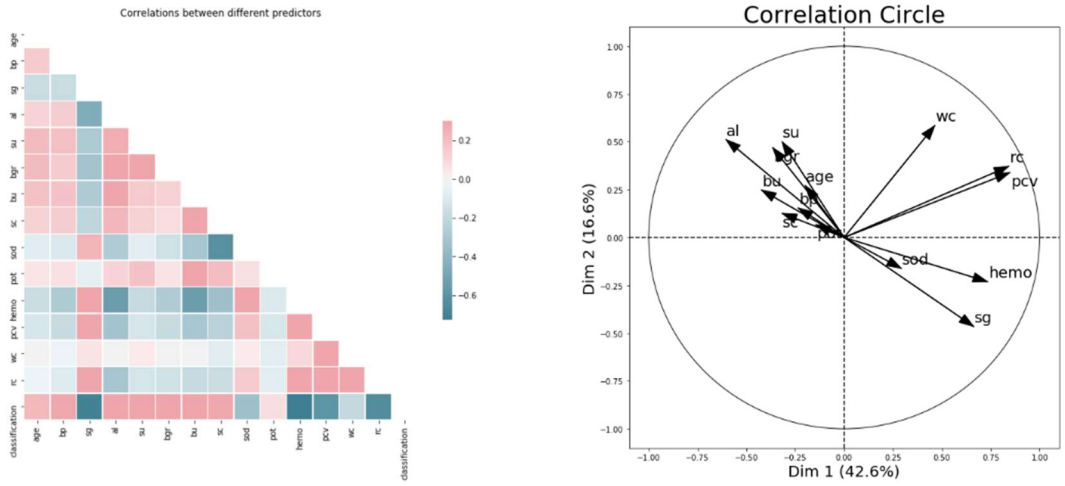

Fig. 2.2 – Correlation between the features of the CKD database (*left*) and *Correlation Circle* (*right*).

In order to remove features with low variance and to generate a more robust classification in the subsequent steps, *Principle Component Analysis* was also applied to the database of Chronic Kidney Disease. The first six components were selected, as they accounted for approximately 82% of the overall variance.

As it can be seen in the correlation circle in Fig. 2.2 the variable *"hemo"* (the *hemoglobin level*) is one of the most correlated with the *first principal component*, meaning that it bears a lot of variance among the variables. At the same time, even the variable representing *potassium* has a strong correlation with the *first principal component*, but it is negative (this means that the individuals with a high value on PC1 will have a low value of potassium).

3. Support Vector Machine

Support Vector Machines (SVMs) are supervised learning models that evaluate the data and recognize the patterns to classify them. They are used for both classification and regression analysis. An SVM is a discriminative classifier, formally defined by a separating hyperplane: it creates a decision boundary to separate the two classes inside data. The data is plotted on the graph and then classification is performed to find the hyperplane that differentiates the two classes [9].

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using two approaches. The first approach is to add positive values that represent the distance between the point and the margin, when the points are in the wrong side. The second approach is what is called the kernel trick: SVM uses a kernel function that projects data from a lower-dimensional space to a higher-dimensional space [10].

3.1 SVM over *Banknote Authentication*

Since the problem is a classification task, it is necessary to use the *Support Vector Classifier* class, which can be found as *SVC* in the *Scikit-Learn's* library. This class requires the specification of one parameter, which is the *kernel type*.

Results were analysed for three different types of *kernel functions*. In the case of a *linear kernel*, the output shows that the SVM prediction accuracy is about 99.41%, with precision and recall equal to 100% for *class 0* (the fake notes) and 99% for *class 1* (the authentic notes).

For what concerns a *polynomial kernel*, the results were worse, with an overall accuracy of 98.25%, precision of 99% and recall of 98% for the *class 0* and precision of 97% and recall of 99% for *class 1*.

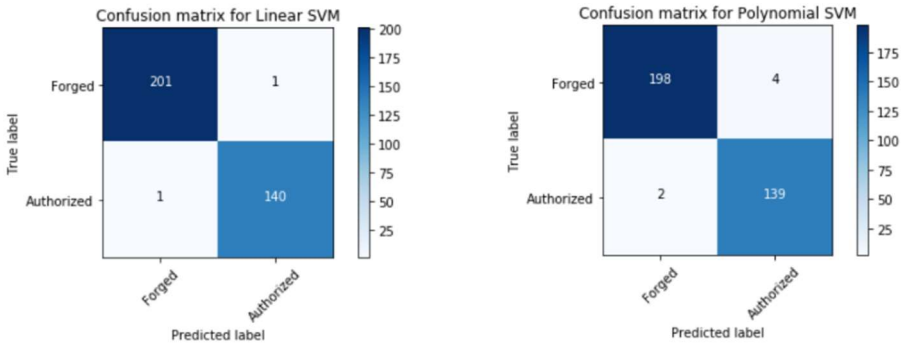

Fig. 3.1 - *Confusion matrix* of an SVM with a *linear kernel* (*left*) and an SVM with a *polynomial kernel* (*right*).

