Institut de Science Financière et d'Assurances

Machine Learning for Credit Scoring

Lucas BLANCHETON, Alexandre BONICEL, Gabriel OLLIER

April 6, 2022

Contents

1 Introduction

1.1 Presentation of the subject

Bankers are the people who make loans, they are responsible for something very important and useful for the economy. This money creation process lets the economy works by having something which makes exchanges possible. Also, it lets people make investments, they can buy their houses for instance, and helps for the development of the economy throughout the country or even worldwide.

Yet bankers can face a problem: the default risk which is a real threat for them. They can make loans but if they are not paid back by customers who make default, they can lose a huge amount of money. It can even lead to bankruptcy in the worst case. The aim of the bank is not to lose money but to make the maximum profit like most companies. Risk default is a crucial issue for each bank. If they impose drastic conditions on their borrowers, it is for a reason, they do not want people to struggle too much with their loans and they do not want to lose money too.

To avoid losing money, they do not lend to everybody. People need to have a good bank record otherwise they are not allowed to borrow money from the bank. To assess the risk of each borrower, banks use statistical methods to attribute a grade to each loan depending on the borrower. The grades show if people are more likely to respect their engagements and if they will use their credits properly. According to those grades, people cannot obtain the same credits. Banks make loans with different interest rates according to the default risk of each borrower. On his side regarding the IFRS 9 regulation which is an international regulation relative to bank liabilities, provisions are made to counter the default risk. Banks make money reserves so if a customer makes a default, they can absorb the shock but, in this case, they lose money.

1.2 Goal of the project

Now bankers know that they need to grade borrowers. So, our new issue is to know how to grade in the best way to avoid problems. Grades are not everything because unexpected events can happen but most of the time, they represent the situation of each borrower properly.

To find a solution to that problem we will work on the grading system of LendingClub. It is an American peer-to-peer lending company that uses grades from A to G. The best one is A and the worst one is G. Grades are not standard, we can find other systems.

Our goal is to make a machine learning model which could attribute a grade using some information about the borrower. To do that we will use a data set coming from LendingClub and the idea would be to do a regression on the rel-

evant characteristics of people.

We will explain to you what machine learning is and why is it interesting for banks and companies to use it nowadays. Then we will build a model to grade borrowers thanks to Machine Learning.

2 Team Presentation

We all three are attending courses at ISFA school in Lyon to become actuaries.
We are doing our first year of master's.

Bonicel Alexandre

Blancheton Lucas

Ollier Gabriel

3 Challenges of the Subject

3.1 The Importance of Credit Scoring

As we said, the main risk for banks is the default risk and they want to limit it. To do that they have several possibilities. For instance, they can apply a higher interest rate for people with more risks. Also, they can refuse to lend money to someone.

Even if it can seem a bit rude to select "good" borrowers and leave "bad" borrowers, it is important to do it. Otherwise, it can be a catastrophe. The best example is the 2008 Sub-primes Crisis.

In the 2000s in the USA, bankers lent money to everyone without considering the default risk. They were paid by commission, so it was very attractive for them to sell as many mortgages as possible. Americans lent a lot of money to buy their own houses and the huge demand made the prices exponentially increase. It was not a problem as getting a loan was so easy. Unfortunately, this was Utopian.

When the loans were not paid back, people had to sell their houses but as they were a lot in that case, the prices fell. That means that even though they sold their houses they still had not enough money to reimburse the loan.
People in this horrible situation were so many that the default rate made Lehman Brothers Bank goes bankrupt. Then one of the worst worldwide financial crisis ever followed and we still are recovering from it in 2022.

We hope that it convinced you about the importance of managing default risk.

So, banks select people to balance their default risk, but we face another problem. Are we sure that banks select people with objective criteria? Indeed, they can have biases like knowing the persons, thinking people are trustable according to their looking, etc... It seems wrong to act like that obviously but, bankers are human and can have cognitive bias.

The best response is grading people. It sounds a bit like the favorite way of processing of a famous communist country, but it reminds to be the safest way to avoid a financial crisis. It offers objective criteria to manage the default risk.

3.2 The Evolution of Machine Learning

We live in an Information (Data) Economy. Every physical or digital transaction generates some data. This trend will continue to grow.

What does it mean in some numbers?

In 2022 there are about 40 zettabytes of data.

• 1 ZB = 109 Terabytes
• If printed in textbooks, the stack of textbooks would reach Pluto and back 20,000 times.

We need the proper tools to make sense of all this data. The best answer is **Data Mining**. It is the use of mathematical methods (regression, clustering, classification, deep learning) to extract patterns and actionable information from a large amount of data (Big Data). In Data Mining there are two main branches which are **Visualization** and **Machine Learning**.

First, let's explain what is **Data Visualization**. It is the act to transform rough data into graphics. This is a good way to have a quick look at the information and present it to others. It is also a strong way to find knowledge from the data.
An example will show that. We can imagine that we have Data with two features: X and Y. With a simple table, we cannot say if X and Y are correlated. We can plot Y against X to check for that. If they are correlated, it will be clear. We can almost estimate how much they are correlated.

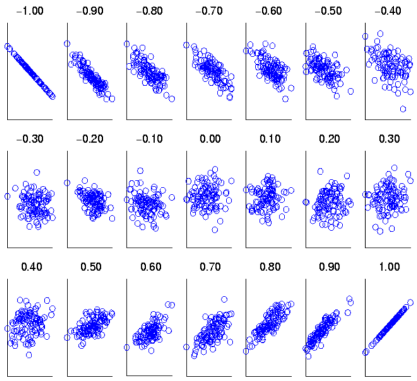

Figure 1: Y plotted against X
Source : HEC Lausanne courses

In the figure, we show twenty-one different graphics of Y plotted against X. Of course, each different graphic does not represent the same data set. They all show how the graphic would look like depending on the correlation between X and Y. The first plot shows a perfect negative correlation and the last one shows a perfect positive correlation. Both have a perfect line! The other ones show different degrees of correlation.
It is easy to see if X and Y are correlated with such a method.

Data Visualization lets us print some statistics through other methods such as boxplots, histograms, bar charts, and heatmaps. The advantages are the discovery of patterns, clusters, or gasps but also the easy communication. It helps to convey a message.

Now let's explain the most important part for us: **Machine Learning**. Briefly, Machine learning is the science of making computers learn and act like humans by feeding data and information without explicitly being programmed. It is a branch of artificial intelligence (IA).

