

# Machine Learning Engineer Nanodegree

## Capstone Project

Ralph Pinotti Leite

January 26, 2018

### 1 Definition

#### 1.1 Project Overview

Families and person are subject to a financial crisis and they may not be prepared to face this moment. Extra money is required to control the situation and in the case of a person with a bank history, it is easier to get a loan, because the bank can control the risk. In the case of people who are not banked, access to a credit can be very difficult and the situation can aggravate.

With this very frequent scenario in several countries, a company called Home Credit Group has started to work with a focus on responsible lending primarily to people with little or no credit history. We have many people struggle to get loans due to insufficient or non-existent credit histories. Unfortunately, this population is often taken advantage of by untrustworthy lenders. Home Credit strives to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. In order to make sure this underserved population has a positive loan experience. Home Credit makes use of a variety of alternative data including telco and transactional information to predict their clients' repayment abilities.

The challenge that home credit faces in question is to unlock the full potential of the data that are possible to use to ensure that clients capable of repayment are not rejected and that loans are given with a principal, maturity, and repayment calendar that will empower their clients to be successful.

The Home Credit company differs from the banks in general, the company needs to find some way to optimize the clients who will have difficulty making the payment of the loan. For this, Home Credit has turned its attention to machine learning techniques that can be a good solution to differentiate the profile

of the client. To apply this type of technique, the company must have information from clients that will be used to create rules and identify clients based on a default history.

The own company has provided client's information to be used by professionals to help build a healthy money landscape. The information can tell us about events of loans that have already happened, many of them, were paid and others were not. For each historical information, we can find personal information such as marital status, age, type of housing, the region of residence, job type, and education level. Home Credit also looks at financial backgrounds, including month-by-month payment performance on any loans or credit card balances that the applicant has previously had with Home Credit, as well as the amount and monthly repayment balances of any loans that the applicant may have received from other lenders.

All information was provided in a tabular format and a dictionary with the respective descriptions of each information. In all 7 sources of different information there is a connection key that allowed us to join the tables as follow:

