Data Mining

Monia Bennici, Gianmarco Pepi

6 July 2020

Contents

1 Introduction

The text is about the analysis of the credit situation of a bank, for which we'll try to understand the main things.
Our goal is to classify the clients, based on their loan status, to understand which are the features of the clients who already paid the loan and the ones who didn't finish. To do so, we have to pass through Data Understanding, Data Cleaning and Data Preparation. Moreover, we will also clusterize the unlabelled data and finally we will look for association rules.

2 Data Understanding

The dataset is composed by 100514 rows and 19 attributes, that give us information about the clients of the bank and their loan situation. Particularly, for every client and every loan there is a "Client ID" and "Loan ID", we have information about the Loan Status and Term. Moreover, every client has a Credit Score and for everyone of them is given the Purpose because they made a Loan and if they have a house. Then, we know their Annual Income , for how long they have worked in the current job, the Year of credit History, the Number of Open Account, how many credit problems they had and the Bankruptcies, how much they pay monthly, the current credit balance, maximum open credit and finally if they have the Tax Liens. The latter, in particular, is a sign that we're talking about an American bank, since this type of lien exists only in America. A tax lien is a lien imposed by law upon a property to secure the payment of taxes. Due to visualization problems we divided the statistics of attributes into two tables containing respectively float and integer attributes(figure 1a and 1b).

	Current Loan Amount	Annual Income	Monthly Debt	Years of Credit History	Current Credit Balance	Maximum Open Credit
count	1.000000e+05	8.084600e+04	100000.000000	100000.000000	1.000000e+05	9.999800e+04
mean	1.176045e+07	1.378277e+06	18472.412336	18.199141	2.946374e+05	7.607984e+05
std	3.178394e+07	1.081360e+06	12174.992609	7.015324	3.761709e+05	8.384503e+06
min	1.080200e+04	7.662700e+04	0.000000	3.600000	0.000000e+00	0.000000e+00
25%	1.796520e+05	8.488440e+05	10214.162500	13.500000	1.126700e+05	2.734380e+05
50%	3.122460e+05	1.174162e+06	16220.300000	16.900000	2.098170e+05	4.678740e+05
75%	5.249420e+05	1.650663e+06	24012.057500	21.700000	3.679588e+05	7.829580e+05
max	1.000000e+08	1.655574e+08	435843.280000	70.500000	3.287897e+07	1.539738e+09

(a) *Statistics of float attributes*

	Credit Score	Months since last delinquent	Number of Open Accounts	Number of Credit Problems	Bankruptcies	Tax Liens
count	80846.000000	46859.000000	100000.00000	100000.000000	99796.000000	99990.000000
mean	1076.456089	34.901321	11.12853	0.168310	0.117740	0.029313
std	1475.403791	21.997829	5.00987	0.482705	0.351424	0.258182
min	585.000000	0.000000	0.00000	0.000000	0.000000	0.000000
25%	705.000000	16.000000	8.00000	0.000000	0.000000	0.000000
50%	724.000000	32.000000	10.00000	0.000000	0.000000	0.000000
75%	741.000000	51.000000	14.00000	0.000000	0.000000	0.000000
max	7510.000000	176.000000	76.00000	15.000000	7.000000	15.000000

(b) *Statistics of integer attributes*

Figure 1: Statistics

Upon 19 attributes, 7 of them are objects ("Loan ID","Customer ID","Loan Status", "Term", "Years in current job", "Home ownership", "Purpose"), while the others are floats.

In general we can say that the majority of the clients is solvent, because there are many sign that make we think that: the loans are fully paid for the great part(Figure 2).

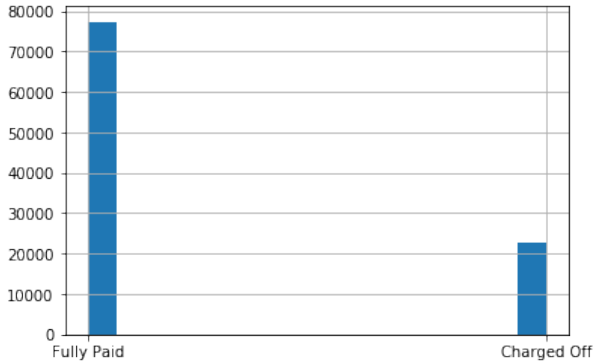


Figure 2: The majority of the client is solvent

Then, the majority of the clients work in the current job since more than 10 years, that is a sign of economic stability and only a small percentage of them had problem of bankruptcies. We also know that for most of the clients of the bank, the loan is useful to consolidate debit that already had but also to buy a new house, since a lot of them has a home mortgage. In Figure 3 we show some histogram that demonstrate this.

2.1 Data Preparation

As first thing, we consider useless for the analysis the attributes "Loan ID" and "Client ID", since they can't tell us general information. Then, we removed also "Months since last delinquent", because more than the half of the values were null, so we consider any substitution could be dangerous (see section 'Missing values'). Studying the attributes we noticed that in "Purpose", the value "Other" appeared twice ("Other" and "other"). So we modified it into "Other". The same thing happened in "Home Ownership", for the value "Home Mortgage"

2.1.1 Data Transformation

We did not consider necessary transform the dataset in the data preparation phase, but we did it to solve some specific task.
In clustering section(3.4), we solved the transactional clustering using the dataset containing only categorical attributes, that are 'Term', 'Years in current job', 'Home Ownership', 'Purpose' and 'Loan Status'. We had to codify these attributes to use the K-modes, assigning one label to each value of the attributes.
To solve the Frequent Pattern Mining (5) we discretized the categorical attributes in 10 bins, transforming the attributes 'Number of Credit Problems', 'Bankruptcies' and 'Tax Liens', casting their type and transforming them in strings. However, in general, to solve the other tasks we applied the **One Hot Encoding** to treat the categorical values.