What does it mean?

Giving some information to the computer, it can predict results on data. We will explain that with an example of images recognition.

If we have some images, we cannot write a code that recognizes if something is a car. This is so complicated, almost impossible. Instead, we can write a very generic program that learns by examples. We would give it many pictures of cars and tell it they are cars. Then when we give it an unknown picture it could compare with those it has. In such a problem it would compute the probability of having a car according to the similarities. If the likelihood is above 50% it would say that it is a car.
Then it is possible to recognize or **predict** anything (car, apple, the stock price will increase...)

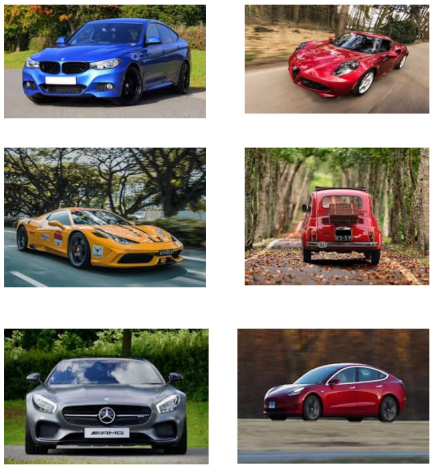

Figure 2: Images of Cars
Source : HEC Lausanne courses

In our case, we will give relevant information about borrowers and their corresponding grades to the computer. Then it is going to build a model which can predict grades when we give it only information about borrowers.

We explain what are Data Mining and Machine Learning but why are they so fashionable now?

Machine Learning is used a lot nowadays thanks to the confluence of several technical advances.
• **Storage** : Disk densities are huge today. A \$100 disk today has $1,000,000$ times more capacity per dollar than the disks 30 years ago.
• **Internet** : Data can be transferred easily between collection, storage, and use.
• **Algorithms** : Advanced algorithms from machine learning, pattern recognition, and applied statistics have become mature enough for mainstream use.
• **Computing power** : Laptops are more powerful than the supercomputers of yesteryear.

All of these technological advances are essential for the success of Machine Learning.

Yet there are still limits to Machine Learning. It is kind of a black box somehow. We give some information, and the machine returns us some results but we do not always know the functioning of algorithms that are behind the results. Nevertheless, it seems to give more accurate results than deterministic methods. Consequently, more and more people are interested in this way of processing and solving some problems and companies are using it more and more.

Also, we can add that it is easy to do it now because a lot of algorithms have been developed by many people who shared their works. We can work on many languages too like Python, R, or SAS. Everybody can use it.

4 Theory of Machine Learning for Credit Scoring

4.1 Importance of loan

In this economic era, loans have become very important. Governments can borrow from individuals, and individuals can as well borrow from governments. Whichever the case, borrowing, and lending of money are essential to the economy of a country.

During an inflation period, loans hold an important position. Inflation refers to a state in which there is a general increase in the prices of goods and services in the economy. As a result, the purchasing power of consumers decreases. Inflation sets in when there is an increase in credit and increases the supply of money in the economy.

How can this situation be controlled? The government, through the central bank, will increase the interest rates on loans and deposits. With high-interest rates on loans, individuals cannot borrow. Instead, the high rates favor saving, which reduces the amount of money in circulation. As a result, inflation decreases.

In a deflation period which is the opposite of inflation, the prices significantly drop, and this hurts the economy. This time the government will do the opposite operation by reducing the interest rates on loans and this will stimulate the economy.

Loans also have significant importance in the investment field. Indeed, investment debt leads to the production of goods and services that may not occur if loans don't exist.

4.2 Functioning of Machine Learning

We introduced Machine Learning and explained its evolution. In that part, we are going to speak about it in our context.

To begin with, let's see some terminologies. Machine Learning deals with **Data** (simple, isolated facts) and **Information** (data in context) to make **Knowledge** (interpreted information). In our project, **Data** are the different values that we can find in our data-set. **Information** is the whole data-set where each value is associated with others and the **Knowledge** is the grades we want to give to borrowers according to their information.

To do that we will make models with Python. They will be **classification models** as they will predict a categorical value (the grade). They should not be confused with **regression models** which predict a numeric value (quantity).

Classification is an instance of a **supervised learning** algorithm which means the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labeled training data consisting of a set of training examples. Data can be words, numbers, or even pictures.

We can imagine that we want to build a model which predicts if the pet on a picture is a cat or dog.

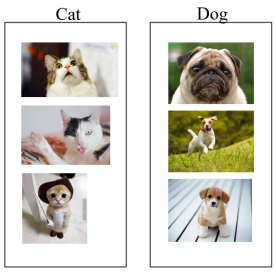

Figure 3: Data-set to build the model
Source : HEC Lausanne courses

We begin by giving many images like those on the left to the computer. For each one, we specify if it is a cat or a dog. Then the machine is going to find similarities between dogs' pictures and similarities for cats' pictures. It is supervised learning because we give examples to the program. When the model is finished, we can give a new image without a label. Thanks to the characteristics found the machine could compute the likelihood that it is a dog or a cat. According to the highest probability, it will find what it is. Of course, this application does not seem to be useful, but it is how Google image works so we use it a lot.

Several types of models exist. The ones that we will use are **logistic regression**, **K Nearest Neighbors**, **decision tree** and **random forest**. We will explain them in the part **Methodology of the project**.

Building a model is well but we need to know how good it works. To that extent, it exists some specific tools for classification models. As we do not predict numerical values, we cannot compute the mean squared error which is the difference (quantity) between predicted values and actual values. The two main tools are the **accuracy** and the **confusion matrix**.

The accuracy lets us know how likely a prediction is good with our model. The calculus is very intuitive.

$$Accuracy = \frac{\#correctpredictions}{\#totalpredictions}$$

Of course, the goal is to have 1. In reality, it is impossible to have exactly 1, a model cannot be perfect. So, the real goal is to do better than random. Let's explain what it means with an example.

We can imagine that we build a model with four possible outcomes. We do the hypothesis that they are equally distributed in the sample. If we randomly choose one outcome we find the good one with a probability of 0.25 ($\frac{1}{\#outcomes}$).

So, in this case, if our accuracy is under 0.25, we do worse than random which is ashamed. The goal is not to have 1 but to have more than 0.25.

The more different possible outcomes we have, the smaller the target accuracy is. That is why if we have very high accuracy with several classes, it can show that the model is false.