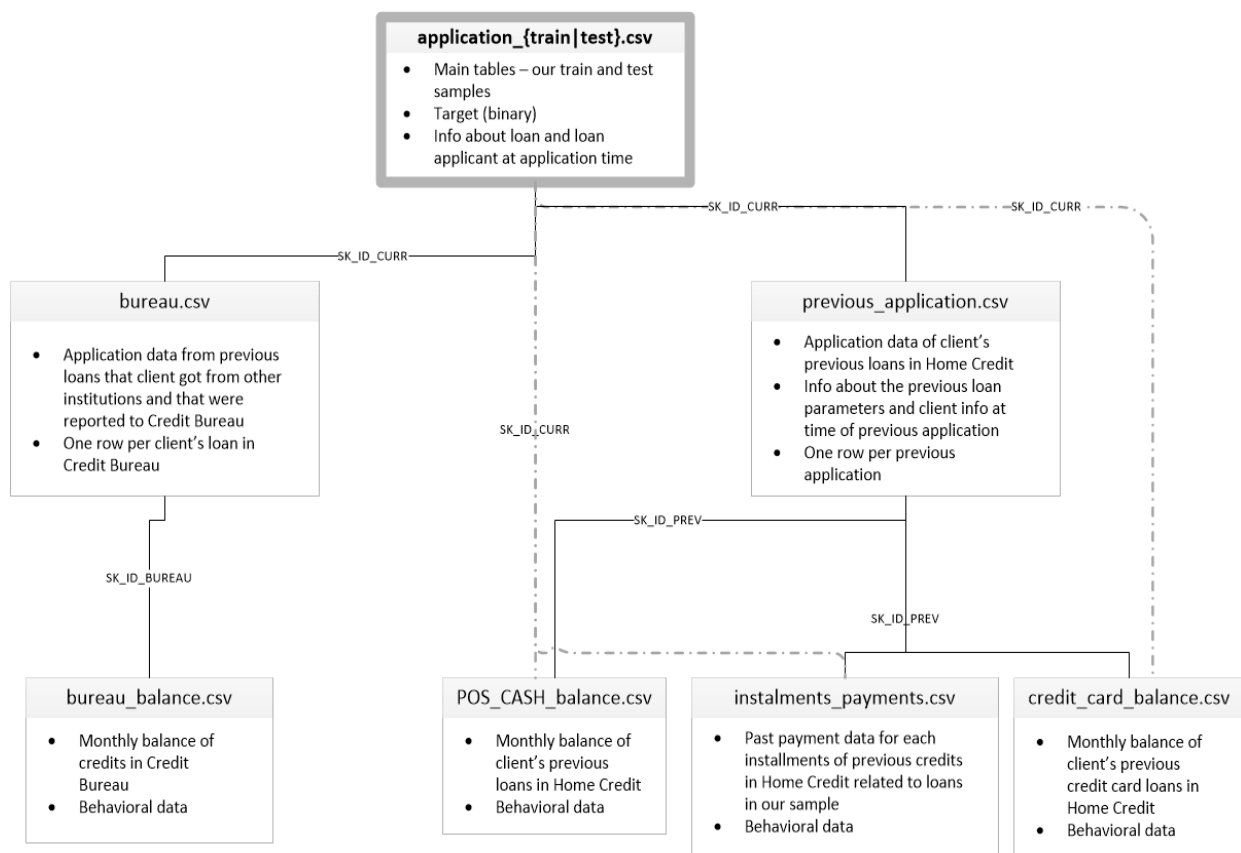


Fig 1: Data Base Diagram

## 1.2 Problem Statement

The objective of this project is to use historical loan application data and bureau information to predict whether or not an applicant will be able to repay a loan. Predicting whether or not a client will repay a loan or have difficulty is a critical business need. For each client information, we should classifier with the probability of repaying and compare with the original result (the observed result of client repays or not) based on an algorithm of our choice aiming the best result. We will measure the result with the Area Under the Curve (AUC).

The strategy for reaching the project goal begins with understanding the information we have available. This information needs to be handled and suited to be processed by a classifier. Different treatment methods will be used in this project, from feature custom (adjust irregularities), feature engineering and feature selection. It is important to remember that all data set must go through the treatment discussed. Another important factor is the connecting of variables with missing and correlated variables that need to be prioritized. The processing step is cyclic and during the execution of the project, the steps can be reviewed.

With the training and test tables ready, I will select classifiers habitually used in machine learning projects to read the data and generate patterns that can be used to profit the Home Credit business. The first classification, called the benchmark, was done using a logistic classifier since it is one of the classifiers that I have knowledge of and is easy to implement. The next classifications will challenge the first one trying to generate a better result that can be achieved using other techniques or parameters tuning. The other technique used will be the Light GBM classifier, a classifier with many characteristics like being fast in training, using little memory and is compatible with large databases.

The results begin to appear throughout all phases of the project, among them, I hope that the use of the light gbm technique can results in a better classification because the method is more attractive for problem-solving based on characteristic mentioned before. After another important step is the selection of parameters that I expect that the classifier will be able to distinguish the defaulters client from no-defaulters.

## 1.3 Metrics

We will measure the result with the Area Under the Curve (AUC), the metric is between 0 and 1 with a better model scoring higher. A model that simply guesses at random will have a ROC AUC of 0.5. After finding our best classifier we can apply the created rule in several clients that have the same information where the model was built.

ROC AUC (<https://medium.com/@andygon/eli5-roc-curve-auc-metrics-ac4fe482f018>)

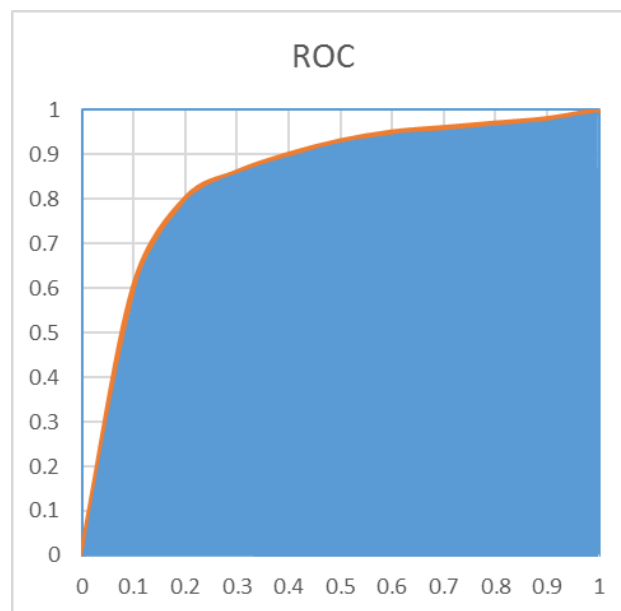


Fig 2: Area under curve example

The area shaded blue is the model's performance. The closer it gets to the top left corner, the better your model is doing at distinguishing your two classes apart. This area is specific to your model. The horizontal axis graphs the False Positive Rate (FPR), which is the complementary/parallel metric of the True Negative Rate, which, just like TPR, is the True Negatives overall recorded Negatives.