2.1.2 Correlation

As shown in Figure 4, we detected a good correlation between "Bankruptcies" and "Number of credit problems". But what captured our attention was the correlation between "Number of credit problems" and "tax Liens".

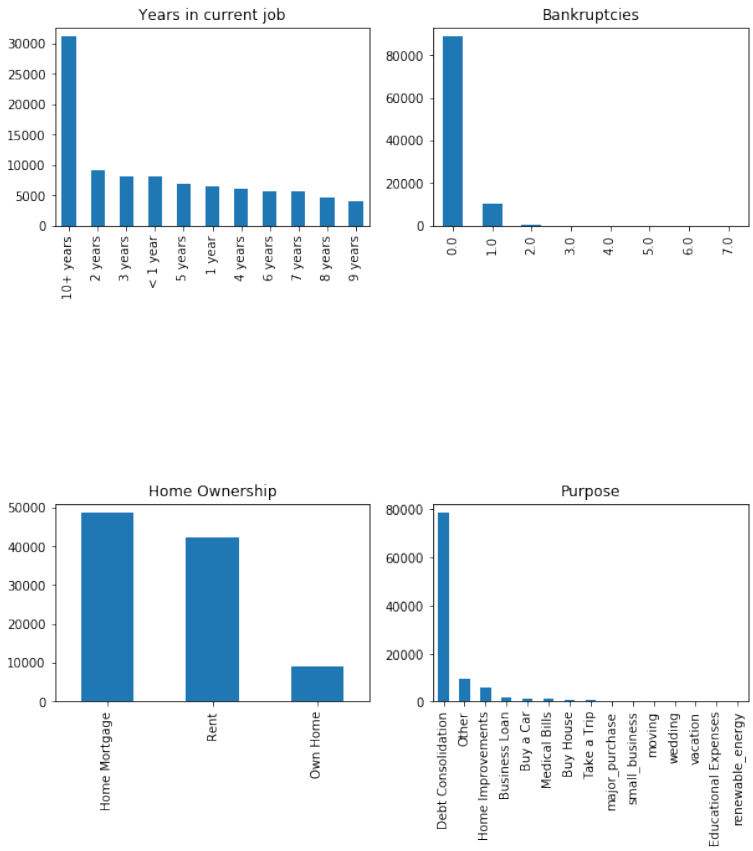


Figure 3: Distribution of the variables

2.1.3 Missing values detection

We immediately noticed that for every attributes there were 514 missing rows, so we eliminated them. Then, as we already said, for the attribute "Months since last delinquent" there are 53141 missing values, so we eliminated it. But there are still attributes for which there are null rows: Bankruptcies, for which there are still 204 missing rows, we decided to fill them with mode (0.0), after trying all the other methods, because it didn't bring any significant change in the statistics of the attribute; we did the same for "Years in current job", for which there were 4222 missing rows; for "Tax Liens" we fill the 10 missing rows with mode(0), since there are only integers, so the mean was not applicable; also for "maximum open credit" we used the mode, while for "Credit Score" and "Annual Income" we used the mean.

2.2 Handling outliers

Watching at the distribution of the attributes, it's evident how in every of them outliers can be found. This is obviously due to the fact that in the majority of the cases we are talking about money amount, that cause a lot of different values. But the same thing happens also in the other cases. We can visualize them in the boxplots, that underline the points that are not in the percentiles(Figure 5). For subsequent task we deleted the outliers using the IQR score and we can see the improvement comparing the boxplots obtained in figure 6

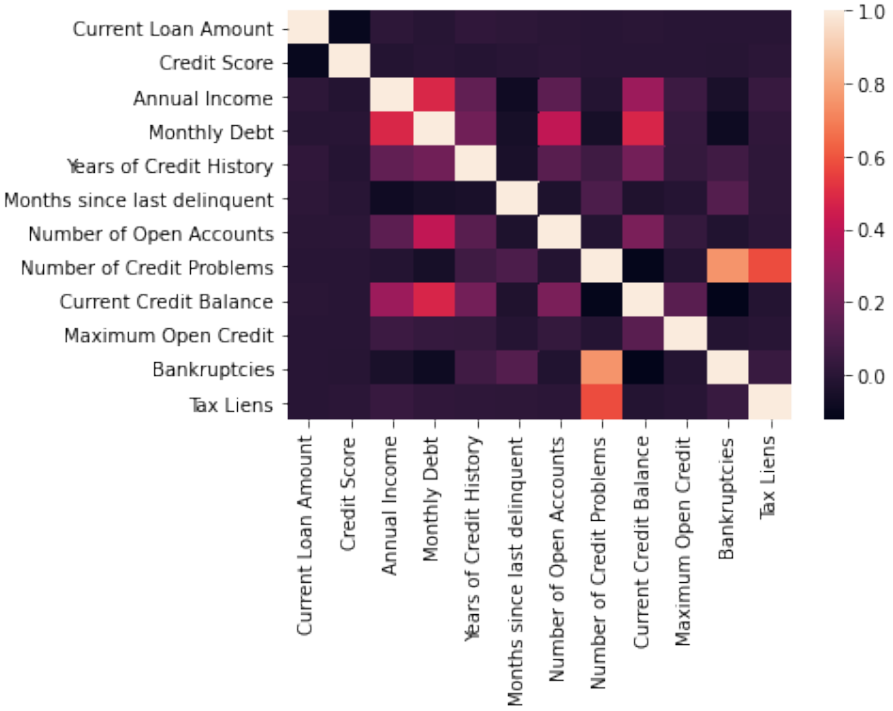


Figure 4: Heatmap for correlation

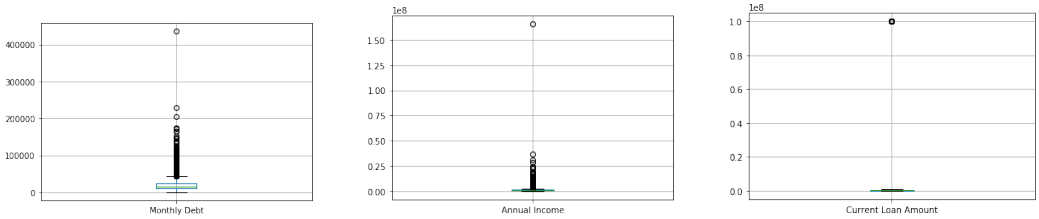


Figure 5: Boxplots obtained from different attributes on initial dataset

3 Clustering

We tried to clusterize our data with the techniques that we know, namely K-means, DBSCAN and hierarchical clustering. In particular, we tried different approach, proposing dataset differently prepared: we tested the complete dataset, the complete dataset without upper outliers, the dataset with only numerical attributes without outliers and finally the dataset with only categorical attributes.
In order to choose the parameters that we need to train the techniques of clustering we tried to respect the following rules:

