Will you be cheated? A Machine Learning approach to credit card fraud detection

RICCARDO BRUSA[1] AND LUCA PESENTI[1]

[1] *Università degli Studi di Milano - Bicocca, CdLM Fisica*

Compiled September 19, 2020

Detecting credit card frauds is an important issue, since a remarkable amount of money can be involved, causing set of problems to banks as well as to the customers. The study presented herein is based on a huge dataset referred to two days of registered transactions: a first analysis on the dataset highlights a strong imbalance of the class attribute which establish if a transaction is fraudulent or not. After pre-processing and features engineering procedures, several classification models have been taken into consideration, trained, tested and compared, without discovering an pronounced superiority of one particular classifier. The second part of this work is devoted to clustering analyses, which have led in identifying 2 clusters. In conclusion, a Monte Carlo procedure has been exploited to perform a hypothesis test, hence a clustering validation.

Keywords: Bank, Classification, Clustering, Credit Card, Fraud, Machine Learning, ML, Prediction, Transactions

CONTENTS

1. INTRODUCTION

Thanks to the growing digitalization, bank transactions are getting easier as well as frauds. Indeed only in the US in 2018, there was an increase of about 183% with respect to 2014, as it can be seen in Fig. 1. Recognizing

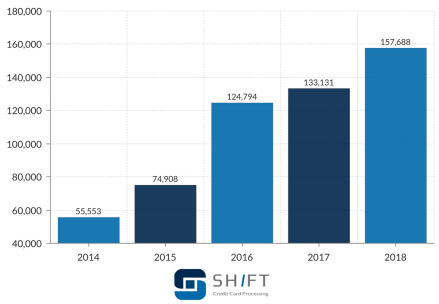

Fig. 1. Credit card fraud reports in the United States [1]

fraudulent transactions is definitely important in the community, in order to prevent customers to be charged for purchases they have not carried out. Moreover, the more

a bank is able to reject fake cash transfers, the safer customers money is and this may be a good advertisement for the credit institutes themselves. Studying the credit card fraud detection problem by means of quantitative methods requires a non-negligible amount of data and a machine learning approach is presented hereinafter. A dataset that contains 284807 transactions made by credit cards is employed; it is referred to two days of registered transactions in September 2013 made by European cardholders and the percentage of frauds is about 0.17 %. Notwithstanding the small rate, the problem must not be underestimated since the implicated amount could be remarkable anyway.

To summarize, the main task is to predict if a credit card transaction is fraudulent or not, by exploiting the attributes available in the dataset and several classifiers. The secondary objective is the identification of potential clusters, comparing different clustering algorithms.

2. DATASET STRUCTURE

The data set is composed of 284807 records with 30 numerical attributes and one binary class characterized by the following convention: 1 for fraudulent transactions, 0 otherwise. Concerning the *understood* numerical attributes, there are the *amount* and the *time*, which represents the number of seconds elapsed between the considered transaction and the first one in the dataset. The remaining 28 attributes (named as V1-V28) can not be easily explained and they may be the result of a Principal Component Analysis (PCA) for dimensionality reduction in order to protect user identities and sensitive features.

First of all, the dataset is deeply imbalanced since it provides 492 frauds, thus a positive-to-negative class ratio equal to $\sim 0.0017 \ll 1$. An equal size sampling has been employed in the classification models studies in order to prevent unbalances (that would be expected by using the as-is dataset) in the performances evaluation.

Taking into account the output of the ***Statistics*** node provided by KNIME, we observe that no records are affected by missing values.

3. CLASSIFICATION MODELS

In this analysis we have decided to use several classification techniques, in order to evaluate which are the best.

A. Classification techniques

A classifier exploits some explanatory attributes (inputs) in the dataset in order to predict the class attributes (outputs). This work takes into considerations a few classifiers belonging to the following main categories:

- Heuristic
- Binomial Logistic Regression
- Support Vector Machine
- Multi-Layer Perceptron
- Naive Bayes
- Bayesian Net

In particular we focused on J48 (Decision Tree), Random Forest, Logistic Regression, SMO Polynomial, MLP, Naive Bayes, NBTree and Bayes Net (TANB).