In our project, we have 7 possible outcomes (the 7 grades) so our goal is to have an accuracy greater than 0.14 ($\frac{1}{7}$).

If we have only two outcomes, accuracy is enough to know how works our model but when we have more classes, we want to have a more precise tool. Sometimes we would like to understand in which cases we make the errors. This is where the confusion matrix helps.

It is a matrix that shows how many right predictions we have for each outcome. It also shows in which category outcomes are wrongly predicted. Again, let's see it with an example.

In this example, we have a population of 100 people. Each one can have a **cold** (20 persons), a **flu** (10 persons) or be **healthy** (70 persons). There are three possible outcomes. A model gives us the following confusion matrix :

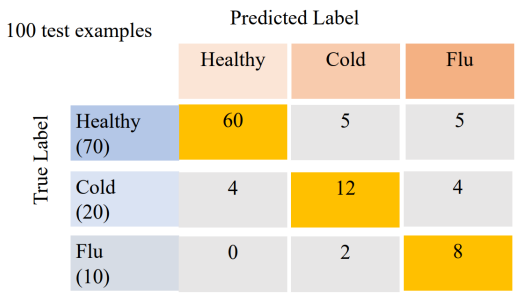

Figure 4: Confusion Matrix Example
Source : HEC Lausanne courses

We can see that we have predicted labels for the columns and true labels for the rows. Let's take a closer look at the first row, it is the true label **Healthy**. In the first column (predicted value healthy), there is the number 60 so the model rightly predicted "healthy" for 60 people. Then there are 5 for the column "cold" and 5 for the column "flu". The model predicted "cold" for 5 healthy people and "flu" for 5 other healthy people. The total of the row is 70, we have all our true healthy people.

The goal is to have a diagonal matrix, it would show that every outcome is well predicted. The confusion matrix is a good way to find which label has a problem.

5 Methodology of the project

5.1 Data cleaning

At the beginning of the project, we had a dataset taken from Kaggle and we have done several changes on it to make it workable for machine learning algorithms. We could not probably have had as good results as we obtained without data cleaning. In general, data cleaning is 80% of the work for a data scientist.

We want to have a dataset on which we could work without filling it.
The first step is to remove variables that have more than 10 percent missing values. It is an arbitrary choice, we do not want to remove too many variables. In the same way, we remove all rows with missing values.
At the end of this part, we have a dataset without any missing values, which is a good starting point.

The second step is to balance our dataset. We want to have the same base rate for each grade from **A** to **G**. It means that we need to have the same number of rows for each grade. Our reference is the **G** one because the number of borrowers with this grade is the minimal number (3119).
It is very important to do it, otherwise if a grade is overrepresented it would skew our algorithm outputs.

In the third step, we remove useless variables without any impact on the target variables. For example, the member id of each borrower is not linked at all with the grade he has.
Furthermore, on the contrary, some variables have a strong link with the target variable, it is the case of the subgrade. If the subgrade is **B2** we already know that the grade associated is **B**. So, we have to remove those variables too.
We also study the correlation between numerical variables using a correlation matrix. We remove variables that are too correlated to avoid issues for machine learning methods such as multicollinearity. For instance, if three variables are correlated altogether, we need to keep only one of them.

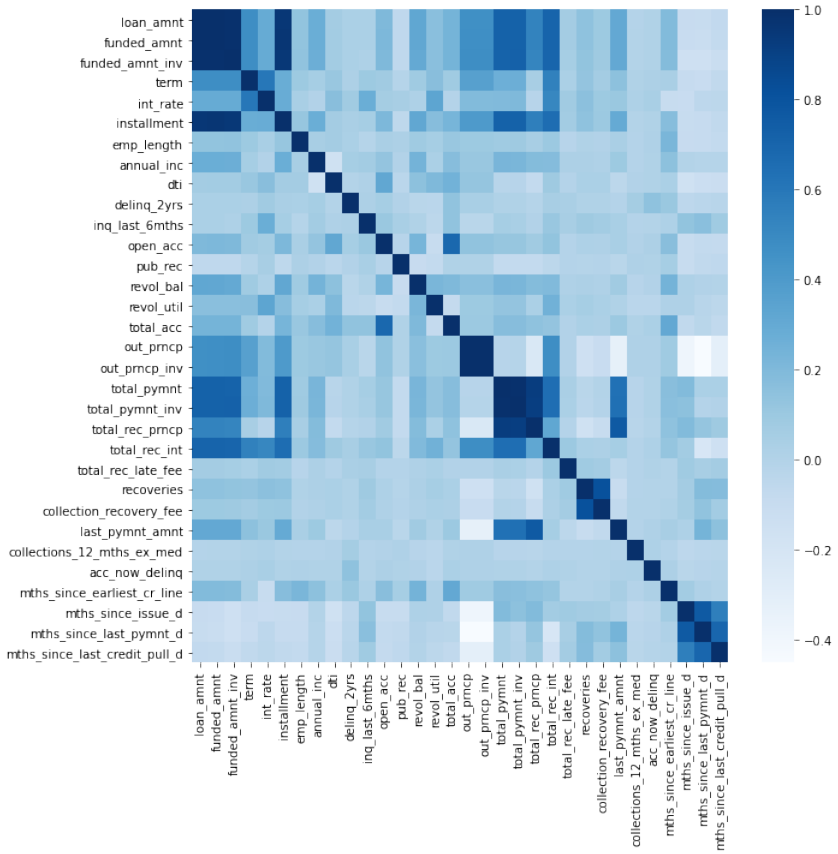

Figure 5: Correlation matrix
Source : Our Python code

Once all this is done, we modify the format of useful dates in our dataset to be able to use them as numbers and not as strings. It is the case of the "term" variable which has the format "36 months". After the operation, it is only written 36 in the dataset.

Then we factorize our categorical variables to only have numbers for the logistic regression. We normalize our numerical variables too.

After all those operations on our dataset, we are ready to use machine learning methods to predict the grade associated with each customer.

5.2 Classification

Now we have a cleaned dataset so we can do the classification models. We do four different models: **logistic regression**, **K Nearest Neighbors**, **decision**

tree and **random forest**. In this section, we are going to mathematically explain them.

Logistic Regression :

Regression is an instance of a supervised learning algorithm.

We are given a set of features for some objects/entities (customers, products, etc) and the category of what we want to predict. In our case, we are given bank information regarding customers, and we would like to predict their grades.