- K-Means:
  - **K:** Find the best result with the Elbow method and the Silhouette method to obtain the optimal number of clusters. With the first we chose the K for which we can observe the maximum curvature (knee point), and with the second we chose K which have maximum Silhouette.

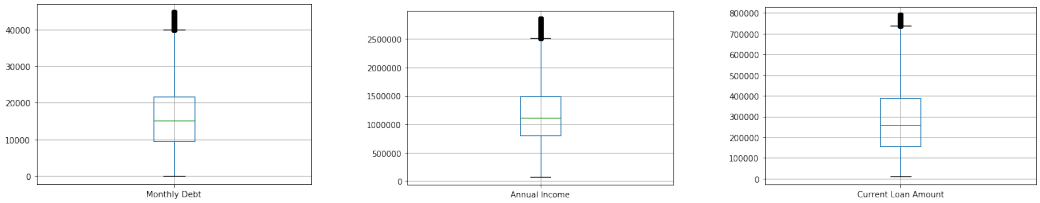


Figure 6: Boxplots obtained from different attributes on dataset without outliers

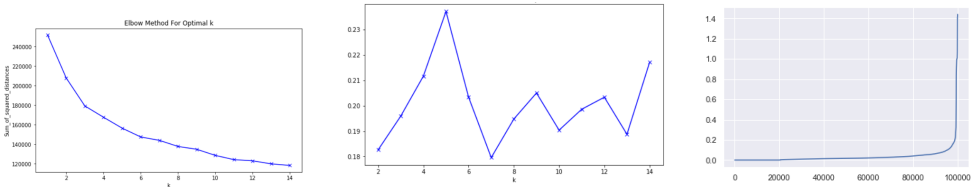
  - **Metrics:** Euclidean distance.

- DBSCAN:
  - **MinPts:** It is choosen as the double size of the used dataset.
  - **Eps:** After computing distances through the NearestNeighbor, we plotted them and chose as Eps the one near the maximum curvature.

- Hierarchical Clustering:
  - **Cut of Dendograms:** We chose to cut the dendograms where the branches are longer.
  - **Metrics:** Euclidean distance.

3.1 Complete Dataset

So as first try, we transformed all the categorical data in numeric one with One Hot Encoder. So our dataset is now composed by 43 attributes and 100 000 rows. We normalized the data with MinMaxScaler and then we applied some method to tune the parameters. In particular,we used the Elbow method(Figure 7a) and silhouette(Figure 7b).



(a) *Elbow method* (b) *Silhouette method* (c) *Knee method*

Figure 7: Choosing parameters

- **K-Means:** Looking at the graphs in figure 7 we chose as number of clusters the one with maximum silhouette (7b) that is the nearest to the knee (7a).
  Applying k-means with k = 5 we obtain an SSE equal to 156265.45 and silhouette of 0.236. The distribution of record through the 5 clusters is shown in figure 8.

- **DB-SCAN:** Considering as parameters MinPts = 86 and Eps = 0.16 (obtained from the knee method in figure 7c, choosen as 2*Dimensionality) and applying the DBSCAN we obtain a really good silhouette, equal to 0.44, but with 40 clusters where data are distributed in heterogeneous way. After making variuos tests we chose MinPts = 360

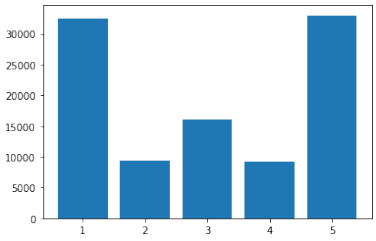
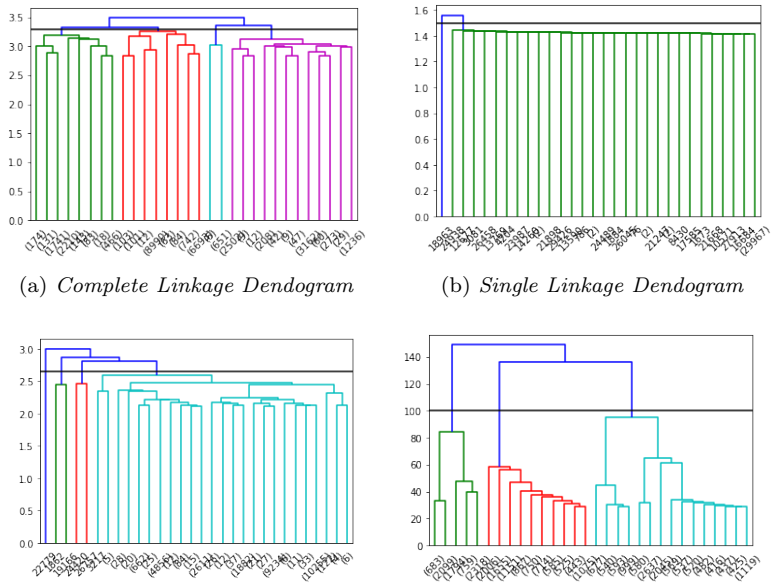


Figure 8: k-Means distribution

and Eps = 0.05, but it came out a really bad Silhouette(-0.22) with homogeneous distribution of the data on the 4 clusters.