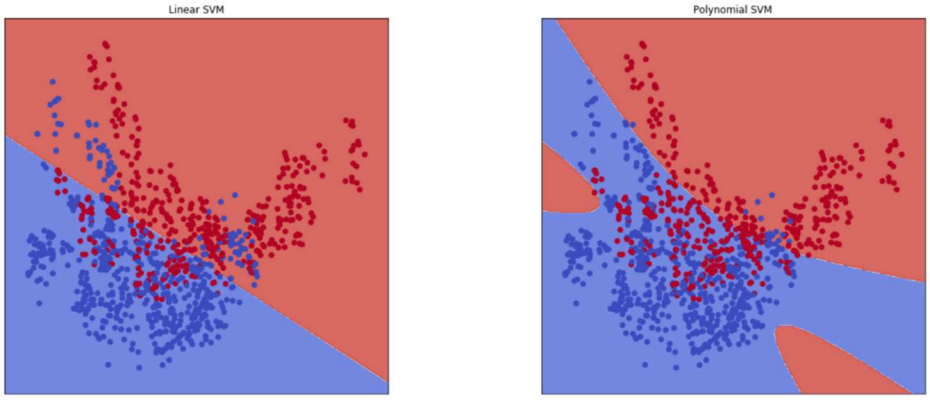

Fig. 3.2 - Plot of the classification performed by the SVM with *linear kernel* (*left*) and *polynomial kernel* (*right*).

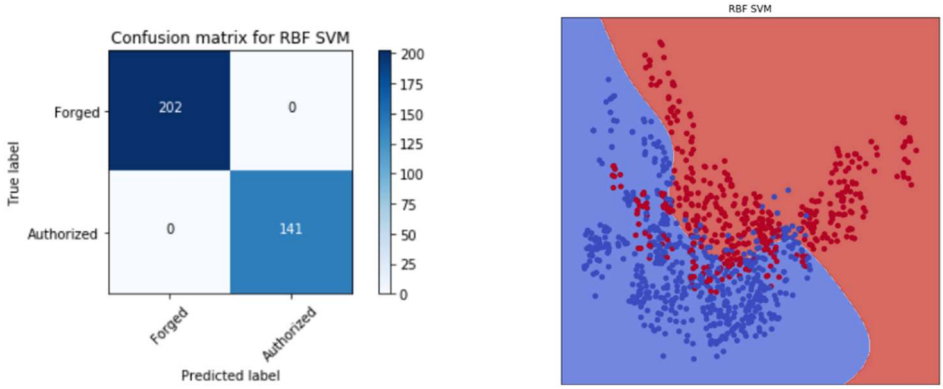

Fig. 3.3 - Confusion matrix (*left*) and plot of the classification (*right*) performed by an SVM with a *Radial Basis Function kernel.*

Finally, the task was performed with a *radial basis kernel function*, which gave a classification accuracy was of 100% achieving a perfect classification [11].

$$K(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right), \gamma > 0$$

3.2 SVM over *Chronic Kidney Disease*

As for the second database, the SVM classification was performed, considering only the first seven *principal components* obtained through the PCA described above. Even in this case, three different *kernel functions* were considered. The results with a *linear kernel* show an overall *accuracy* of 98%, *precision* of 100% and *recall* of 95% for the class corresponding to the *absence* of the *kidney disease*, with only two misplaced elements. Moreover, a *precision* of 97% and *recall* of 100% have been obtained for the class corresponding to the *presence* of the *kidney disease*. The *polynomial kernel* and the *radial basis function kernel* showed pretty much the same results, with an *accuracy* of 97%.

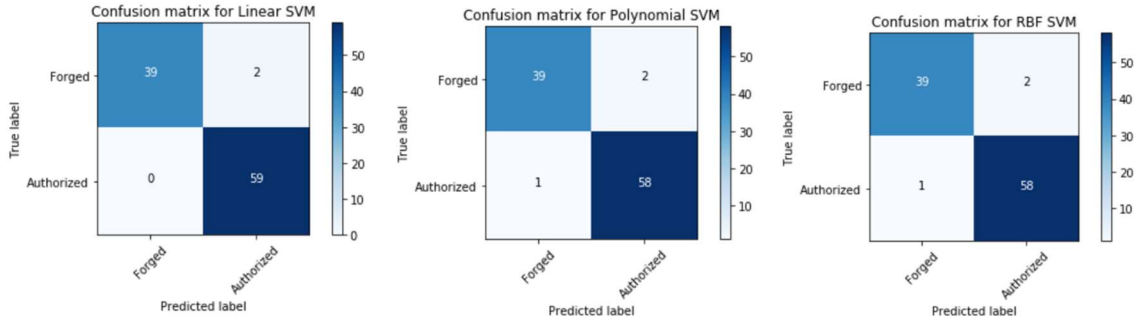

Fig. 3.4 - Confusion Matrix for the *linear kernel* (*left*), *polynomial kernel* (*center*) and *rbf kernel* (*right*).

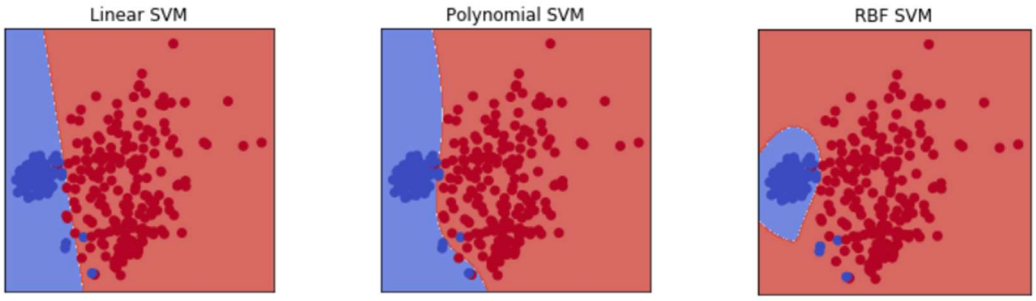

Fig. 3.5 – Plots of the classification done by different SVM kernels.

4. Neural Networks

Neural Networks (NNs) are computational systems inspired by biological neural networks. An Artificial Neural Network is based on a collection of connected units or nodes called artificial neurons, which loosely model neurons inside a biological brain.

Each connection, like synapses of a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal, processes it and send the output to the neurons connected. The connections are called *edges*. Neurons and edges typically have a *weight* that changes as learning proceeds [12].