B. Performance evaluation

Classifiers performances are evaluated by using the following parameters: Accuracy, Recall, Precision, F1-Measure and the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve.

B.1. Accuracy

This parameter gives the fraction of events correctly classified,

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FN} \quad \textbf{(1)}$$

where TP and TN are the data correctly classified as positive/negative whilst FP or FN stands for the data wrongly classified as positive/negative. Therefore, the higher the Accuracy value the better the classification model used. Nevertheless, other parameters are needed in order to compare different classification techniques.

B.2. Recall

This index measures the fraction of positive records correctly predicted by the Classification Model and in particular it is defined as follows,

$$Recall \equiv r = \frac{TP}{TP + FN} \quad \textbf{(2)}$$

A higher r-value means that a few positive records have been misidentified as belonging to the negative class.

B.3. Precision

This parameter determines how many records turns out to be positive in the group of positive class declared by the Classification Model.

$$Precision \equiv p = \frac{TP}{TP + FP} \quad \textbf{(3)}$$

The higher the precision is, the lower the number of false positive errors committed by the Classification Model are.

B.4. F_1 -Measure

Precision and Recall indices are strictly correlated, indeed if the former is higher, probably we will obtain a lower value of the latter and *vice versa*: this could also lead to build Classification Models that maximize only one of the two parameters. In addition, it is not always possible to evaluate these parameters because p could not be defined and r could be equal to zero. To overcome these issues a

new parameter is introduced, the F_1-Measure. It is defined as the harmonic mean between r and p,

$$F_1 = \frac{2 \cdot r \cdot p}{r + p} \quad \textbf{(4)}$$

High values of the F_1 index mean that both the Precision and the Recall have high values.

B.5. AUC

ROC curve plots have on the y-axis the number of positive records expressed as the percentage of the total number of records (%TP), whereas on the x-axis the number of negative records expressed as the total number of records percentage (%FP). In particular, the relevant index is the Area Under the Curve (AUC) that is a good estimator of the goodness of a classifier.

4. FIRST RESEARCH QUESTION

A. Pre-processing

Before proceeding with the analysis, data pre-processing is needed. As a first step, an equal size sampling is applied to the whole dataset, achieving both an imbalance removal and a reshaping which allows to employ complex models without expecting an extreme computation time. Then a *Holdout* method is used and the dataset is split into a *training* (2/3) and a *test* (1/3) set, by means of a *stratified sampling* based on the class attribute.

B. Features selection

Given the significant dimensionality of the dataset (30 explanatory attributes) the *curse of dimensionality*[1] problem

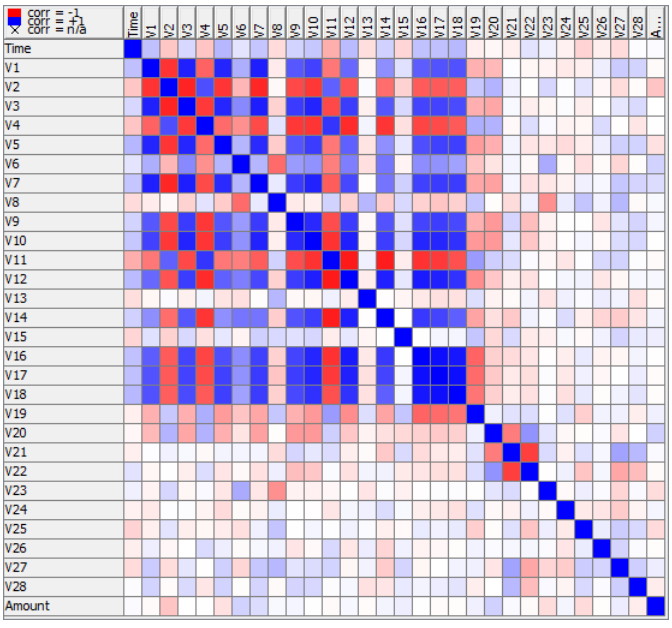

Fig. 2. Correlation matrix.