- **Hierarchical Clustering:** In order to apply the different methods of hierarchical clustering we had to scale back dataset deleting the 70% of the records in random way. So we could apply the four techniques: Complete Linkage, Single Linkage, Group Average Linkage and the Ward's Method. In general, we cut the dendogram selecting distances in the longer branches but sometimes, to avoid having to much clusters we didn't follow this rule.
  In figure 9 are shown the dendogram of the four metods.



(a) *Complete Linkage Dendogram*

(b) *Single Linkage Dendogram*

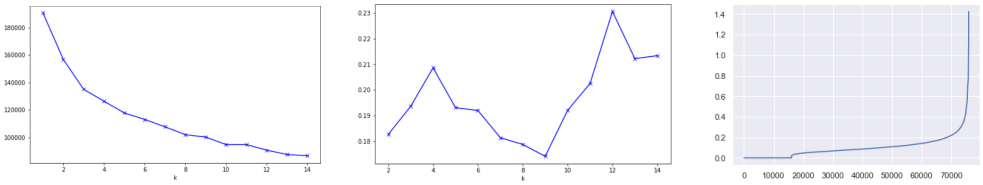
(c) *Group Average Linkage Dendogram*

(d) *Ward's Method Dendogram*

Figure 9: Dendograms applying different techniques on whole

3.2 Complete Dataset without upper outliers

As before, we used One Hot Encoding to deal with categorical attributes. Also, we deleted the upper outliers using the IQR score, obtaining a new dataset with dimension 75 800×43. Again, we normalized data with MinMaxScaler and, as it can be seen in (Figure 10), the results are different with respect to the previous case in which outliers were considered.



(a) *Elbow method* (b) *Silhouette method* (c) *Knee method*

Figure 10: Choosing parameters

- **K-Means:** Looking at the graphs in figure 10a and figure 10b the best choice seems to be 4 as clusters number. This parameters let us obtain SSE equal to 126381.72 and silhouette of 0.208. In figure 11 it is shown the distribution into the 4 clusters.

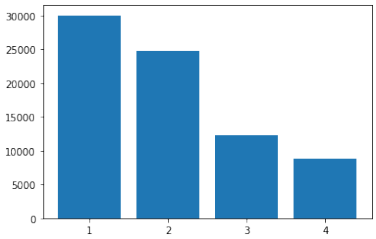
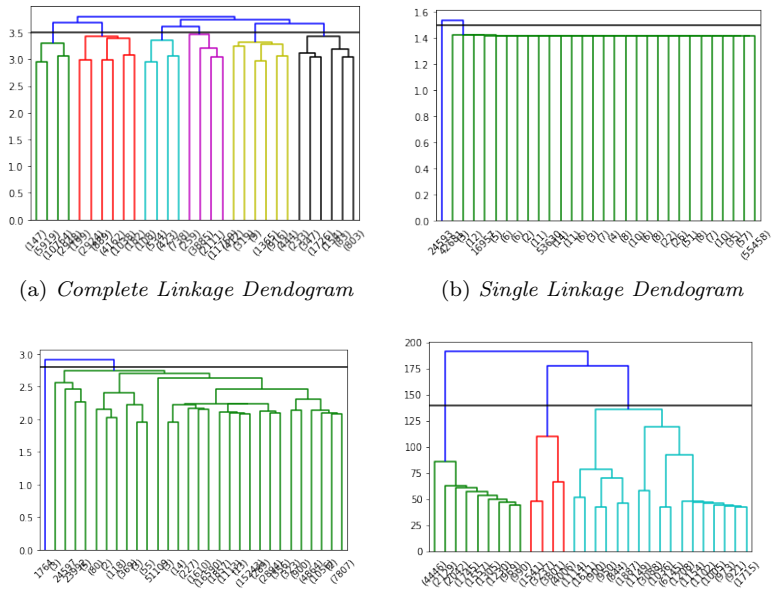


Figure 11: k-Means distribution

- **DBSCAN:** Following the general rule for MinPts choice, we selected 90 (2*dimensionality) as optimal value. For the Eps, we exploited the knee method in figure 10c, obtaing Eps equal to 0.3.
  The Silhouette value is 0.227, despite we obtain 129 clusters.

- **Hierarchical Clustering:** Also in this case, as in the previous application of hierarchical clustering, we had to scale back the dimensionality of the dataset, deleting 20000 records in random way. After normalizing data, we were able to build dendrograms using the techniques we know: Complete Linkage, Single Linkage, Group Average Linkage and the Ward's Method(Figure 12). In this case we followed the general rule for the cut of the dendogram.



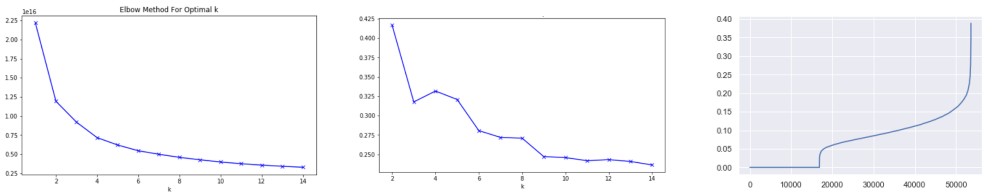
(a) *Complete Linkage Dendogram*

(b) *Single Linkage Dendogram*

(c) *Group Average Linkage Dendogram*

(d) *Ward's Method Dendogram*

Figure 12: Dendograms applying different techniques on the whole dataset without upper outliers



(a) *Elbow method*

(b) *Silhouette method*

(c) *Knee method*

Figure 13: Choosing parameters

3.3 Numerical Attributes without outliers

As further try, we deleted all the categorical attributes and outliers considering the IQR score, as we already saw in section "Handling outliers", obtaining a new dataset with dimension 53620×7. We then normalized the data and used different methods for parameter tuning, obtaining the results in figure 13