Neural Networks can be used in both *supervised* and *unsupervised learning*. In the case of *supervised learning algorithm*, the *backpropagation algorithm* is used in order to adjust the *weights*, or *parameters*, until they are optimal for obtaining the correct outputs [13]. Backpropagation NN works in two phases: *propagation* phase (where the input is *forwarded* through the network to generate the output) and *weight update phase* (where the *parameters* in the network are updated according to the optimization algorithm applied on the *error*, computed by subtracting the output obtained to the one expected).

4.1 Neural Networks over *Banknote Authentication*

In this case, the NN is represented by a *Multilayer Perceptron* [14]. The number of *input nodes*, here *four*, is determined by the structure of the problem data. The number of *hidden layers* (*two*) and the number of *nodes* in each *hidden layer* (*four* for the first and *eight* for the second) are *free parameters* (often called *hyperparameters*), which be determined by trial and error. For the *output layer*, whenever the task is a *binary classification*, the number of neurons used is *one* and the activation function is the *sigmoid*.

The model was trained with the *stochastic gradient descent* optimization algorithm, with a learning rate of 0.01. This method is efficient in implementing the *backpropagation algorithm*, adjusting the weights in the network by comparing the desired output with the actual output and distributing the *error* back to the *hidden layers*. The *cost function*, also called *loss function*, is a *binary cross-entropy*, specific for the *binary classification*.

The number of *epochs* to use represents the number of iterations during the training of the network. The simulation uses a *batch* of size 20, which is called *mini-batch training*. *Deep neural networks* are very sensitive to the batch size. Therefore, when training fails, this is one of the first hyperparameters to adjust.

The accuracy on the *training data* is of 99.51%, while for *test data* it is of 98.54%.

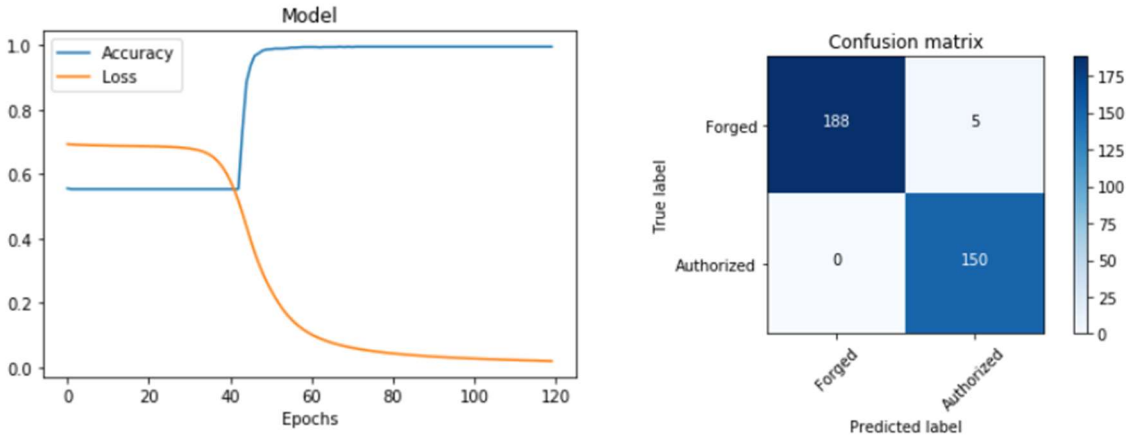

Fig. 4.1 – Evaluation of the model on the training data (*left*) and *confusion matrix* (*right*).

The plot of the *confusion matrix*, as shown in Fig. 4.1, demonstrates that there are only five misplaced notes (i.e., considered as authorized while being fake) out of 323 (which is the dimension of the dataset used for testing).

4.2 Neural Networks over *Chronic Kidney Disease*

For the other dataset, the Neural Network implemented consisted of *two* layers: the first with 256 neurons and second with 1 (used to perform the classification). Since the goal is again a *binary classification*, as for the *Banknote Authentication* dataset, the activation function of the last layer is again a *sigmoid* and the *loss function* is a *binary cross-entropy*.

With a *batch* of size 20 and a number of iterations equal to 500, the accuracy obtained on the *training data* was equal to 99.33%, while the one obtained on *test data* was 96%.

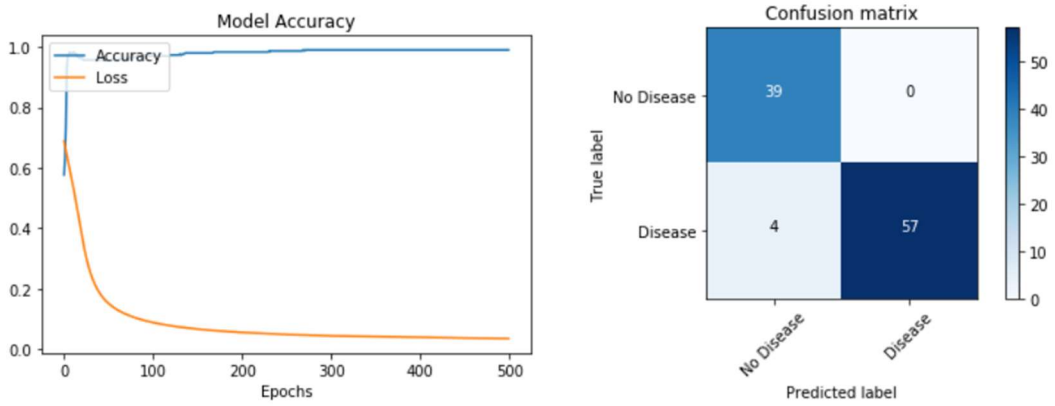

Fig. 4.2 - Evaluation of the model on *training data* (*left*) and *confusion matrix* (*right*)

The classification was once more performed on the dataset obtained after dimensionality reduction performed with the PCA. The *confusion matrix* in Fig. 4.2 shows that there are four misplaced notes (that were considered healthy while having a kidney disease) out of 100, dimension of the dataset used for testing, which is coherent with the result of accuracy equal to 96%.