In our example we need to distinguish between many categories which are represented by the grades from A to G, it is a multi-class classification. However, we will explain the method when the target variable is binary. Then it can be extended to multi-class classification.

Logistic regression is a statistical technique that is commonly used in modeling a dichotomous dependent variable. This means that the dependent variable is binary containing only two distinct values, often 0 and 1 but there is no importance. The logistic regression function which is also known as the sigmoid is shown in the following equation:

$$p_i = \frac{1}{1 + e^(\alpha + \beta_1 x_{i_1} + \beta_2 x_{i_2} + ... + \beta_m x_{i_m})}$$

where p_i is the probability that $y_i = 1$, which varies from 0 to 1, α is the intercept of the model, β_j is the coefficient of the model for each explanatory variable, x_{i_j}(j = 1, 2, ..., m). As our explanatory variables are qualitative, we need to factorize them to be able to compute each probability.
If $p_i < 0.5$ then we can say that the value of our target variable is 0, otherwise, it is 1.
We end up with a function (green line) that cut the group into two groups, the one with p < 0.5 and the other with p ≥ 0.5

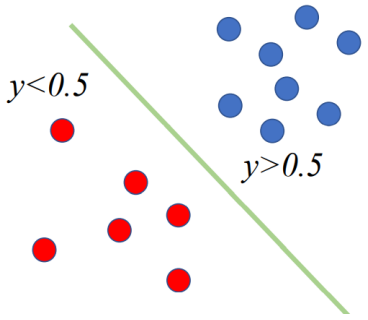

Figure 6: Binary classification
Source : HEC Lausanne courses

K Nearest Neighbors :

KNN is also an intuitive supervised classification algorithm.
That technique has approximately the same approach as the logistic regression. Indeed, we have again the same goal to predict their grades with a set of features and a target variable (the variable that we want to predict). Again we will explain the method when the target variable is binary but for multi-class classification, it is the same logic.

K Nearest Neighbors method aims at classifying target points (our target class) according to their distances from points constituting a training sample (i.e. whose class is known).

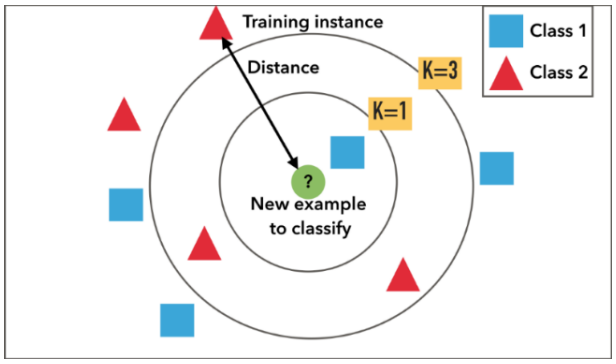

Figure 7: Example of Knn
Source : Medium

In this example, we must find the class of the green point between the **blue** class and the **red** one.

In the first step, we must choose a value for K, the number of nearest neighbors. Here in the example if we choose 3 neighbors the algorithm looks after just the 3 nearest points by using a euclidean distance.

Among these K neighbors, count the number of points belonging to each category. Assign the new point to the most present category among these K neighbors.

In the example, if we choose 3 for the K value, the algorithm will attribute the class **red** for the green point.

Decision Tree :

Decision Tree is a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piece-wise constant approximation.

Here is an example of a decision tree in medicine :

With this (real) decision tree, doctors decide if someone that arrived in the emergency room should be directed to the coronary care unit.

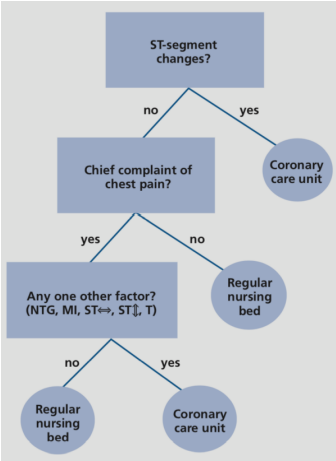

Figure 8: Example of Decision Tree
Source : HEC Lausanne courses

Out of many attributes, we have to decide on which attribute to split the data, so that it leads to the "purest" sub-nodes. We can do this using several metrics as the classification error. The aim of choosing the way to split trees is to minimize the classification error. In other words, we have to know how to assign each people to the right group.

In our case, we have to know what is the profile of the borrower who is going to get attributed a nice grade as **A** or **B** and the profile of the "bad" borrowers who are going to have a grade **F** or **G**.

There are some advantages to using Decision Trees :

- It can be easily visualized and interpreted. Looking at the Decision Tree we can see the nodes and the way people are assigned to categories.
- There is no need to normalize or scale features.
- It can work with all kinds of data types (binary, categorical, continuous)

The negative point of Decision Tree is that it is easy to overfit and so it will be adapted to your dataset only and will not predict results properly for another random dataset.

Random Forest :

Random Forest is what is called an ensemble method, i.e. it "puts together" or combines results to obtain a superb final result.

This method is an extension of the previous approach of Decision Tree. Simply because the random forest method combines the different results of the decision trees that compose it. Random Forest can be composed of several tens or even hundreds of trees, the number of trees is a parameter that is generally adjusted by cross-validation.

Each tree is trained on a subset of the dataset and gives a result (yes or no in the case of the coronary care unit example). The results of all the decision trees are then combined to give a final answer. Each tree "votes" (yes or no) and the final answer is the one with the majority of votes.

In this way, we build a robust model from several models that are not necessarily as robust.

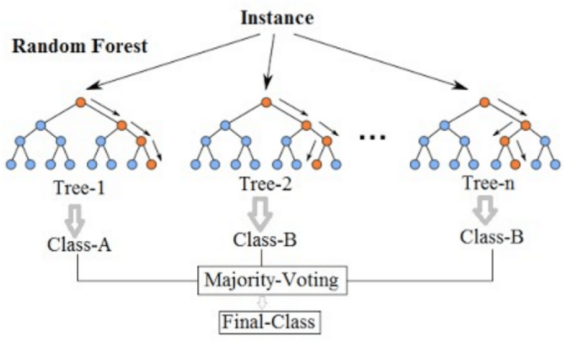

Figure 9: Example of Random Forest
Source : Datascientest

Here is an example of a random forest that predicts a class with multiple decision trees and in the end gives us a result by averaging the results.