should be remembered, furthermore, possible sources of overfitting should be rejected. For these reasons, a features selection is extremely important to remove useless or superfluous attributes. The correlation matrix shown in Fig. 2 highlights strong (anti)correlations for some features and it is clear that we should better end up with non-correlated features so that each one of them only improves the information contained in the dataset. As an

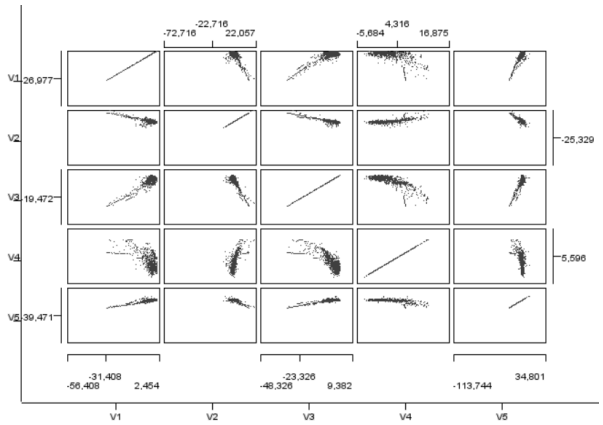

Fig. 3. Scatter matrix referred to V1-V5 attributes.

example, a visualization of the correlation between features V1-V5 is shown in Fig. 3 .

Concerning the Multi-Layer Perceptron classifiers, a ***Correlation Filter*** node has been used, removing V1-V5, V7, V9, V11, V12, V14, V16-V18 and V22 that exhibit correlations above the fixed threshold equal to 0.7. The remaining classifiers benefits from a feature selection that relies on the *filter* approach, where attributes are selected thanks to the objective function analysis before the classifier is invoked: the adopted evaluator is *CfsSubsetEval* with the *BestFirst* search method (selected attributes: V4, V10, V11, V14, V17, V20, Amount).

C. Classifiers analysis

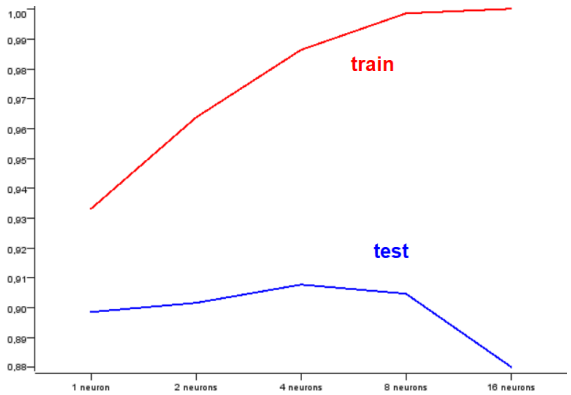

Fig. 4. Overfitting behaviour. On the y-axis the accuracy is represented, while the x-axis is related to the number of neurons per hidden layer.

[1] A higher dataset dimensionality does not strictly imply better performances of the tested models.

Subsequent to features selection, several classifiers are trained then their performances have been evaluated on the test dataset. A preliminary study has been carried out on the Multi-Layer Perceptron[2] by using 100 training epochs at maximum, one hidden layer and an increasing number of neurons per hidden layer, in order to stress the overfitting behaviour as a result of the increasing complexity of the model. By observing Fig. 4, it is clear that the best choice is represented by the MLP model which has 4 neurons per hidden layer, where the obtained test accuracy peaks.