5. Decision Trees

Decision trees are non-parametric supervised learning algorithms used for both classification and regression. In this case, we will work with classification decision trees. In classification, the aim of decision trees is to create a model that classifies the variable by learning simple decision rules inferred from data features. Tree-based learning algorithms are considered to be one of the best and mostly used supervised learning methods. Tree based methods empower predictive models with *high accuracy* and *stability*. In addition, they are easy to understand and interpret. Unlike linear models, they map nonlinear relationships quite well [15]. They can also work with any type of data: binary, categorical and numerical.

5.1 Decision Trees over *Banknote Authentication*

All of the computations performed on this dataset were done without normalization of the data. For this reason, the first result using *Decision Trees* showed an accuracy rather lower than the other algorithms. Because of that, a normalization was performed on the dataset. It was possible to reach an accuracy of 96% with a tree of depth equal to 3, using *entropy* for *splitting criterion*.

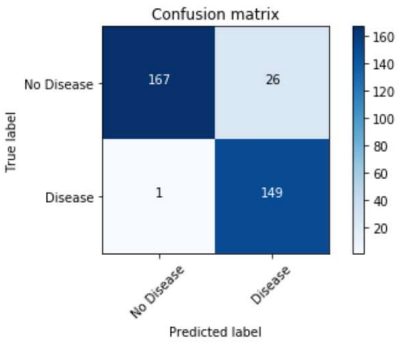

Fig. 5.1 – *Confusion Matrix*.

5.2 Decision Trees over *Chronic Kidney Disease*

In this dataset, several *decision trees* were considered. Throughout a comparison between the different *splitting criteria* of the *decision tree* it was possible to observe that there is no significant difference in the performances when using either *entropy* or *gini*. With respect to the *maximum depth* of the tree, the *classification* results did not improve when adding more layers. Consequently, in order to avoid overfitting, it was fixed a maximum depth of 2, reaching an accuracy of 99%.

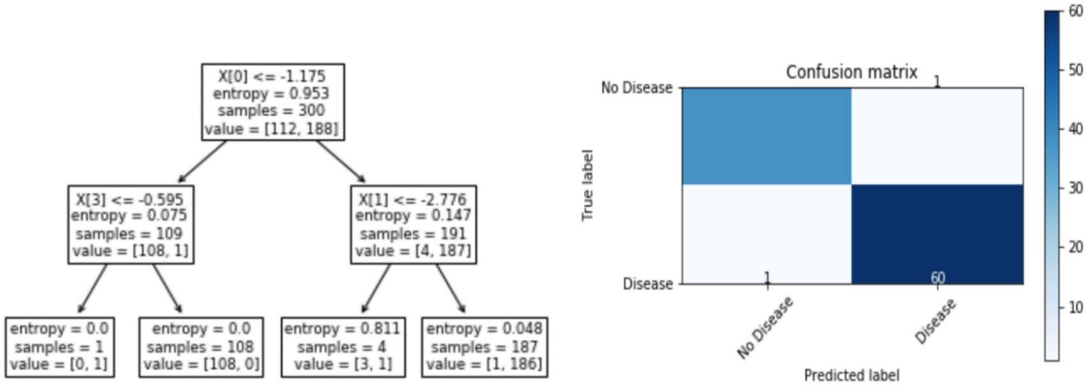

Fig. 5.2 – Representation of the tree (*left*) and *confusion matrix* (*right*)

6. Conclusion

The technique of Principal Component Analysis (PCA) proved to be very powerful: in the first dataset analysed it was possible to reduce the number of variables to two, maintaining approximately 87% of variance. Similarly, for the second dataset, approximately 80% of the variance of the data was preserved by considering only the first 7 *principal components*, which contributes to lower computational complexity in the successive algorithms implemented.

The first classifier analysed was the Support Vector Machine (SVM) which gave slightly different results depending on the *kernel function* used. In this project, three different kernel functions were considered: *linear*, *polynomial* and *radial basis* kernels. In the *Banknote Authentication (BA)* dataset, the best results were found using the *radial basis kernel* with an accuracy of 100%, while for the *Chronic Kidney Disease (CKD)* dataset, the *linear kernel* obtained the highest accuracy, at 98%.

With the *Neural Network* the results were moderately lower, with and accuracy of 98% in the BA test set and of 96% for CKD, but since there are several hyperparameters to fix (like the number of hidden layers and the number of neurons), it may be possible to obtain better scores.

The last algorithm chosen for classification was the *Decision Tree*, which performed as well as the others, with an accuracy around 96% for *Banknote Authentication* and of 99% for the *Chronic Kidney Disease.* In this particular case, it is possible to conclude that the SVM gives the best results for the BA dataset, while the Decision Trees have a better performance for the CKD dataset.

Obviously, there are other *classification* algorithms that could have been explored such as the *Naive Bayes Classifier*, *K-Nearest Neighbour* and the *Random Forest Classifier*, which could be interesting for future experiments [16].

7. References

[1] O. Simeone, *A Very Brief Introduction to Machine Learning With Applications to Communication Systems,* IEEE Transactions on Cognitive Communications and Networking, vol. 4, pp. 648-664, 2018.

[2] S. B. Kotsiantis, *Supervised Machine Learning: A Review of Classification Techniques,* Informatica, vol. 31, pp. 249–268, 2007.