6 Results

We use four different classification models and we will compare them. To do it we will use the **Accuracy** and the **Confusion Matrix** which we explained before.

First, we did our four models and we get those accuracies : Decision Tree has a very high accuracy (0.98). It is not realistic according to what we said before. Our target was 14%. It must be over-fitted on the data, which means that we have a very high accuracy but only on our data. If we take another dataset, the prediction would be less good. Random Forest lets us do not over-fit the data and its accuracy is lower. It is normal but its accuracy reminds too high (0.91). It is possible that we have a **perfectly correlated variable** and we should remove it otherwise it would be cheating.

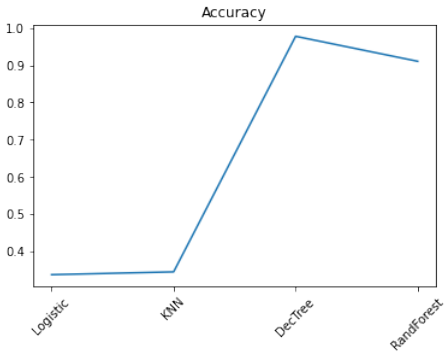

Figure 10: Comparison of accuracy
Source : Our Python Code

After taking a look at our variables, it appears that the **interest rates** could be this too correlated variable. Indeed it is calculated according to the grades. We decided to do the models again without the interest rate.

Now we are going to show the confusion matrix and the accuracy for each model, then we will compare them.

Logistic Regression :

With this model, we get an accuracy of 30%. It is quite good because our goal is to do better than 14% accuracy. However, if we look at the confusion matrix, the result is disappointing. Our matrix is not diagonal at all. In fact we predict almost only two values : **A** and **G**. We have almost one half which is predicted as a **A** and the other as a **G**. As almost all grades **A** and **G** are well predict, it is normal to have an accuracy of about $\frac{2}{7} \approx 30\%$. The accuracy is falsely good. Logistic Regression is

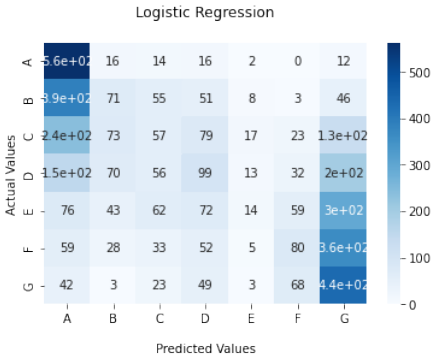

Figure 11: Confusion Matrix
Source : Our Python Code

not a pertinent model for our project.

K Nearest Neighbors :

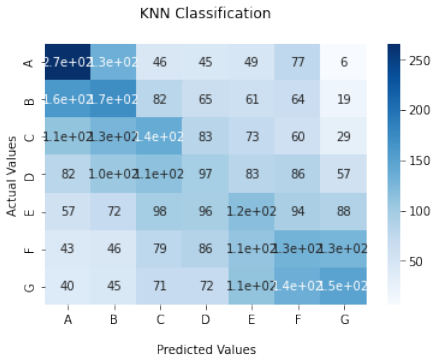

Figure 12: Confusion Matrix
Source : Our Python Code

This model gives us an accuracy of 25%. It is less than Logistic Regression and we could think that it is not a better method. Yet the confusion matrix seems to be more diagonal and the model predicts more than only **A** and **G**. It seems to be more interesting for our problem. The middle classes remind not predicted so the model is objectively not that pertinent but it is less wrong than the previous one even if the accuracy is lower. We can observe an amelioration but it is still not a right model for our project.

Decision Tree :

Decision Tree classification model gives an accuracy of 39%. It is better than the two other methods (Logistic Regression and KNN Classification) but what about the confusion matrix. Actually, it has a satisfying shape, we can see a darker blue diagonal, and this time every outcome is predicted in the model. The errors are mostly between close values. For instance, many **"A"** are wrongly predicted as **"B"** but not as **"G"**. Maybe with a better data-cleaning, we could find characteristics to make better choices between two groups of grades. It looks to be a pertinent model for us.

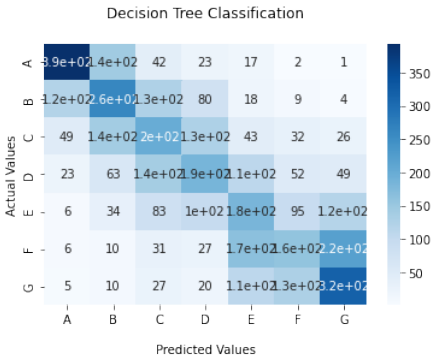

Figure 13: Confusion Matrix
Source : Our Python Code

Random Forest :

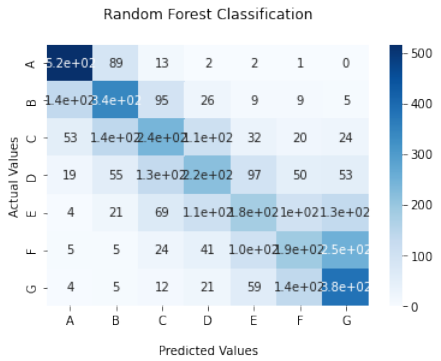

Figure 14: Confusion Matrix
Source : Our Python Code

Random Forest gives an accuracy of 48%. It is a good result, we are 3.4 times more likely to find the right grade than random. It is our best result, it is 10% more than Decision Tree. Moreover, the confusion matrix looks quite diagonal. As for Decision Tree model, the confusions are often between close labels and it could be improved with better data cleaning. Also, random forest does not over-fit the data and its accuracy is higher than the decision tree's one so maybe decision tree does not fit our data enough. According to the high accuracy and the good-looking of the confusion matrix, Random Forest is highly pertinent for our project.