- **K-Means:** Using the Elbow method and silhouette method (Figure 13a e 13b) we could choose k = 2, thanks to which we obtained a SSE equal to 11657.56 and Silhouette of 0.305. The clusters contain respectively [33625, 19995] records.
- **DBSCAN:** From knee method, the best Eps seems to be 0.16 with Minpts = 14. Using this parameters we otbain too many clusters but heterogeneous, so trying between different values, it came out that with MinPts equal to 42 e we obtain two clusters well distributed, as shown in the histogram in figure 14.
- **Hierarchical Clustering:** Due to problems involving computing power and memory,

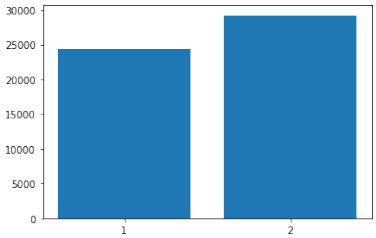
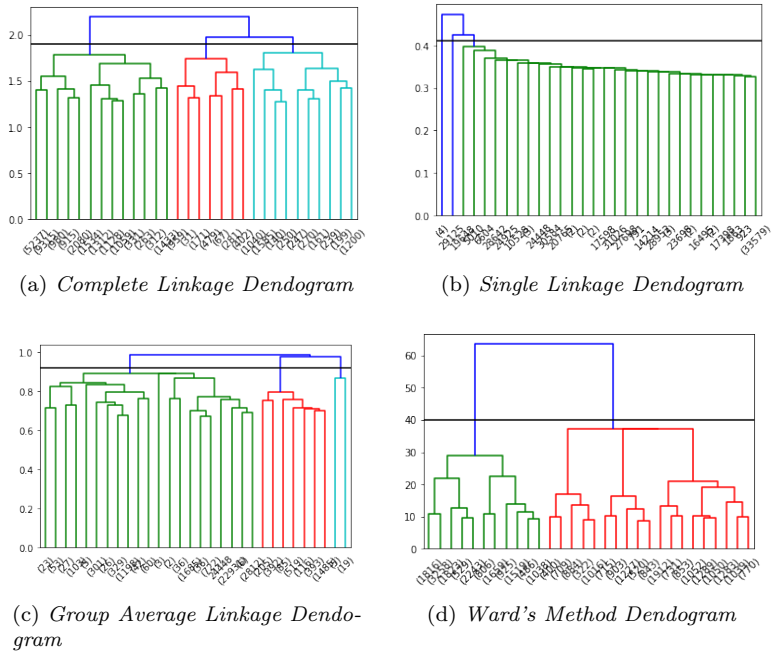


Figure 14: DBSCAN distribution

we had also in this case scale back the dataset dimensionality, deleting in random way 20000 records. After the normalization with MinMaxScaler, we could apply the 4 techniques of hierarchical clustering, using as metrics the euclidean distance, obtaining the dendrograms shown in figure 15.



(a) *Complete Linkage Dendogram*

(b) *Single Linkage Dendogram*

(c) *Group Average Linkage Dendogram*

(d) *Ward's Method Dendogram*

Figure 15: Dendrograms applying different techniques on dataset with numeric attributes deleting outliers

3.4 Categorical Attributes

Finally, we decided to clusterize a dataset containing only categorical data: to do so we assigned a label to them, that are Term, Years in current job, Home Ownership and Purpose. The techniques that we used so far, are not useful for this type of dataset, so we decided to use K-modes between the various existing techniques of transactional clustering.
Selecting k = 3 through the Elbow method in figure 16, we can see the composition of the 3 clusters thanks to the histograms shown in figure 17.

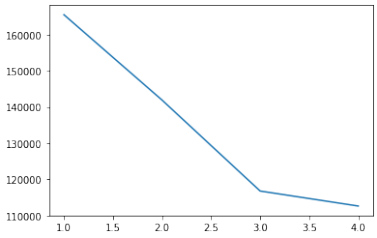
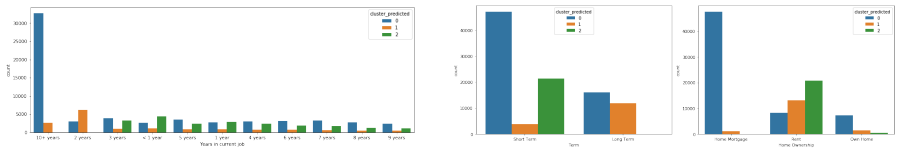
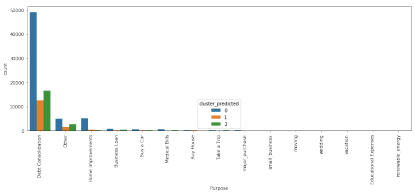


Figure 16: Elbow method



(a) *Years in Current Job distribution* (b) *Term and Home Ownership distributions*



(c) *Purpose distribution*

Figure 17: Distribution of the clusters obtained with K-Modes

3.5 Clustering Analysis

To analyse the results, it is shown the composition of the clusters obtained on different dataset. The following scatter plots show various features.
Starting with the complete dataset, we couldn't study its composition because of the many outliers. The situation become better in the dataset without upper outliers, in which, using k-means we can plot scatter plots on different attributes, even though this technique offers a bad separation of the data(Figure 19a).
The best results are found applying clustering on the dataset containing only numerica attributes and in which we deleted the outliers.
In fact, as we can see in figure 19b, the Ward method of the hierarchical clustering shows 3 well-separated clusters of the attribute 'Current Loan Amount', where each cluster indicates 3 different loan amount of the clients, respectively high, medium and low.
Also from DBSCAN we obtained a really good clustering: for example for the attribute 'Current Loan Amount', it can easily be recognized two different clusters, as shown in the scatter plot in figure 19c.
K-means clusterizes the dataset in two well separated clusters, where the first represents clients with low income and current credit balance, while the second one represents clients having high Annual Income and high current credit balance (figure 19d). This result is also observable looking centroids in figure 18.
Finally, we decided to analyse the composition of clusters using just categorical attributes of the dataset, using the k-modes. The result is shown in figure 17. We can notice that the first cluster corresponds to clients having a job since more than 10 years and also opened a