[3] F. Y. Osisanwo, J. E. T. Akinsola, and O. Awodele, *Supervised Machine Learning Algorithms: Classification and Comparison,* International Journal of Computer Trends and Technology (IJCTT), vol. 48, June 2017.

[4] https://archive.ics.uci.edu/ml/datasets/banknote+authentication

[5] https://archive.ics.uci.edu/ml/datasets/Chronic_Kidney_Disease

[6] A. El-Houssainy and A. S. Anwar, *Prediction of kidney disease stages using data mining algorithms*, Informatics in Medicine Unlocked, vol. 15, p. 100178, 2019.

[7] I. T. Jolliffe and J. Cadima, *Principal component analysis: a review and recent developments*, Philos Trans A Math Phys Eng Sci, vol. 374, p. 2065, 2016.

[8] M. Tejasvi, A. Nayeemulla Khan, and A. Shahina, *Perfecting Counterfeit Banknote Detection-A Classification Strategy,* Int. Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, pp. 435-440, 2019.

[9] S. Shahani, A. Jagiasi, and R. L. Priya, *Analysis of Banknote Authentication System using Machine Learning Techniques*, International Journal of Computer Applications, vol. 179, pp. 22-26, 2018.

[10] A. Patle and D. S. Chouhan, *SVM Kernel Functions for Classification*, Proc. 2013 International Conference on Advances in Technology and Engineering (ICATE), doi: 10.1109, 2013.

[11] S. l. Lin and Z. Liu, *Parameter selection in svm with rbf kernel function*, Journal-Zhejiang University of Technology, vol. 35, p. 163, 2007.

[12] K. Gurney, *An Introduction to Neural Networks*, Taylor & Francis, Inc. Bristol, PA, USA, 1995.

[13] X. Yu, M. Ö. Efe, O. Kaynak, *A general backpropagation algorithm for feedforward neural networks learning*, IEEE Trans. Neural Networks 2002

[14] E. Wilson, D.W. Tufts, *Multilayer perceptron design algorithm*, Proceedings of IEEE Workshop on Neural Networks for Signal Processing, 0-7803-2026-3

[15] H. Ishwaran, J. S. Rao, *Decision Tree: Introduction*, Computer Science, Encyclopedia of Medical Decision Making. Sage Publications. pp. 323--328 (2009)

[16] D. R. Amancio, C. H. Comin, D. Casanova, *A Systematic Comparison of Supervised Classifiers*, 10.1371/journal.pone.0094137 (2014)