A ROC curve is a graphical plot that illustrates the distinguishing ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate measures the proportion of actual positives that are correctly identified as positive while the false-positive is calculated as the ratio between the number of negative events wrongly categorized as positive and the total number of actual negative events (regardless of classification). Let's think in our problems: the best scenario is to classify all no-defaulters as no-defaulters (True positive rate = 1) and all defaulters as defaulters (smallest

false positive rate). In this case, we are leading with machine learning where statistical technics are used and error are a consequence but it needs to be controlled. Others metrics can be applied but we need to pay attention in the project goal so, a metric like F1 score that is based on a balance between precision and recall is not suggested.

## 2 Analysis

### 2.1 Data Exploration

The data is provided by Home Credit, a service dedicated to providing lines of credit (loans) to the unbanked population. The information is used for a challenge in Kaggle. The name of the challenge is “Home credit default risk”. The data set contains information from 356,255 individuals who had previously been recipients of loans from Home Credit and each individual represented by their loan ID. The full data are composed by the following data source and can be found in (<https://www.kaggle.com/c/home-credit-default-risk/data>).

Are 7 data frame in total:

- (1) Application\_train/application\_test: the main training and testing data with information about each loan application at Home Credit. Every loan has its own row and is identified by the feature SK\_ID\_CURR. The training application data comes with the TARGET indicating 0: the loan was repaid or 1: the loan was not repaid.
- (2) Bureau: data concerning client's previous credits from other financial institutions. Each previous credit has its own row in bureau, but one loan in the application data can have multiple previous credits.
- (3) Bureau\_balance: monthly data about the previous credits in bureau. Each row is one month of a previous credit, and a single previous credit can have multiple rows, one for each month of the credit length.
- (4) Previous\_application: previous applications for loans at Home Credit of clients who have loans in the application data. Each current loan in the application data can have multiple previous loans. Each previous application has one row and is identified by the feature SK\_ID\_PREV.
- (5) POS\_CASH\_BALANCE: monthly data about previous point of sale or cash loans clients have had with Home Credit. Each row is one month of a previous point of sale or cash loan, and a single previous loan can have many rows.

- (6) Credit\_card\_balance: monthly data about previous credit cards clients have had with Home Credit. Each row is one month of a credit card balance, and a single credit card can have many rows.
- (7) Installments\_payment: payment history for previous loans at Home Credit. There is one row for every made payment and one row for every missed payment.

(<https://www.kaggle.com/c/home-credit-default-risk/data>)

The first table, Application\_train/application\_test, has 307,511 rows (91.93% Non-Defaulters and 8.07% Defaulters), each representing a unique individual who has previously borrowed money from Home Credit.

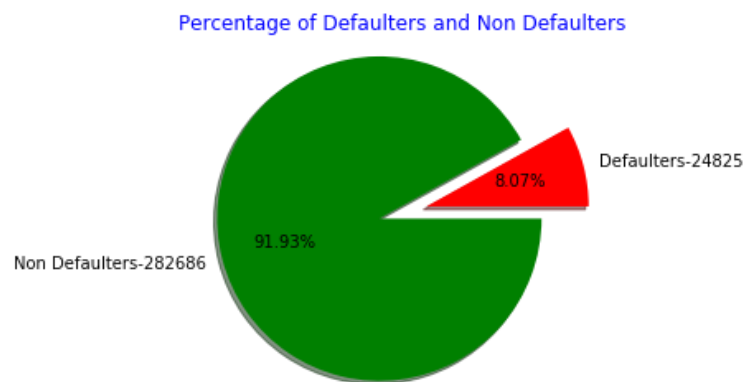


Fig 3: Percentage of Defaulters and Non Defaulters

The first two columns, SK\_ID\_CURR and TARGET, represent a borrower's loan ID and target value, respectively. The loan ID is a unique identifier assigned to each borrower. A target value of 1 indicates that the borrower eventually made at least one late loan payment. A target value of 0 indicates that the borrower always paid on time. The remaining 120 columns contain features that encapsulate various information from a borrower's loan application profile.