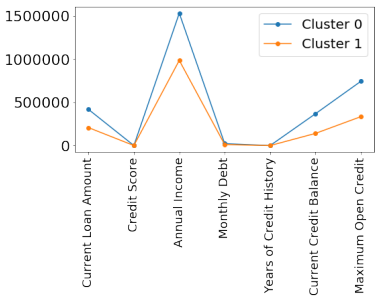
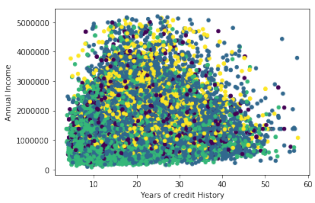
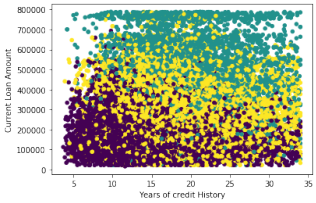


Figure 18: Centroids

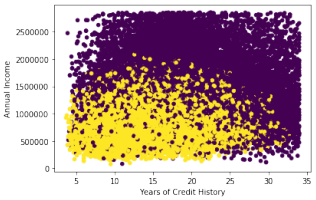
home mortgage. The second and third clusters represent clients who pay rent for the house where they live; plus, in the second cluster clients demands long term loans, in the third they ask for a short term loan. Then, we can see the distribution of data with respect to 'Purpose', noticing that all the clusters have in common the reason why they asked for a loan: debt consolidation.



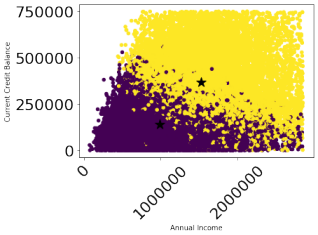
(a) *k-means on complete dataset without upper outliers*



(b) *Ward's method on numerical dataset without outliers*



(c) *DBSCAN on numerical dataset without outliers*



(d) *k-means on numerical dataset without outliers*

Figure 19: Scatter plot on different datasets and after applying clustering techniques.

4 Classification

We tried to classify our data using basic models, such as the **Decision Tree classifiers** ans **K-NearestNeighbor**, and an advanced one, the **Random Forest**. Training and testing the classifiers on the dataset created during the cluster section, we noticed that, not considering the upper outliers the performance get worst, so we applied the classification on the complete dataset and in the one with only categorical attributes.
In all cases we split the dataset in training and test set(70% e 30%) and used 'Loan Status' as class. For parameter tuning we used the Grid Search method and Cross Validation, for which we considered the training composed by validation set and training set.
Performances are shown in the table 1 and having an imbalanced dataset to the 77% we should compare the accuracy obtained with the ones of the the trivial case classifier.

4.1 Decision Tree

The method we used for parameter tuning agreed on the value of max depth using the complete dataset, while we obtained different results on the other case with transactional dataset. In fact, in the first case with Grid Search we obtain [min_samples_leaf = 1 ; min_samples_split = 2 ; max_depth = 1] and cross validation is a confirm for this choice, as we can see in figure 20; in the second case, we obtain [min_samples_leaf=5 ; min_samples_split=2 ; max_depth = 5] with Grid Search, but Cross validation suggests max_depth = 4 (Figure 21). However, training the model before with one parameter and then with the other, we decided randomly to leave the GridSearch parameters, because the results didn't change in both cases.

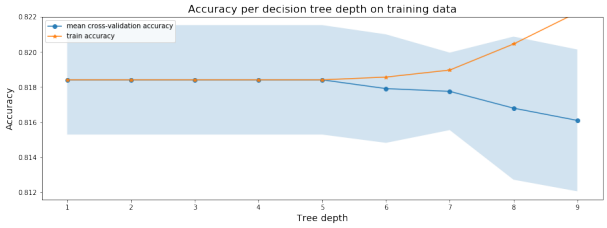


Figure 20: Cross Validation of decision tree on complete dataset

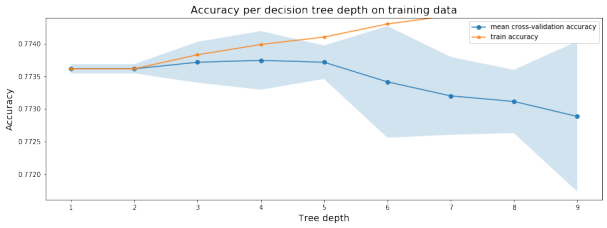


Figure 21: Cross Validation for Decision Tree on transactional clustering

From the table 1 it can be seen that the classifier has better performances when it is applied to the complete dataset with respect to the case of transactional dataset, and in the latter case we can see that the accuracy is equal to the accuracy of the trivial classifier. We plotted the decision tree in figure 22 and 23. We can see that the most important features is 'Credit Score' for complete dataset and 'Term' for transactional dataset, meaning that for

this features the Gini Index is the lowest.