8. Appendix

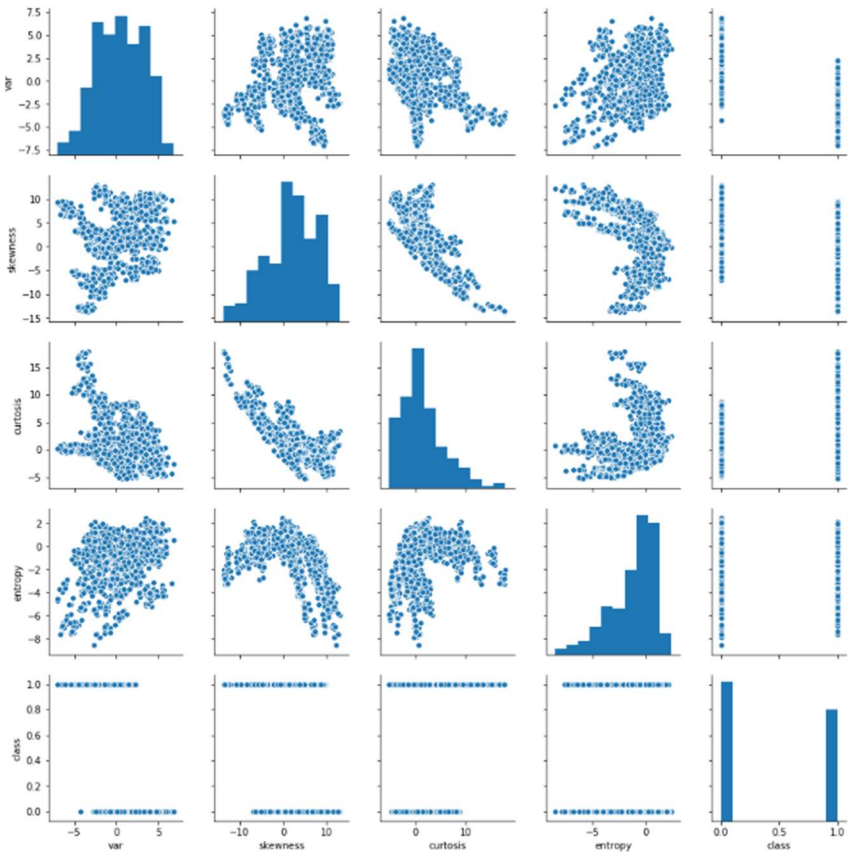

Fig. 8.1 – *Pairplots* and *histograms* of *Banknote Authentication* dataset

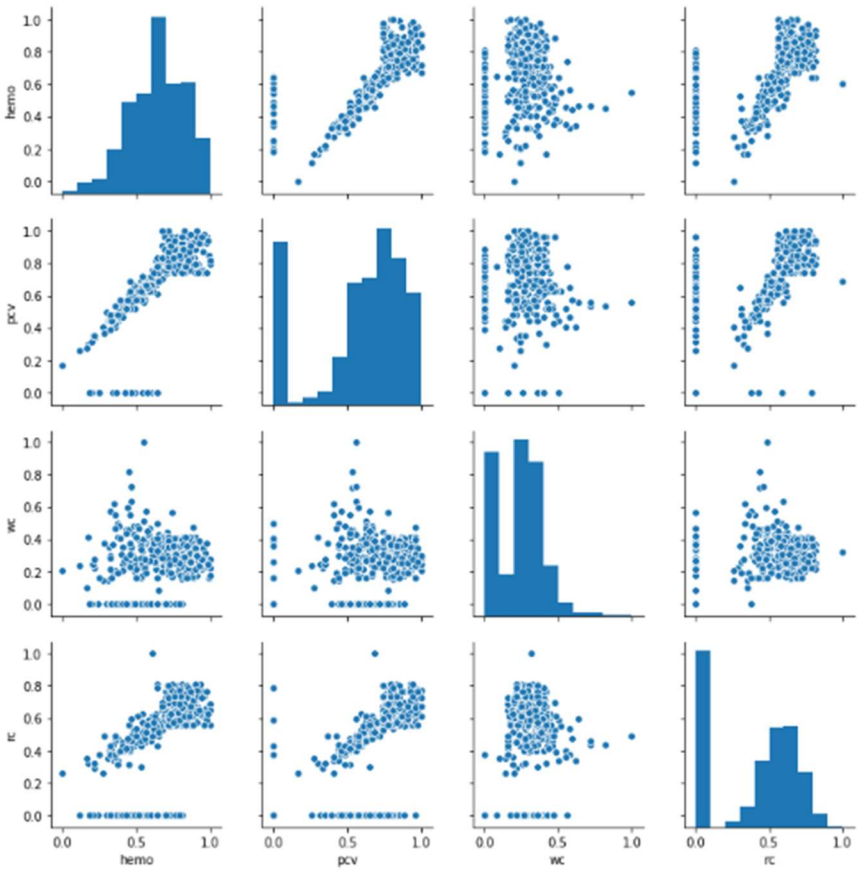

Fig. 8.2 – *Pairplots* and *histograms* of *Chronic Kidney Disease* dataset

8.1 GitHub Log

```
commit f20ff96bac340004a0be1425ea673c9409c7515e
Merge: 72afdde 1e0620b
Author: gustavordr <55882949+gustavordr@users.noreply.github.com>
Date:   Wed Dec 4 21:13:29 2019 +0100

    Merge pull request #19 from morokhalid16/martina

    Changes in the  cleaning_function

commit 1e0620b43d4cec0ee7c67bb8567084a54638274d
Author: Gustavo Rodrigues <gleitor@gmail.com>
Date:   Wed Dec 4 21:11:41 2019 +0100

    Changes in the  cleaning_function

commit 72afdde8658a10c3190afe19550d17893c916cba
Merge: 8b9064e fa390b8
Author: gustavordr <55882949+gustavordr@users.noreply.github.com>
Date:   Wed Dec 4 19:54:24 2019 +0100

    Merge pull request #18 from morokhalid16/martina

    Added cleaning_data function

commit fa390b84c7c91d510c4e7b18a087a466e2ad6420
Author: Gustavo Rodrigues <gleitor@gmail.com>
Date:   Wed Dec 4 19:53:28 2019 +0100

    Added cleaning_data function
```

```
commit 8b9064e9b8f87a1418c466b59e4b221e4f59be31
Merge: e197ca1 b25f481
Author: gustavordr <55882949+gustavordr@users.noreply.github.com>
Date:   Wed Dec 4 17:11:33 2019 +0100

    Merge pull request #17 from morokhalid16/martina

    Gustavo

commit b25f481e5c4e21d3e9755365b408c21a1d000235
Merge: 7e9aab2 e197ca1
Author: gustavordr <55882949+gustavordr@users.noreply.github.com>
Date:   Wed Dec 4 17:11:11 2019 +0100

    Merge branch 'master' into martina

commit 7e9aab2336ea00460acfc076c5bfb00f627aa01e
Author: Gustavo Rodrigues <gleitor@gmail.com>
Date:   Wed Dec 4 17:08:13 2019 +0100

    Changes in cross-validation

commit b508bee9c8657a39dba2b8ca6d8f859bd3f10517
Author: Gustavo Rodrigues <gleitor@gmail.com>
Date:   Tue Dec 3 22:15:32 2019 +0100

    Addition of Decision Trees Classifier and Cross Validation function

commit e197ca1b93b8a5ea9b5ab12d4695ec9872447970
Merge: d48cac5 ffbb4dd
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Dec 2 23:34:06 2019 +0100

    Merge branch 'khalid'

commit ffbb4dd189c4b4b1b51533228f1e85eaf1f512e7
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Dec 2 23:28:00 2019 +0100

    Round up pca and split fils in only one file dataClassification.py

commit d48cac5249b6c2b7a041e93bb9fc87e94e9e6921
Merge: 68920aa 2d935d1
Author: morokhalid16 <50218858+morokhalid16@users.noreply.github.com>
Date:   Mon Dec 2 22:44:17 2019 +0100

    Merge pull request #16 from morokhalid16/khalid

    Merge remote-tracking branch 'origin/master' into khalid

commit 68920aa3388187ddd0247edd9c728164b078170b
Merge: a42714f 839677b
Author: Maiti96 <57621946+Maiti96@users.noreply.github.com>
Date:   Mon Dec 2 22:42:53 2019 +0100

    Merge pull request #15 from morokhalid16/martina

    little changes

commit 839677bfbddf6a23173a225892670ea2760a0ec5
Author: Martina <martina.pastorino7@gmail.com>
Date:   Mon Dec 2 22:41:31 2019 +0100

    little changes

commit 2d935d16f5386bd7533929ca546181274b5cb485
Merge: ca963b2 a42714f
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Dec 2 22:38:21 2019 +0100

    Merge remote-tracking branch 'origin/master' into khalid

commit a42714f887beebe601953b229185e70c51c34320
Merge: 617725f 2374997
Author: mmeri <mrieramarin@gmail.com>
Date:   Mon Dec 2 20:08:32 2019 +0100
```

```
    Merge pull request #14 from morokhalid16/meritxell

    svm function

commit 237499792bee8c16205f2017762bc2f92070b94e
Author: meritxell <mrieramarin@gmail.com>
Date:   Mon Dec 2 19:58:17 2019 +0100

    svm function

commit 617725fb8b806e7cfc116c792007041cecab853d
Merge: fab11d8 6e461d1
Author: Maiti96 <57621946+Maiti96@users.noreply.github.com>
Date:   Mon Dec 2 18:43:26 2019 +0100

    Merge pull request #13 from morokhalid16/martina

    neural network

commit 6e461d103408c5736f6f70a262cdf00016eaf57f
Author: Martina <martina.pastorino7@gmail.com>
Date:   Mon Dec 2 18:41:32 2019 +0100

    neural network

commit fab11d81fb3aea267d827d5df618eca534630bff
Merge: c7fa216 4178d6b
Author: Maiti96 <57621946+Maiti96@users.noreply.github.com>
Date:   Mon Dec 2 16:47:11 2019 +0100

    Merge pull request #12 from morokhalid16/martina

    adding pca modified function in original file