Example of a data frame:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL
0	100002	1	Cash loans	M	N	Y	0	202500.0
1	100003	0	Cash loans	F	N	N	0	270000.0
2	100004	0	Revolving loans	M	Y	Y	0	67500.0
3	100006	0	Cash loans	F	N	Y	0	135000.0
4	100007	0	Cash loans	M	N	Y	0	121500.0

Fig 4: Data Frame Example

We can show all data set in numbers:

Data Set Name	Numbers Row	Numbers Columns
Application_train	307,511	123
Application_test	48,744	122
Bureau	1,716,428	17
Bureau Balance	27,299,925	3
Previous_application	1,670,214	37
POS_CASH_BALANCE	10,001,358	8
Credit_card_balance	3,840,312	23
Installments_payment	1,3605,401	8

### 2.1.1 Encoding Categorical Variables

In some cases, we need to deal with pesky categorical variables. A machine learning model, unfortunately, cannot deal with categorical variables (except for some models such as LightGBM). Therefore, we have to find a way to encode (represent) these variables as numbers before handing them off to the model. There are two main ways to carry out this process:

**Label encoding:** assign each unique category in a categorical variable with an integer. No new columns are created. An example is shown below.



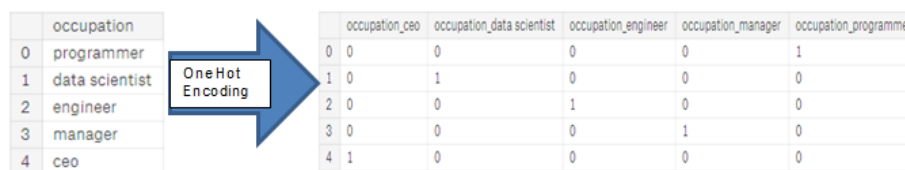
The diagram illustrates the process of label encoding. On the left, a table with two columns, 'id' and 'occupation', shows five categories: programmer (0), data scientist (1), engineer (2), manager (3), and ceo (4). A large blue arrow labeled 'Label Encoding' points to the right. On the right, the same table is shown, but the 'occupation' column has been replaced with numerical values: 4 for programmer, 1 for data scientist, 2 for engineer, 3 for manager, and 0 for ceo.

	occupation
0	programmer
1	data scientist
2	engineer
3	manager
4	ceo

	occupation
0	4
1	1
2	2
3	3
4	0

Fig 5: Example of label encoding

**One-hot encoding:** create a new column for each unique category in a categorical variable. Each observation receives a 1 in the column for its corresponding category and a 0 in all other new columns.



The diagram illustrates the process of one-hot encoding. On the left, a table with two columns, 'id' and 'occupation', shows five categories: programmer (0), data scientist (1), engineer (2), manager (3), and ceo (4). A large blue arrow labeled 'One Hot Encoding' points to the right. On the right, a new table is shown with six columns: 'id' and five binary columns representing each occupation: 'occupation\_ceo', 'occupation\_data scientist', 'occupation\_engineer', 'occupation\_manager', and 'occupation\_programmer'. Each row has a 1 in the column corresponding to its category and 0s in all other columns.

	occupation
0	programmer
1	data scientist
2	engineer
3	manager
4	ceo

	occupation_ceo	occupation_data scientist	occupation_engineer	occupation_manager	occupation_programmer
0	0	0	0	0	1
1	0	1	0	0	0
2	0	0	1	0	0
3	0	0	0	1	0
4	1	0	0	0	0

Fig 6: Example of process of create dummies

<https://www.kaggle.com/willkoehrsen/start-here-a-gentle-introduction>

We can observe some anomalies that occur in the data in real machine learning problems. These problems can originate from badly made records at the time of purchase or loan, poor data storage or some systemic error. A very common anomaly happens when a parameter in the client's registry is a default (male for example) and the same field is not mandatory for the registration completion. This causes the data to become biased as male. We can understand anomalies in general as outliers or information that do not make sense within the context being analyzed. When an anomaly is found we have some techniques to adjust the data that can be: To fill the missing data with the average value of the same feature in the case of a numerical data or delete the the feature.