Classifier	Accuracy	r	p	F_1	AUC
J48	0.923	0.890	0.954	0.921	0.953
Random Forest	0.932	0.896	0.967	0.930	0.950
Logistic	0.932	0.877	0.986	0.929	0.968
TANB	0.920	0.883	0.954	0.917	0.971
SMO poly	0.914	0.828	1.000	0.906	0.966
Naive Bayes	0.926	0.865	0.986	0.922	0.914
NBTree	0.920	0.877	0.960	0.917	0.964
MLP	0.908	0.908	0.908	0.908	0.957

Table 1. Classifiers performances.

Table 1 sums up the performances achieved by the different classification models that have been considered. All of the classifiers perform accuracy above 90 % and this fact predisposes us to not exclude some of them at this stage. Hence, other performance indices are taken into account and particular attention is paid to F_1 (see Fig. 5), i.e. the harmonic mean between the *precision* and the *recall*, and the AUC related to ROC curves displayed in Fig. 6. Concerning ROC curves, it is possible to see that every classifier is better than the simplest classification method which relies on the target and ignores all predictors (represented by the bisector). By the way, thanks to results presented above, it is not possible to claim that there is one special method that allows to achieve performances remarkably better than the other ones.

Nevertheless, we want to make a quantitative comparison, thus, in light of the discussed results, we chose the *Logistic regression* and the *Random Forest* classifiers as the possible *best* ones. A *cross-validation* procedure has been used in order to avoid a generalization error.

Fig. 7 shows the result of a same test set carried out on the selected models, where confidence intervals have been employed. Since the above-mentioned box plot is

[2]In order to avoid neurons saturation, MLP models require inputs to be normalized. For this reason a ***Normalizer*** node has been used before the training procedure.

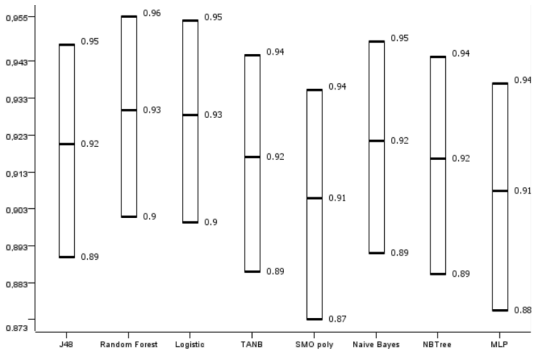

Fig. 5. F_1-measure confidence intervals computed by means of the Wilson formula with a significance level fixed.

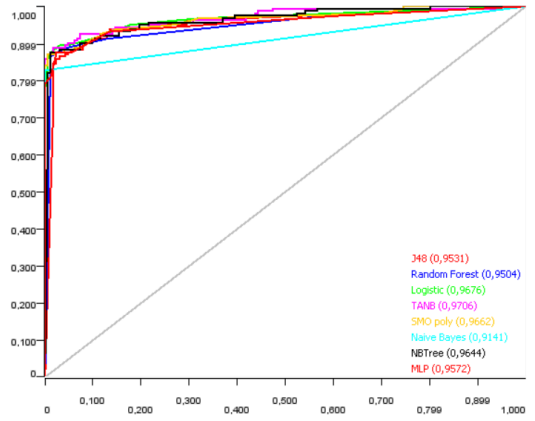

Fig. 6. ROC curves.

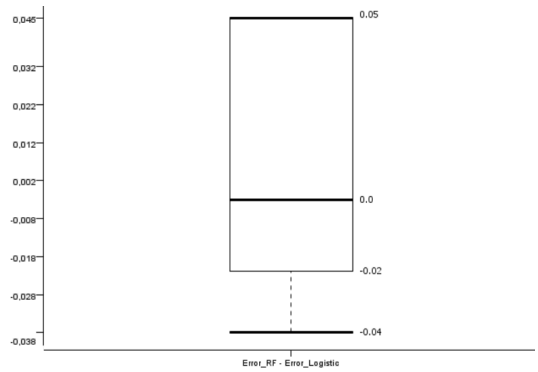

Fig. 7. Box plot of the error difference between *Random Forest* and *Logistic Regression* models.

compatible with 0, we can conclude that the considered models achieves equivalent performances, as suspected.