commit 4178d6be537e444c1d0bc8793155f822776b62c4
Author: Martina <martina.pastorino7@gmail.com>
Date:   Mon Dec 2 16:46:04 2019 +0100

    adding pca modified function in original file

commit c7fa216ce6d7c48ed4ad1a9dc5646c0cead66191
Merge: 773ae7a 7aeead9
Author: Maiti96 <57621946+Maiti96@users.noreply.github.com>
Date:   Mon Dec 2 16:42:49 2019 +0100

    Merge pull request #11 from morokhalid16/martina

    addiction and update functions

commit 7aeead919fcf03d64a5ce19e85e7082f397132ab
Author: Martina <martina.pastorino7@gmail.com>
Date:   Mon Dec 2 16:40:58 2019 +0100

    addiction and update functions

commit 773ae7a1216a1515c9c841fdf463aab2938f1c9c
Merge: 7bfd9f1 ca963b2
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 04:24:42 2019 +0100

    Merge branch 'khalid'

commit ca963b25689bf7ace7236807801d6aad76d409ee
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 04:24:22 2019 +0100

    Edit methods load_data & col_target_def

    for load_data add condition because when we work with bank data we need to have
no header
    for col_target_def, I remark that we need to write if flage=="true" rather than
if flag. Thus, both datasets can be cleaned with clean_data function.

commit 8743f017d0b772a81a2bfd835090e417f600a1cb
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 04:17:54 2019 +0100
```

```
    flag condition

commit 7bfd9f1ae2267d17bcda97369c7714c80358031d
Merge: 48aa887 0569cab
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 03:55:25 2019 +0100

    Merge branch 'khalid'

commit 0569cab74c1938f258bb124ff86cd643128f0ea6
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 03:55:09 2019 +0100

    Adapted for kidney

    split now returns train&test datasets for features as well as for  target data
for kidney

commit 48aa8875bebc507f5390f14c8205562d6cc044d4
Merge: e7bf6a8 ac54ea9
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 03:06:24 2019 +0100

    Merge branch 'khalid'

commit ac54ea99236b16af5d2bd1389666a3976ef41afe
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 03:04:30 2019 +0100

    Add split.py file to split data into test and train

commit e7bf6a8946a73de864eca20b0c4eadf54c803510
Merge: e56adb5 f62b012
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 02:55:12 2019 +0100

    Merge branch 'khalid'

commit f62b0123901d00d2337dc4034618060f7e147817
Merge: f1d02f6 e56adb5
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 02:52:07 2019 +0100

    Merge branch 'master' into khalid

commit e56adb54e8bc3616d8654a84dbd89a716cc2df29
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 02:51:33 2019 +0100

    Edit the return of the function clean_data

    Now it returns a complete dataset with the same columns as the original one

commit f1d02f6cbc5e9194a783408d0d86ffc189a6b261
Merge: fdbf820 baf3824
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 02:07:20 2019 +0100

    Merge remote-tracking branch 'origin/khalid' into khalid

commit fdbf82046c0000ecbbd2ded587b456f364d1363c
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 02:06:23 2019 +0100

    All files

commit baf38241dc15fa467ebfe46ae70058dfe5fe1d21
Author: MORO <morokhalid16@gmail.com>
Date:   Wed Nov 20 02:04:18 2019 +0100

    Add all necessary files to the project

commit a91cb75e91919de2a2d4397e70776b4f20c2721f
Author: MORO <morokhalid16@gmail.com>
Date:   Tue Nov 19 09:38:46 2019 +0100
```

```
    Add pca functions

commit b0fe35dec8a8e9738fc3df809bc52acb3855877c
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 18 18:10:22 2019 +0100

    for windows we need to put r before data_path

commit 11ad68afe0eab1cf59039093c6165405de2f1c4d
Merge: 7fcb97e 59abb0f
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 18 18:08:29 2019 +0100

    Merge remote-tracking branch 'origin/master' into khalid

commit 59abb0f0b8d151626e62a0fe441c1a74e424f5d6
Merge: e21af15 6ecabee
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 18 17:52:59 2019 +0100

    Merge branch 'khalid'

commit 6ecabeeddff6bdf43b18fd893da90de67fb982fe
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 18 17:52:45 2019 +0100

    read_data(data_path) without deleting and small modification for windows

commit 5a81c3a5065ebefe0152d8357b17bda7c3c31919
Merge: 7fcb97e e21af15
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 18 17:36:03 2019 +0100