One of the anomalies is, the numbers in the DAYS\_BIRTH column are negative because they are recorded relative to the current loan application. When we multiply the birthday day and divide by 365, we reach the age and distribution of the data makes sense.

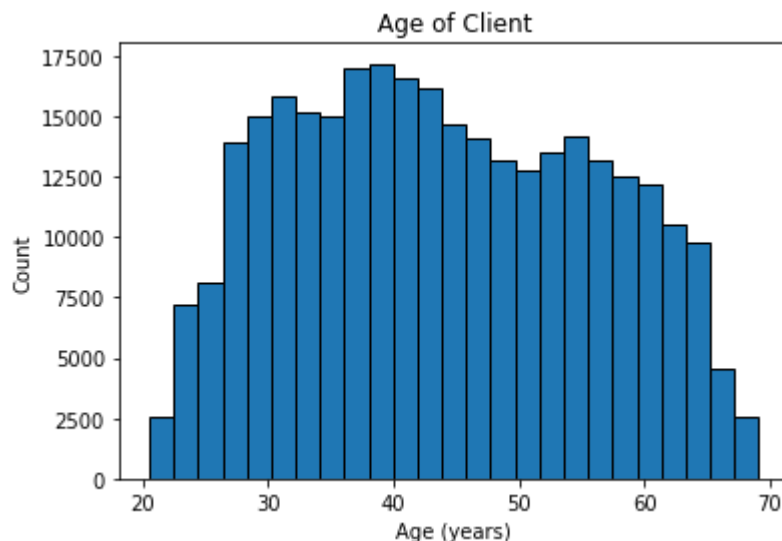


Fig 7: Age of Client distribution

The range of working days also has an anomaly. In this case, we can see that there is a very high frequency of working time in the table around 370,000 that seems to be a systemic or cadastral error.



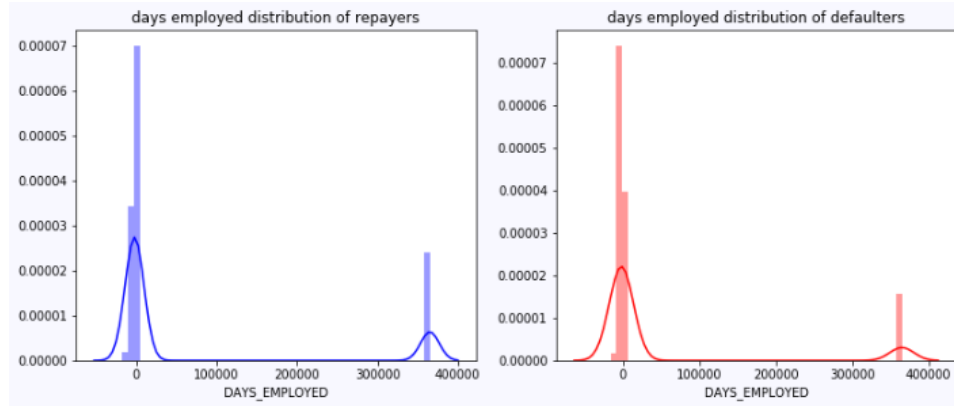


Fig 8: Days employed distribution

In the tables available to solve the problem in question, some variables have missing values. There is no rule for the amount of missing in each variable and this should happen due to registration or systemic problems, but the root cause is not important, we need at that moment to count the amount of missing in each feature and solve this anomaly.

To understand in a very simple way, we can build a graph that shows if there are variables with a large amount of missing inside database. We can say that there is a fairly large amount of variables with an average of 50% with missing as we can see in the graph below (Fig 9). An interesting point is that the test sample is faithfully following the missing distribution of the test variables which makes the rule developed by the classifier fit very well in the test data.