To recap we print the comparison of our four models (accuracy) :

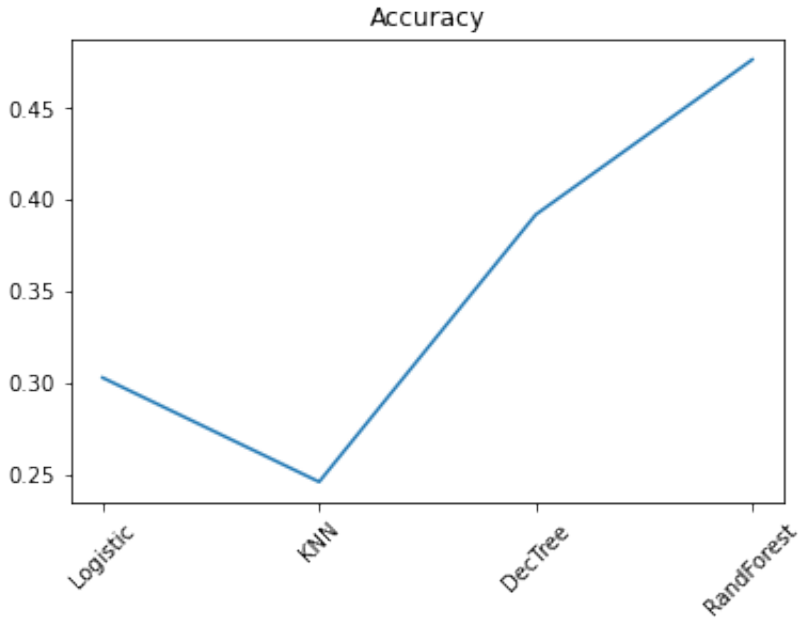

Figure 15: Comparison of accuracy
Source : Our Python Code

7 Project Management

We have worked regularly on our own and together also. Each week, we were finding a day to work as a team for at least two hours to take stock and discuss the project.

In the beginning, we had to find a topic to work on, which could be interesting for us and quite linked with our studies. We have chosen to deal with credit scoring using machine learning methods.

Here is our logbook relative to our project.

Session 1 (02/03/22):

We create a GitHub repository to share our work. It can be found here : **GitHub Repository**

We choose our dataset on Kaggle: https://www.kaggle.com/devanshi23/loan-data-2007-2014 . It is a Dataset on different mortgages in the USA.

Session 2 (09/03/22):

First, we open the data and check for the base rate. In our model, we will have 7 different outputs (A, B, C, D, E, F, G) so we expect a base rate of about 0.14 (1/7). Our current base rate is 0.17 and mostly the minimum rate is 0.01 for G. Our dataset is not balanced so we need to change that.

There are 3322 observations for G so if we delete enough observations in the other groups we could have a balanced dataset and still more than 23254 observations (3322 x 7). It is the strategy that we think to do.

We search for a function to randomly erase observations in our overrepresented groups.

Session 3 (15/03/22):

To begin with we want to remove the missing values: We tried to remove every row with at least a missing value but it deleted the whole dataset. We decided to change the strategy by deleting columns with a missing value. It is 'less professional' but it is an easy way to remove many parameters which seemed useless while keeping the same number of observations.

Then we balance the dataset: There are many ways to achieve that and we choose to do it randomly. Our idea is to randomly delete enough observations for each grade to balance the dataset. To do it we split our dataset into 7 datasets, one for each group. Then we randomly erase the right number for each grade (our target is to get about 3322 rows for each grade). It is very long to run when we delete rows one by one (about 3h). To deal with that problem, we delete rows ten by ten (about 15 min). After that, we concatenate the 7 new dataframes and we get one new balanced dataset. Finally, we download our new dataset so we can take it next time without running the 15 min code.

Session 4 (20/03/22):

We create the latex report on overleaf. We do the structure of the report, the introduction, the team, and the data presentation.

We do a qualitative study of the parameters. The idea is to delete useless parameters but it is only for those we can predict before doing a correlation study. It can be because of their nature (member id carries no information for the grades) or because of their quality (title is a sentence written by the borrower, it is not usable for a regression). We write that part in the report and we do the code in Python.

We discuss the way we deal with missing values and we want to change that to keep more useful parameters.

Session 5 (22/03/22):

We deal with missing values. First, we delete columns with more than 10 percent of missing values. Then we drop rows with missing values. That process lets us keep more columns without leaving too many features. We explain that part in the report.

We deal with dates to change their form. Then we begin to explain the process in the report.

We also explain how we balanced the dataset in the report.

We begin to do the correlation matrix for the quantitative study of our features.

Session 6 (30/03/22):

We change the structure of the report according to what Jiao Ying told us. It is less technical and more narrative. The technical part is now in the appendix part. We begin to write the new parts.

We continue the data cleaning.

Session 7 (31/03/22):

We finish the data cleaning. We do the Khi2 test for the categorical values and the correlation test for the numerical values. The goal is to remove every correlated value. We also factorize categorical values and normalize numerical values.

Then we prepare the cleaned data set for the classification. We split it into two parts : the train set and the test set.

We do the four following classification methods :

Logistic classification

KNN classification

Decision tree classification

Random forest classification

For each classification, we compute the accuracy and plot the confusion matrix.

Finally, we compare the methods, we conclude that random forest is the best method.

Session 8 (04/04/22):

We have an accuracy too high for decision tree and random forest (more than 90 percent. It shows a problem in our models. We find a perfectly correlated variable (interest rate). We remove it otherwise our model is cheated. We do the classifications again and we find our best accuracy of 48 percent for random forest. It is more realistic for a 7 labels model.

8 Conclusion

We have an accuracy of 48% with our best model. It is a high score compared to our target but we can wonder if it is an efficient model in reality. Indeed is the bank going to take a high risk if it trusts the algorithm to decide the interest rates?

The answer depends on how "bad" are the wrong results. If the model says someone is "A" instead of "B", it is not as bad as saying someone is "A" instead of "G". With the confusion matrix of random forest model, we observe that the model confuses almost only with close grades. So our model could work in a bank without being too risky for it.

Regarding the limits of our work, we are not sure that our dataset is honest. That means that we could have people falsely well graded for political reasons. It could bias the model but there are few probabilities because it would need a lot of them to have a real impact on the model.

We can conclude that our model is safe to predict grades even if it is not always exactly true. However, it could not be used for professional work yet but it could be a serious base for a real model.

9 Summary

Bankers make loans and to limit the default risk, they give different interest rates to borrowers according to their risks. To do that they give a grade to each borrower. Our goal is to make a strong way to find the right grades using Machine Learning.

We take a dataset from an American bank (LendingClub) as our base to build our model. The bigger part of the work is to clean the data. Then we apply four different methods: Logistic Regression, KNN Classification, Decision Tree, and Random Forest.

In the end, we can predict the right grade among seven possibilities with an accuracy of 48% and those that are mispredicted are mostly close to the good one.