5. SECOND RESEARCH QUESTION

In the second part of the analysis we focused the attention on the clustering of our dataset. Since clustering algorithms are computationally demanding, we decided to resize our dataset from 284807 records to 1000 using a ***Partitioning*** node with stratified sampling technique. To accomplish this task we decided to use three different clustering methods: *hierarchical*, *K-means* and *K-medoids*. These can be implemented in KNIME using their respective nodes and also through the usage of the language R that let us go deeper in the clustering analysis. In addition, we decided to do not apply an equal size sampling in order to avoid a deformation in the potential clustering structure and to not normalize the data because of the hidden meaning of attributes V1-V28.

Before starting with the clustering analysis we compute the Distance Matrix on our partition, using a ***R Table (View)*** node to evaluate the presence of a structure in our data. As it can be seen in Fig. 8 it is possible to see a pattern in the data that probably leads to a meaningful clustering analysis.

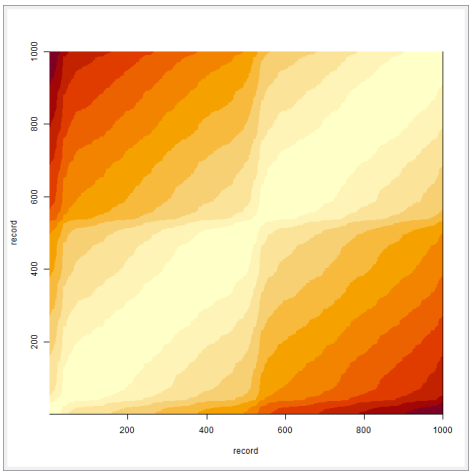

Fig. 8. Distance Matrix

A. Hierarchical model

These type of clustering methods can be divided into two groups:

- **Agglomerative** starts with all records as individual clusters and, at each step, merges the closest pair of clusters. This requires defining a notion of cluster proximity;
- **Divisive** starts with one, all inclusive cluster and, at each step, splits a cluster until only singleton clusters of individual objects remain. In this case, we need to decide which cluster has to be split at each step and how to do the splitting.

In our case we decided to use an agglomerative one, which is also the most common and in particular we used a ***Hierarchical Clustering*** node provided by KNIME. The algorithm used by this method is shown below: The den-

Algorithm 1. Hierarchical Clustering algorithm

```
Compute the proximity matrix                ▷ If necessary
while # clusters > 1 do
    Merge the closest two clusters
    Update the proximity matrix
return dendrogram
```

drogram given in the output can be used to choose the number of clusters, indeed it shows the relations between clusters and sub-clusters. In particular this plot has on the x-axis the records taken into account whereas on the y-axis it has the distance between groups. As it can be no-

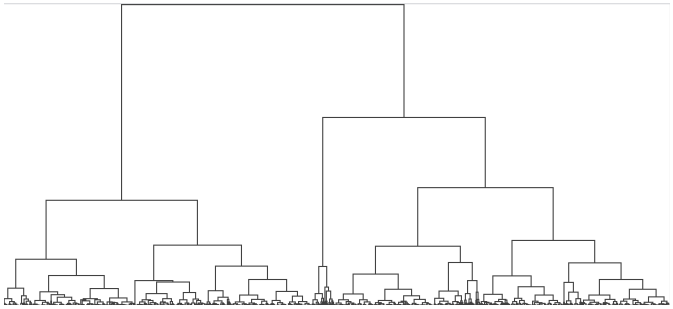

Fig. 9. Dendrogram.

ticed in Fig. 9 different clustering structures can be found and in particular two main clusters are clearly present. In order to maximize the distance between groups, the dendrogram can be split in different ways, however we decided to cut the tree to obtain only two clusters. This choice has been made on the basis of both Silhouette and Dunn indices. For further details see Sec. 6.