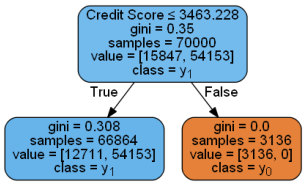


Figure 22: Decision Tree on complete dataset

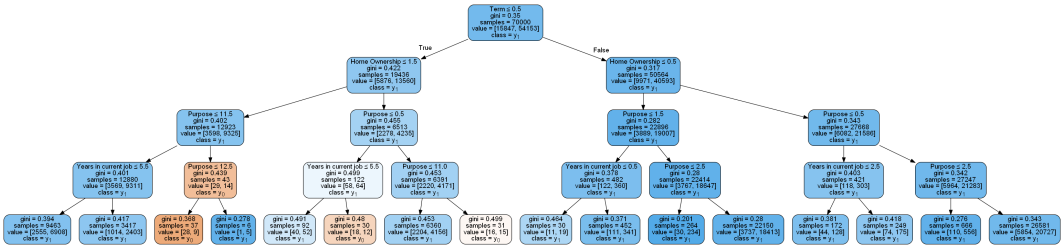


Figure 23: Decision Tree on transactional dataset

4.2 Random Forest

Also in this case we decided which parameters to use with GridSearch and Cross Validation, obtaining the following parameters, respectively on complete dataset and transactional one:

$$[max\_depth = 9; min\_samples\_leaf = 1; min\_samples\_split = 2; n\_estimators = 50]$$

$$[max\_depth = 4; min\_samples\_leaf = 1; min\_samples\_split = 2; n\_estimators = 150]$$

In this case we need one more parameter to estimates, that is the number of estimators, being the random forest an ensemble classifier.
As we can see in the table 1, There is a good change with respect to the previous classifier, in both cases (complete dataset and transactional one): improve of AUC on complete dataset, and in the other case the precision get better. Also in this case the classifier in consistent, because it has accuracy bigger than the trivial classifier, but just in the case with the complete dataset, because it losts consistence in the other case.

4.3 K-NearestNeighbor

For the last try, we trained and tested the K-NearestNeighbor classifier,for which we had to estimate the number of neighbors and the weight function used in prediction, always using GridSearch and cross validation. On the complete dataset, we obtained the following GridSearch results : [n_neighbors = 20 ; Weights = 'distance'] meanwhile on the transaction clustering: [n_neighbors = 17 ; Weights = 'uniform']. The parameters were confirmed by the Cross validation.
After training the classifiers, we tested them in the test set to evaluate their performances (Tab. 1). In both cases, we didn't obtain good results. In particular the classifier become inconsistent because the accuracy is the same as the trivial classifier. The only eception is

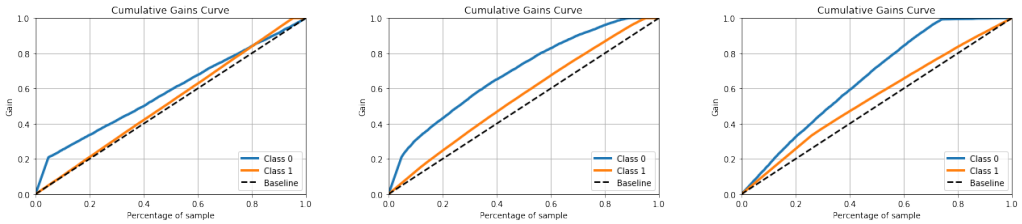
for the increasing AUC, that is bigger than the Decision tree AUC working in the complete dataset.

Classifiers	Dataset	Accuracy	Precision	Recall	F1-score	AUC
Decision Tree	Complete Dataset	0.82	0.85	0.82	0.77	0.60
	Transactional Dataset	0.77	0.73	0.77	0.68	0.58
Random Forest	Complete Dataset	0.82	0.85	0.82	0.77	0.75
	Transactional Dataset	0.77	0.77	0.77	0.68	0.58
K-NN	Complete Dataset	0.77	0.69	0.77	0.69	0.70
	Transactional Dataset	0.77	0.68	0.77	0.68	0.55

Table 1: Performances of classifiers of 2 datasets

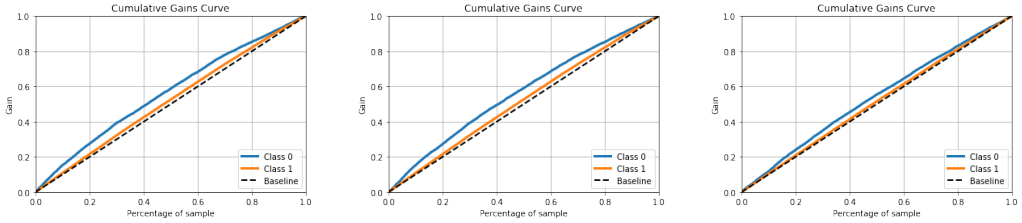
4.4 Classifiers Analysis

As we saw, the classifiers don't give a accurate performance. As further confirm, it can be seen in figura 24 and 25 where the Gains Cumulative are shown . In particular, the worst job is made on the class 1, that is the minority class.



(a) *Cumulative Gains Curve of Decision Tree classifier* (b) *Cumulative Gains Curve of Random Forest classifier* (c) *Cumulative Gains Curve of K-NN classifier*

Figure 24: Cumulative Gains Curve on Complete dataset



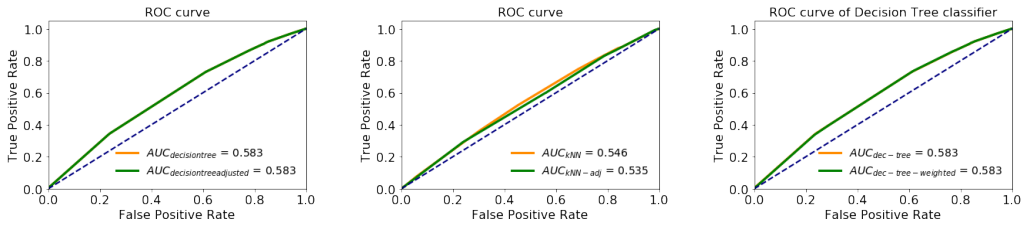
(a) *Cumulative Gains Curve of Decision Tree classifier* (b) *Cumulative Gains Curve of Random Forest classifier* (c) *Cumulative Gains Curve of K-NN classifier*

Figure 25: Cumulative Gains Curve on Transactional dataset

As we can see, the classifiers trained and tested on transactional dataset offer accuracy at most equal to trivial classifier, we tried to apply some techniques to handle imbalanced data, which work at algorithm level:

- **Adjust the decision threshold**
- **Adjust the class weight**

The first techniques was applicable only to KNN and decision tree classifier; the second, indeed, was applicable just to decision tree, being this one the only the assign a weight to classes. Then we tries to solve again the classification task and we noticed the performances were worst, excepted for the that keep to be the same in two cases over three(figure 26).