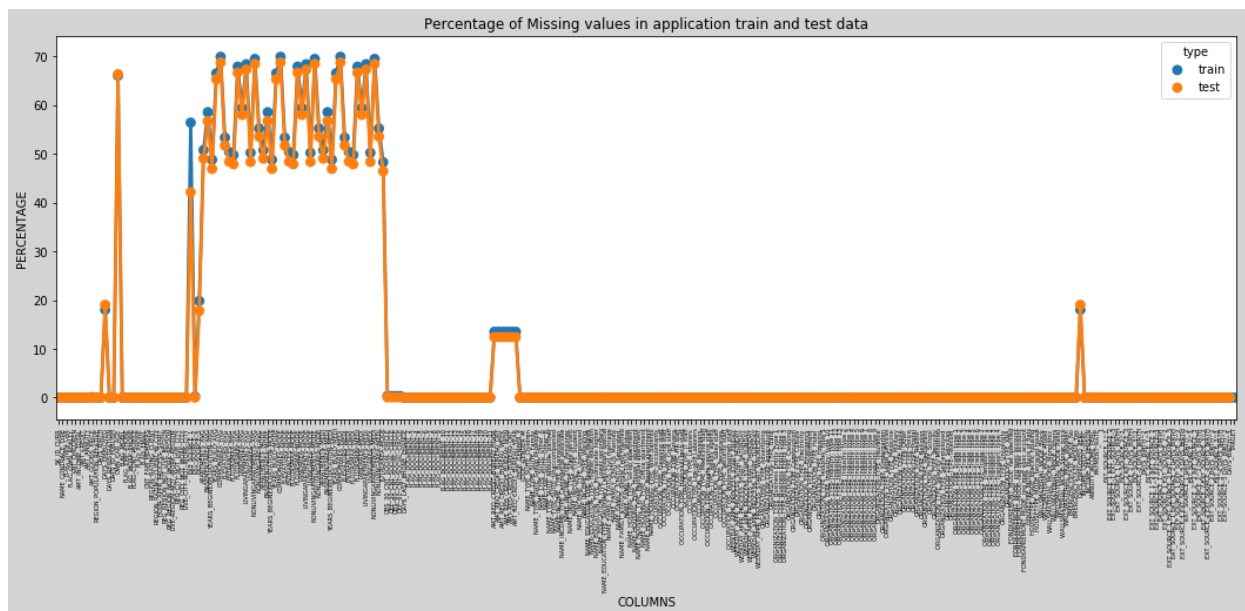


Fig 9: Percentage of missing in each feature

The data set BUREAU, for example, count 17 variables, 4 has a missing percentual more than 30%. In the process of feature selection, we will construct a rule where variables above a missing threshold will be excluded from the final table that will be used in the classifier.