B. K-Means and K-Medoids methods

The other two clustering methods used are the *K-means* and *K-medoids* that can be employed using their corresponding KNIME nodes. These methods use a different clustering procedure, in particular they follow an iterative algorithm as reported below, The differences be-

Algorithm 2. K-Means & K-Medoids algorithm

```
Select K records as centroids                ▷ or medoids
for i in range (# records): do
    while centroid[i] ≠ centroid[i-1] do
        Assign each record to the closest centroid ▷ or medoids
        Recompute the centroid of each cluster     ▷ or medoids
return clusters
```

tween these two clustering methods are essentially two:

K-Medoids algorithm is extremely computationally demanding but it is robust with respect to outliers.

In this case the optimal choice of the number of clusters has been made not only by considering the previous results, obtained with the Hierarchical Clustering algorithm, but also by looking at two different indices: Silhouette and Dunn index. Both must be maximized to obtain the best results. As it can be seen in Fig. 10 the number of

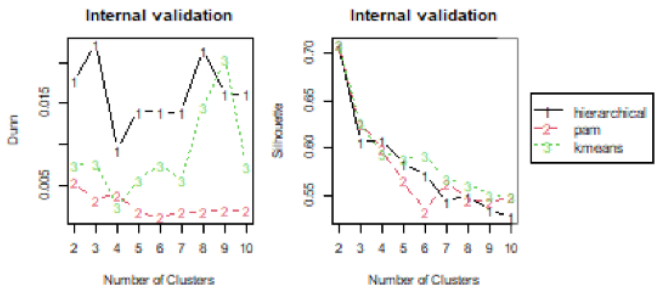

Fig. 10. Variations of the Dunn and Silhouette indices for all the three clustering methods.

clusters that maximize the Dunn index is 2 or 8 but for the Silhouette index the maximum is reached with 2 clusters. Therefore the optimal choice of the number of clusters is 2 because it maximizes both the indices.

In Fig. 11 is shown the output of the Hierarchical clustering algorithm with 2 clusters.

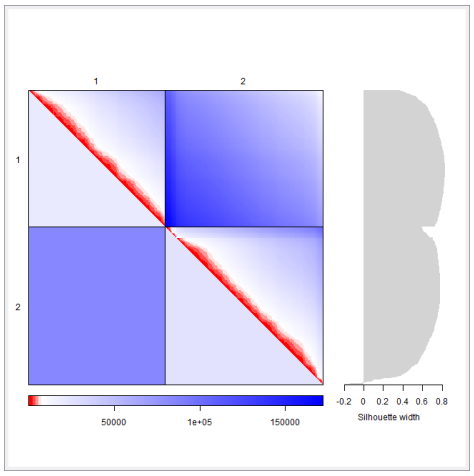

Fig. 11. Output of the Hierarchical clustering algorithm with 2 clusters chosen.

6. CLUSTERING VALIDATION

A known problem when dealing with clustering algorithms is that each one can always produce a partition whether or not a particular structure in the data really exists. Because most of the attributes are unknown, as explained in Sec. 2 it is not possible to use supervised or external indices, therefore only unsupervised or internal indices are employed.

Index	Dunn	Silhouette
Hierarchical	0.0178	0.7070
K-Means	0.0075	0.7091
K-Medoids (PAM)	0.0055	0.7093

Table 2. Internal indices.

A. Internal or unsupervised indices

Many internal measures or indices of cluster validity are based on the notions of:

- *Cohesion*: it tells how much the records in a given cluster are similar to each other;
- *Separation*: this parameter points out the distance between records that belong to different clusters.

In a validation analysis higher values of cohesion and lower values of separation are preferred in order to achieve compact and well-separated clusters.

As mentioned before, in our analysis we employed two different indices: Silhouette and Dunn coefficients. Both combine cohesion and separation. In particular the **Silhouette coefficient** is defined as follows,

$$s_i = \frac{b_i - a_i}{max(a_1, b_i)} \in [-1, +1] \quad \textbf{(5)}$$

where

- a_i is the average distance between the i^{th} object and all the other ones in its cluster;
- b_i is the minimum of the average distances between the i^{th} object and all the other ones, in each given cluster different from the one the i^{th} object belongs to.