(a) *ROC curve of Decision Tree after adjusting the decision threshold*

(b) *ROC curve of K-NN after adjusting the decision threshold*

(c) *ROC curve of Decision Tree after adjusting class weight*

Figure 26: ROC Curve on Transactional dataset after handling imbalanced data

5 Frequent Pattern Mining

To solve this task we used the complete dataset. To find the rules, we replaced the missing values using the techniques saw in the 'Data Understanding' section. We then transformed the continuous attributes in categorical one, discretizing them in 10 bins. Going on with the work, the dataset containing missing values will be taken again to replace missing values through the rules we found.

5.1 Frequent Patterns Extraction

We computed the frequent item sets with Apriori algohorithm and we ordered them basing on the support. It can be noticed that the frequent itemset with higher support is ['No bankruptcies', '0.0_tax Liens'] with support = 89,12 %. The second frequent itemset with respect to support is ['(10802.0, 10009721.7)_LoanAmount', '0.0_tax Liens'] with support = 88,00 %. So, what we can notice is that when tax liens is 0(so the client has not tax liens), usually the bankrupt is never declared and the current loan amount is low.
We visualize the trend of the number of patterns on three different types (frequent, closed, maximal) in figure 27.

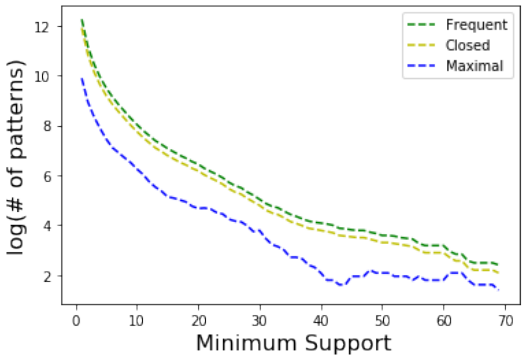
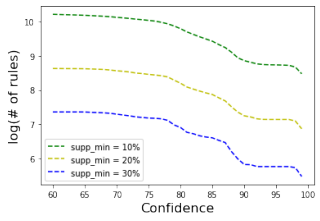


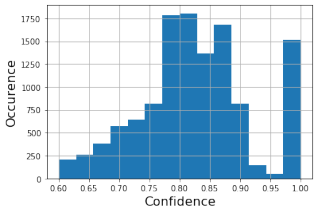
Figure 27: Number of different types of item sets as the support change

5.2 Association Rules Extraction

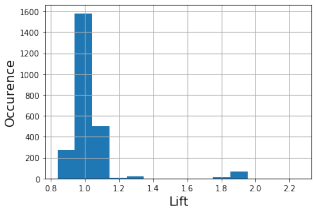
Using the Apriori algorithm we extracted the association rules, trying to analyse their behaviour, while changing some parameter such as confidence and lift.
In figure 28a, we can see the trend of the number of rules as the confidence change for three different values of support and we can notice that the number of rules decrease slightly when the confidence increases.
Finally, from the histograms of rules' confidence and lift we can see that the major part of rules have a lift that has value about 1 (figure 28b), and we can see the distribution of rules with respect to the confidence. Most of the rules have 80% of confidence (figure28c).



(a) *Numero di association rules al variare della confidence*



(b) *Histogram of rules' confidence*



(c) *Histogram of rules' lift*

Figure 28: Analysis of association rules

5.3 Applications of Association Rules

The initial datased contained some missing values. At the beginning, we replaced them using some general rule, but now, knowing the association rules, we can exploit them to better replace missing values. We set the parameters of Apriori as follows:

- Minimum support of an item set = 15
- Minimum number of items per item set = 2
- Type of frequent item sets to find = rules
- Minimum confidence of an association rule = 60

In (figure 29a), we can see which were the missing values. We could obtain some significant rules for the attributes 'Bankrupticies' and 'Tax liens', but not for 'Years in current job', for which we can't say anything. On the other hand, for 'Credit Score' and 'Annual Income' we found some rule, and we also noticed that every rule that has 'Credit Score' as consequent, has as antecedent a value of 'Annual Income' and viceversa. We also noticed that for every record, these attributes are both present or both missing, so we couldn't exploit any rule to replace them[1].

```
Credit Score            19154
Annual Income           19154
Years in current job     4222
Maximum Open Credit         2
Bankruptcies              204
Tax Liens                  10
dtype: int64
```

(a) *Missing values on initial dataset*

```
Credit Score            19154
Annual Income           19154
Years in current job     4222
Maximum Open Credit         2
Bankruptcies                8
Tax Liens                   2
dtype: int64
```

(b) *Missing values after replacing with association rules*

Figure 29: Missing values

Finally, using the list of rules we already computed, we used the most meaningful ones to predict the target variable('Loan Status') and evaluate the accuracy. We ordered the rules to have before the ones with higher confidence and when two rules have same confidence, come before the ones having higher support.
So, we split the dataset in training and test and used the 14 rules with higher confidence and minimum support 15% to classifie the test set and evaluate the accuracy comparing the predicted values with the real ones.
In figure 30, we can see the performance of the classifier built with the association rules. Despite the accuracy is not excellent, it's higher than the trivial classifiers' one (77%).

	precision	recall	f1-score	support
Charged Off	0.35	0.12	0.17	5234
Fully Paid	0.82	0.95	0.88	22275
accuracy			0.79	27509
macro avg	0.59	0.53	0.53	27509
weighted avg	0.73	0.79	0.75	27509

Figure 30: Performance of decision rules classifier

[1]Actually, we noticed that a rule was found but it didn't respect our conditions. This happens if we change the initial parameters, in particular, if we set the confidence at 54%. However, it didn't worth it to change it because also the support was too low (7%)