    Merge branch 'master' into khalid

commit e21af1592af2075da62b0ec037630ba8de7eeaa4
Merge: 65ec1c4 f05413a
Author: mmeri <mrieramarin@gmail.com>
Date:   Sun Nov 17 19:41:13 2019 +0100

    Merge pull request #8 from morokhalid16/meritxell

    visualize_data()

commit f05413a30b37cae025373da87849fedbbe834063
Author: meritxell <mrieramarin@gmail.com>
Date:   Sun Nov 17 19:32:30 2019 +0100

    visualize_data()

commit 65ec1c4f2237f3db5c23700d2c24d62b80c78610
Merge: ea9cf9a 3e47451
Author: mmeri <mrieramarin@gmail.com>
Date:   Sun Nov 17 18:58:34 2019 +0100

    Merge pull request #7 from morokhalid16/meritxell

    modifications

commit 3e474513796ff4c13ec009f8e6c7a53c0a4ef195
Author: meritxell <mrieramarin@gmail.com>
Date:   Sun Nov 17 18:57:20 2019 +0100

    small modification in clean_data()
    created col_target_def()

commit ea9cf9ad9e315406e3b9fca9d27ca9019f9065fd
Merge: 89bad08 6dbd09a
Author: Maiti96 <57621946+Maiti96@users.noreply.github.com>
Date:   Sun Nov 17 15:48:15 2019 +0100

    Merge pull request #6 from morokhalid16/martina

    Update of the functions

commit 6dbd09abe5b0e325d4576a8ba2301e26162da43b
Merge: d9376dc 9ba3b19
```

```
Author: Martina <martina.pastorino7@gmail.com>
Date:   Sun Nov 17 15:44:46 2019 +0100

    Merge remote-tracking branch 'origin/martina' into martina

commit d9376dc4beaf2125e6a4816d92ecce6afbc0dfca
Author: Martina <martina.pastorino7@gmail.com>
Date:   Sun Nov 17 15:44:14 2019 +0100

    update functions

commit 9ba3b19678ad892c3939c4a0446349e0a0f15ff0
Merge: f4755c9 89bad08
Author: Maiti96 <57621946+Maiti96@users.noreply.github.com>
Date:   Sun Nov 17 15:41:47 2019 +0100

    Merge pull request #5 from morokhalid16/master

    upload files

commit 356a3f01478af1f28c50dab8af3fea56c4a22a99
Author: Martina <martina.pastorino7@gmail.com>
Date:   Sun Nov 17 15:28:15 2019 +0100

    update functions

commit 89bad08a0c5060a09292a7f229e71f1c454bd62b
Merge: 97bbfb9 14813f1
Author: mmeri <mrieramarin@gmail.com>
Date:   Mon Nov 11 13:07:09 2019 +0100

    Merge pull request #4 from morokhalid16/meritxell

    remove test.py

commit 14813f19accf79f7fffc6872f146469814fb0483
Author: meritxell <mrieramarin@gmail.com>
Date:   Mon Nov 11 13:06:10 2019 +0100

    remove test.py

commit 97bbfb9e56074cdbe817717d3f05ce392c25c26d
Merge: efc7dc3 f4755c9
Author: Maiti96 <57621946+Maiti96@users.noreply.github.com>
Date:   Mon Nov 11 13:05:00 2019 +0100

    Merge pull request #3 from morokhalid16/martina

    init read_data

commit f4755c99067f9b2e74bee40fe5cac7c3f5461960
Author: Martina <martina.pastorino7@gmail.com>
Date:   Mon Nov 11 13:01:33 2019 +0100

    init read_data

commit efc7dc3be785d11d4d9fb40f63b1a6027aaf03b2
Merge: 7fcb97e ee20da2
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 11 12:54:06 2019 +0100

    Merge branch 'khalid'

commit ee20da2541e56afb5d57c9a0732d706cc9ea01e8
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 11 12:51:46 2019 +0100

    second edit

commit 7fcb97e0a912c62eb703a549c80b83fbf42200ba
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 11 12:46:54 2019 +0100

    changing function read_data()

commit 97e3da8af069a5ae0f3201215d83e23c5d297f2a
Merge: 3941955 e590fb7
```

```
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 11 12:40:19 2019 +0100

    Merge branch 'master' into khalid

commit e590fb7ff7f63cee22cfebc2c088ce058e096cdb
Merge: 4469c83 3941955
Author: morokhalid16 <50218858+morokhalid16@users.noreply.github.com>
Date:   Mon Nov 11 12:20:37 2019 +0100

    Merge pull request #2 from morokhalid16/khalid

    test file

commit 3941955a57bf848cf676a90ef1189a24a11a75bb
Author: MORO <morokhalid16@gmail.com>
Date:   Mon Nov 11 12:15:35 2019 +0100

    test file

commit 4469c839224f856098215ff81b147d8948ff4c1c
Merge: 9dbda54 61384ad
Author: mmeri <mrieramarin@gmail.com>
Date:   Mon Nov 11 12:08:43 2019 +0100

    Merge pull request #1 from morokhalid16/meritxell

    function names

commit 61384ad5d8dfafd1064240a0b3a9c2ed6dd75f02
Author: meritxell <mrieramarin@gmail.com>
Date:   Mon Nov 11 12:00:56 2019 +0100

    function names

commit 9dbda540313fc9ac81b413d70f29898823388bce
Author: morokhalid16 <50218858+morokhalid16@users.noreply.github.com>
Date:   Mon Nov 11 11:37:12 2019 +0100

    Initial commit
```