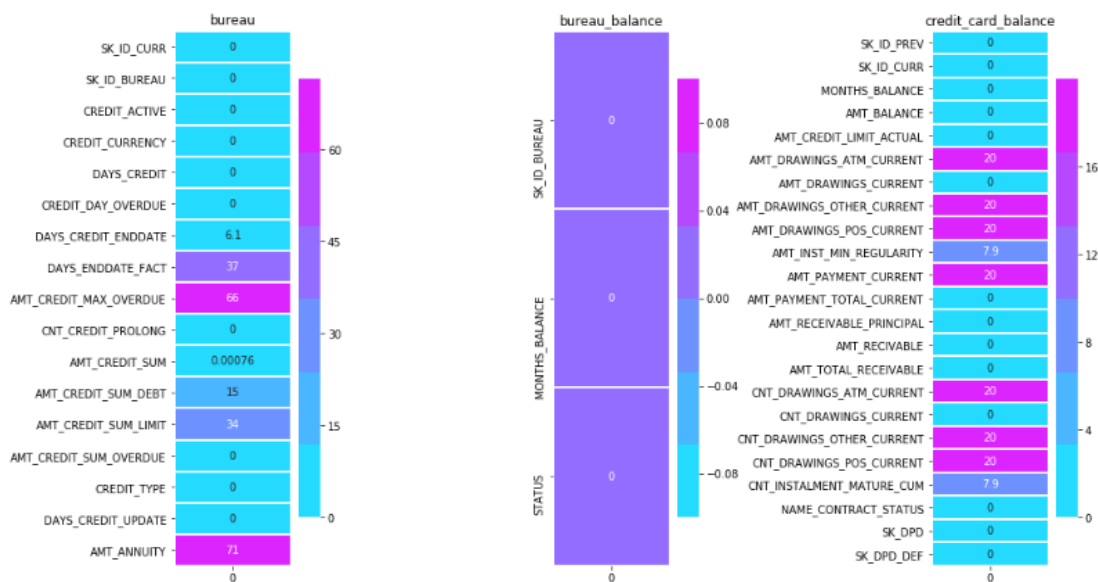


Fig 9a: Percentage of missing in each table

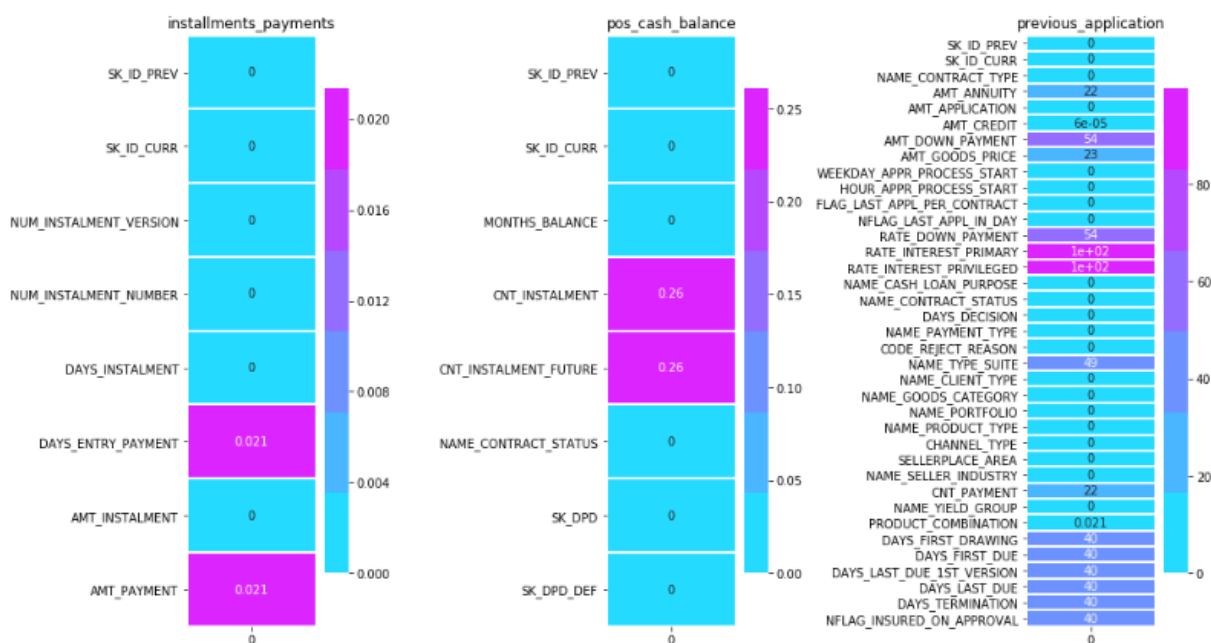


Fig 9b: Percentage of missing in each table

## 2.1.2 Feature Engineering - Polynomial Features

One simple feature construction method is called polynomial features. In this method, we make features that are powers of existing features as well as interaction terms between existing features. For example, we can create variables  $EXT\_SOURCE\_1^2$  and  $EXT\_SOURCE\_2^2$  and also variables such as  $EXT\_SOURCE\_1 \times EXT\_SOURCE\_2$ ,  $EXT\_SOURCE\_1 \times EXT\_SOURCE\_2^2$ ,  $EXT\_SOURCE\_1^2 \times EXT\_SOURCE\_2^2$ , and so on. These features that are a combination of multiple individual variables are called interaction terms because they capture the interactions between variables. In other words, while two variables by themselves may not have a strong influence on the target, combining them together into a single interaction variable might show a relationship with the target. Interaction terms are commonly used in statistical models to capture the effects of multiple variables, but I do not see them used as often in machine learning. Nonetheless, we can try out a few to see if they might help our model to predict whether or not a client will repay a loan.

In this project, we will create polynomial features using the  $EXT\_SOURCE$  variables and the  $DAYS\_BIRTH$  variable. We will use a technique called `PolynomialFeatures` that creates the polynomials and the interaction terms up to a specified degree. We can use a degree of 3 to see the results (when we are

creating polynomial features, we want to avoid using too high of a degree, both because the number of features scales exponentially with the degree, and because we can run into problems with overfitting).