10 Bibliography

LendingClub Website :
https://www.lendingclub.com/
Loan grades information :
https://www.lendingclub.com/foliofn/rateDetail.action
Benefit of Credit Scoring to Banks :
https://www.herald.co.zw/benefits-of-credit-scoring-to-banks/
The Big Short :
2015 movie on the Sub-primes mortgages crisis from Adam McKay.
Data Mining and Machine Learning :
2021 UNIL courses from Michalis Vlachos
Supervised Learning, Wikipedia :
https://en.wikipedia.org/wiki/Supervised_learning#:~:text=Supervised

11 Appendix

11.1 The rough data

The data set that we choose comes from Kaggle. It can be downloaded **here**. It is data on many mortgages from LendingClub between 2007 and 2014. LendingClub is a peer-to-peer lending company headquartered in San Francisco, California. It was created in 2006.

In the data, each row is a loan and each column is a characteristic of the loan. There are a lot of characteristics as there are 74 columns. Some of them are almost empty or useless for our work so we won't deal with them. Nevertheless, we still have many and enough parameters.

Let's introduce the parameters :

id : A unique LC assigned ID for the loan listing.
member_id : A unique LC assigned Id for the borrower member.
loan_amnt : The listed amount of the loan applied for by the borrower.
funded_amnt : The total amount committed to that loan then.
funded_amnt_inv : The total amount committed by investors for that loan then.
term : The number of payments on the loan. Values are in months and can be either 36 or 60.
int_rate : Interest Rate on the loan
installment : The monthly payment owed by the borrower if the loan originates.
grade : LendingClub assigned loan grades.
sub_grade : LendingClub assigned loan subgrade.
emp_title : The job title supplied by the Borrower when applying for the loan.
emp_length : Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more.
home_ownership : The homeownership status provided by the borrower during registration.
annual_inc : The self-reported annual income provided by the borrower during registration.
verification_status : Verified, source verified or not verified.
issue_d : The month which the loan was funded.
loan_status : Current status of the loan.
pymnt_plan : Indicates if a payment plan has been put in place for the loan.
url : URL for the LendingClub page with listing data.
desc : Loan description provided by the borrower.
purpose : A category provided by the borrower for the loan request.
title : The loan title provided by the borrower.
zip_code : The first 3 numbers of the zip code provided by the borrower in the loan application.

addr_state : The state provided by the borrower in the loan application.
dti : A ratio calculated using the borrower's total monthly debt payments on the total debt obligations.
delinq_2yrs : The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years.
earliest_cr_line : The month the borrower's earliest reported credit line was opened.
inq_last_6mths : The number of inquiries in the past 6 months (excluding auto and mortgage inquiries).
mths_since_last_delinq : The number of months since last delinquency.
mths_since_last_record : The number of months since the last public record.
open_acc : The number of open credit lines in the borrower's credit file.
pub_rec : Number of derogatory public records.
revol_bal : Total credit revolving balance.
revol_util : Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.
total_acc : The total number of credit lines currently in the borrower's credit file.
initial_list_status : The initial listing status of the loan. Possible values are – W, F.
out_prncp : Remaining outstanding principal for total amount funded.
out_prncp_inv : Remaining outstanding principal for a portion of the total amount funded by investors.
total_pymnt : Payments received to date for total amount funded.
total_pymnt_inv : Payments received to date for portion of total amount funded by investors.
total_rec_prncp : Principal received to date.
total_rec_int : Interest received to date.
total_rec_late_fee : Late fees received to date.
recoveries : Post charge off gross recovery.
collection_recovery_fee : Post charge off collection fee.
last_pymnt_d : Last month payment was received.
last_pymnt_amnt : Last total payment amount received.
next_pymnt_d : Next scheduled payment date.
last_credit_pull_d : The most recent month LendingClub pulled credit for this loan.
collections_12_mths_ex_med : Number of collections in 12 months excluding medical collections.
policy_code : Publicly available policy_code=1 new products not publicly available policy_code=2.
application_type : Indicates whether the loan is an individual application or a joint application with two co-borrowers.
acc_now_delinq : The number of accounts on which the borrower is now delinquent.

There are a lot of parameters and we will have to select which ones we keep and

which ones we leave.

There are even more parameters in the original data set but we did not introduce whose with too many missing values.

11.2 Presentation of grades

LendingClub gives a grade for every borrower. The grade reflects the risk from the borrower for LendingClub. Those grades are letters between A and G and have the following characteristics :

A : Risk of default is negligible, interest rate between 8.46% and 10.81%.
B : Risk of default is very low, interest grade between 13.33% and 16.08%.
C : Risk of default is moderate, interest rate between 17.30% and 20.74%.
D : Risk of default is average, interest rate between 22.62% and 30.99%.
E : Risk of default is possible, interest rate between 28.90% and 29.00%.
F : Risk of default is likely, interest rate between 29.35% and 30.75%.
G : Risk of default is very high, interest rate between 30.79% and 30.99%.

It is those grades that we want to predict using machine learning.

11.3 Suppression of null values

To begin with we want to remove the missing values. We tried to remove every row with at least a missing value but it deleted the whole dataset. We decided to change the strategy by deleting columns with a missing value. It is 'less professional' but it is an easy way to remove many parameters which seemed useless while keeping the same number of observations.
It is a good idea but we have a new problem, we delete some important parameters which have only a few missing values such as *emp_length*. We need to change that.

Our new idea is to remove only parameters that have more than k missing values. It let us keep some important values. Then we can remove rows with missing values. We have to choose k to select the columns we do not want to lose.

Our final way to deal with missing values :

We decide to remove columns with more than 10% of missing values. This value has been found after several trials. It appears to be the best one to keep features we want and drop those we do not want.

Then we remove rows with missing values. With this method, we do not drop too many observations.

11.4 Qualitative study

We are going to delete useless parameters. To do that we will discuss them qualitatively.
id and **member_id** : They are clearly useless for our model. Indeed they are only random numbers. **We do not keep it**.
int_rate : We hesitated for that one. If the interest rate comes from the grades, it is not legit to use it to find the grade. Yet we are not sure that it comes from grades so **we keep it**.
sub_grade : This parameter comes from the grades so we cannot use it to find the grade. On the one hand, using it would be cheating and on the other hand, we do not have it before having the grade. **We do not keep it**.
emp_title : There are too many null values, it should have already been deleted.
loan_status : It says if the loan is fully paid back or not. As we cannot know that at the beginning of the loan, it is irrelevant. **We do not use it**.
pymnt_plan : There are only 9 'True' and every other values are 'False'. It is not relevant to build a model. **We erase it as well**.
url : There is one unique categorical value for each row. It is useless. **We do not keep it**.
desc : It is a description of the use of the loan by the borrower. It is full sentences that are not usable. **We do not keep it**. Moreover, there are some null values so they should have already been deleted.
title : As desc, it is sentences written by the borrowers such as "*My wedding loan I promise to pay back*". It is funny but not usable. **We do not keep it**.
zip_code : There are too many different values. The base rate is 1%. **We delete it**.
initial_list_status : There is almost one unique value "f". It is useless for a regression or a classification. **We do not keep it**.
application_type : There is only one value "INDIVIDUAL". **We erase it**.