A negative Silhouette coefficient means that the average distance to point in its cluster a_i is greater than the minimum average distance to points in another cluster b_i. Therefore, the greater this index[3] is the better the clustering is.

The **Dunn coefficient** can take values in the interval $[0, +\infty]$ and it is the ratio of the smallest distance between observations not in the same cluster and the largest intercluster distance . Indeed, the value of K corresponding to its maximum is taken to be the optimal number of clusters. For further details see [2].

The results reported in Table 2 suggest that the model that achieves the better accuracy in clustering is the hierarchical method, then this method is chosen for the the next analysis.

[3] Remembering that it belongs to the interval $[-1, +1]$

B. Test on the null hypothesis

At the end of the clustering process, performing a test to verify if the dataset really has a cluster structure is mandatory. The validity paradigm consists in the formulation of the null hypothesis H_0 "there is no structures" and in particular it can be summarized in the following statement: *all the locations of the m data points in some specific region of a n-dimensional space are equally likely*. This type of hypothesis is called *Random Position Hypothesis*.

To test the null, hypothesis we used the Silhouette coefficient relative to the hierarchical clustering algorithm. First of all, we have performed a Monte Carlo simulation, reported in Fig. 12, where 1000 samples have been generated. Then we compare the quantile evaluated on this distribution with respect to the value of the statistic-test (Silhouette coefficient), considering a significance level $\alpha = 0.05$. Since the Silhouette coefficient is greater than

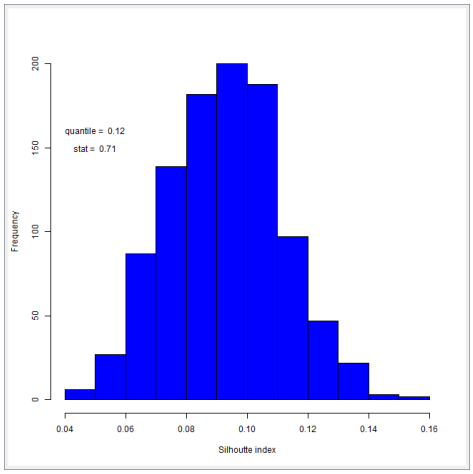

Fig. 12. Monte Carlo simulation with the comparison between the quantile and the statistic-test value.

the quantile ($s = 0.7070 > q = 0.12$) we can reject the null hypothesis H_0 and conclude that our clustering is meaningful.

7. CONCLUSIONS AND FUTURE TASKS

Since the classifiers analysis has not highlighted a model with distinctive performances, we have looked for more sophisticated methods as the application of Generative Adversarial Networks (GANs) where a generator and a discriminator are trying to outsmart each other and we have found an implementation [3] that could be deepened further.

The cluster analysis has reveled that the hierarchical clustering model is the better one. In addition also the test on the null hypothesis confirmed the presence of a data structure in the credit card dataset. Nevertheless, the absence of any kind of information about the attributes V1-V28 does not permit to give exhaustive conclusions about the content of the clusters. Indeed, in the clustering analysis the biggest limits derived from the unknown content of the attributes we dealt with. Having more information about the latter ones would be helpful, in order to evaluate possible correlations that may lead to exhaustive conclusions about credit card frauds. Another possible future task can be the usage of PCA (*Principal Component Analysis*) for clustering.

REFERENCES

1. *Credit card fraud statistics.* https://shiftprocessing.com/credit-card-fraud-statistics/.
2. *clValid package.* http://cran.us.r-project.org/web/packages/clValid/vignettes/clValid.pdf.
3. *Improving Detection of Credit Card Fraudulent Transactions using Generative Adversarial Network.* https://arxiv.org/abs/1907.03355.