11.5 Dates rescaling

Now we are going to discuss Dates Rescaling. In our dataset, we can see that we have many date variables which are computed in the wrong format. That's why we are going to operate different modifications to make them fit our needs.

First of all, we are going to modify the variable **term** to remove the **"month"** at the end of the variable.

After that, we create a function **emp_length_converter** to modify the variable **emp_length** by removing the different terminologies of the variable and only keeping the number of years.

Finally, we are going to create the function **date_columns** to operate the difference between the date value in our dataset and today's date in order to have the same unit throughout the dataset i.e. the amount of months between two dates.We have applied that function to four variables : **earliest_cr_line**, **issue_d**, **last_pymnt_d** and **last_credit_pull_d**.

11.6 Balance of the dataset

We check for our base rate. As we have 7 outcomes, we should have a base rate around $1/7 \simeq 0.14$.

Unfortunately, the current base rate is above 0.29 and it is for the categorical value B. We need to balance our dataset if do not want to have absurd results. Moreover the rate of categorical value G is under 0.01. It is not balanced at all.

There are several ways to balance a dataset and we choose the easier one : to randomly delete observations in the over-represented groups.

The way we do it is simple. We know the number of observations for each categorical value and we want them to have the same number as the smaller group which is G.

We subtract the number of G rows to each number of every other group to know which number we need to delete in each other group.

Then we cut each group into different dataframes. Now we have 7 dataframes; one for each grade.

For a group of A, we define N as the number of rows that we need to remove. N times, we randomly pick a number between 0 and the size of the dataframe of grade A minus one and we delete the corresponding rows. At the end of the process, we have a new dataframe for A whose shape is the same as the dataframe of G.

We do that for every other dataframe and we concatenate them. Finally, we get our new balanced dataset.

While running the code we faced a little problem, the time of execution. Indeed it was approximately three hours to balance the dataset. So we had the idea to remove rows ten by ten to earn time and it worked. We could balance our dataframe in only twenty minutes.

To do that we change the following code :

```
for i in range(6):
    while  ListNbRowsToDelete[i] > 0:
        n = random.randrange(0, len(listGroup[i].index))
        listGroup[i] = listGroup[i].drop([listGroup[i].index[n]])
        ListNbRowsToDelete[i] = ListNbRowsToDelete[i] - 1
```

for that one :

```
for i in range(6):
    while  ListNbRowsToDelete[i] > 0:
        n = random.randrange(0, len(listGroup[i].index)-9)
        listGroup[i] = listGroup[i].drop([listGroup[i].index[n],
                                          listGroup[i].index[n+1],
                                          listGroup[i].index[n+2],
```

```
                                        listGroup[i].index[n+3],
                                        listGroup[i].index[n+4],
                                        listGroup[i].index[n+5],
                                        listGroup[i].index[n+6],
                                        listGroup[i].index[n+7],
                                        listGroup[i].index[n+8],
                                        listGroup[i].index[n+9]])
        ListNbRowsToDelete[i] = ListNbRowsToDelete[i] - 10
```

The second code is less random but still enough for us. The smaller group of grades that we transform has 12432 rows (group of F). When we delete 10 by 10, we still delete tiny parts (10^{-3}) each time.

11.7 Quantitative study

We do two studies, one for the categorical values with the Khi squared test and one with the correlation matrix for the numerical values.

Khi squared test :

We use the library "scipy.stats" and the code is as follow :

```
# Define an empty dictionary to store khi-squared test results :
khi2_check = {}

from scipy.stats import chi2_contingency
# We loop over each column in the training set to compute
# khi-statistic with the target variable
for column in X_train_cat:
    khi, p, dof, ex = chi2_contingency(pd.crosstab(y_train, X_train_cat[column]))
    khi2_check.setdefault('Feature',[]).append(column)
    khi2_check.setdefault('p-value',[]).append(round(p, 10))

# convert the dictionary to a DF
khi2_result = pd.DataFrame(data = khi2_check)
khi2_result.sort_values(by = ['p-value'],
                        ascending = True,
                        ignore_index = True,
                        inplace = True)
khi2_result
```

There is nothing relevant to removing some categorical variables at the end.

Correlation Matrix :

We use the library "seaborn". The code is easy with that library :

```
import seaborn as sns
import matplotlib.pyplot as plt
# calculate pair-wise correlations between them
corrmat = X_train_num.corr()
plt.figure(figsize=(10,10))
sns.heatmap(corrmat, cmap="Blues")

# We delete all the features whose correlations are greater
# than 0.7 in absolute value :

corrmat[abs(corrmat) > 0.7]
DataNew.drop(columns = ['funded_amnt',
                        'funded_amnt_inv',
                        'installment',
                        'total_pymnt_inv',
                        'collection_recovery_fee',
                        'mths_since_last_pymnt_d',
                        'out_prncp_inv',
                        'total_rec_prncp'],
             inplace = True)
```

We find some too correlated variables and we delete them.

11.8 Factorizing and Re-scaling

This is the final part of the data preparation. If we want that the machine understands the data, we need to "translate" them. To do so we **factorize** the categorical features and **re-scale** the numerical features. We use the following code :

```
# We factorize categorical features :

for j in range(len(X_train.columns)):
    if(X_train.dtypes[j] == 'object'):
        X_train[X_train.columns[j]] = pd.factorize(X_train[X_train.columns[j]])[0]
        .tolist()
        X_test[X_test.columns[j]] = pd.factorize(X_test[X_test.columns[j]])[0]
        .tolist()

# We normalize numerical features :

X_train[X_train_num.columns] = mean_norm(X_train[X_train_num.columns])
```

```
X_test[X_train_num.columns] = mean_norm(X_test[X_train_num.columns])
```