## 2.2 Exploratory Visualization

In many machine learning projects, there are unknown challenges. Exploration of the data helps to determine the form or composition of the data. When we transform information from a DataFrame into a graph, anomalies and outliers can be quickly optimized as well as the distribution between our Target, as for example in the variable OWN\_CAR\_AGE:

We can see a concentration of information around 75 years that is totally far from the distribution that has a mean of 15 years, an option is to treat this information substituting that value by the average of the population. Another relevant point is that the distribution of payers has a different form from non-payers and this can be a very relevant variable for the classifier.

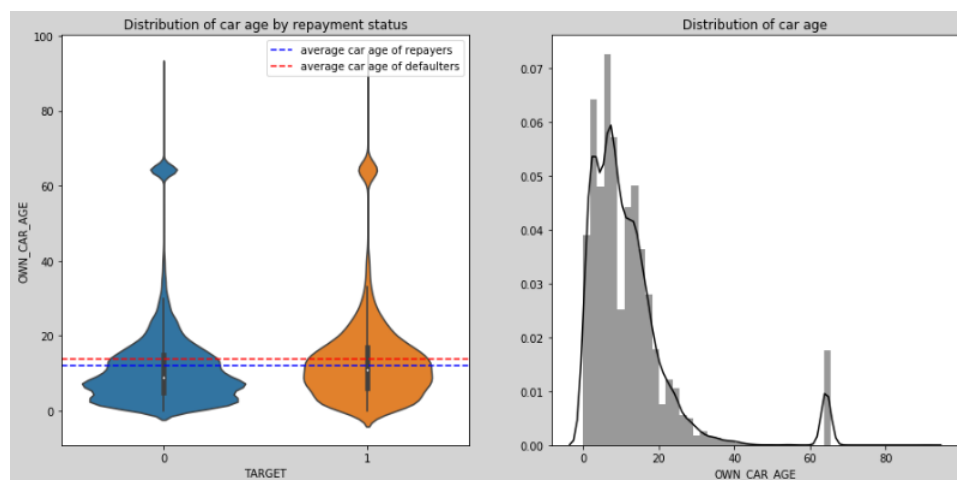


Fig 10: Distribution of OWN\_CAR\_AGE feature

Other relevant information when viewed in the graph is the variable EXT\_SOURCE\_3. The distribution of the variable showed in the chart that is separated by repayers and defaulters can show us different in the form of populations. It shows what exactly we are looking for, differentiating payers and non-payers.

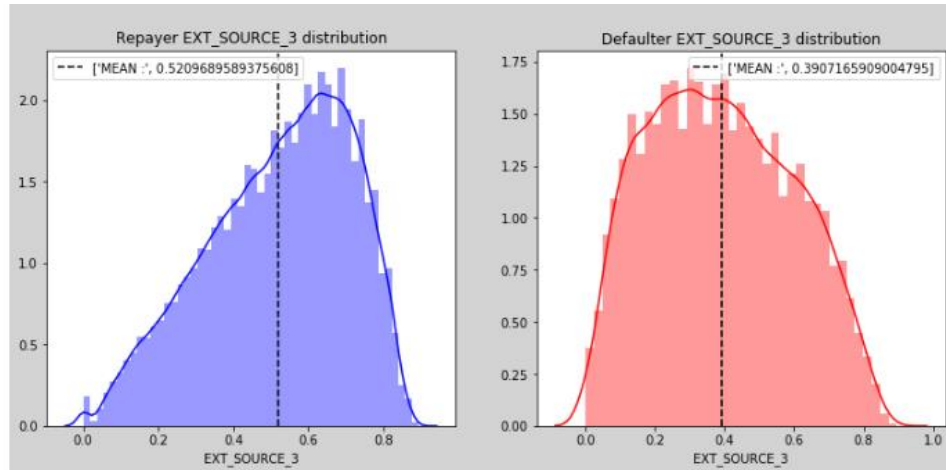


Fig 11: EXT\_SOURCE\_3 feature distribution

The graph of the distribution of age separated by defaulters and non-defaulters can bring interesting information that can be used to train our classifier. We can see the distribution of days of life (variable DAYS\_BIRTH) of public defaulters quite distinct from the public non-defaulters. We can notice that there is a greater concentration of information about  $-10,000$  for the distribution of defaulters, which means that the clients are young. In the distribution of non-defaulters, we can see a concentration around  $-15,000$ . Another very important information is when we compare the gender versus Days\_birth feature separated by the target variable. The male and female gender has a lower average age when we look at the defaulters.

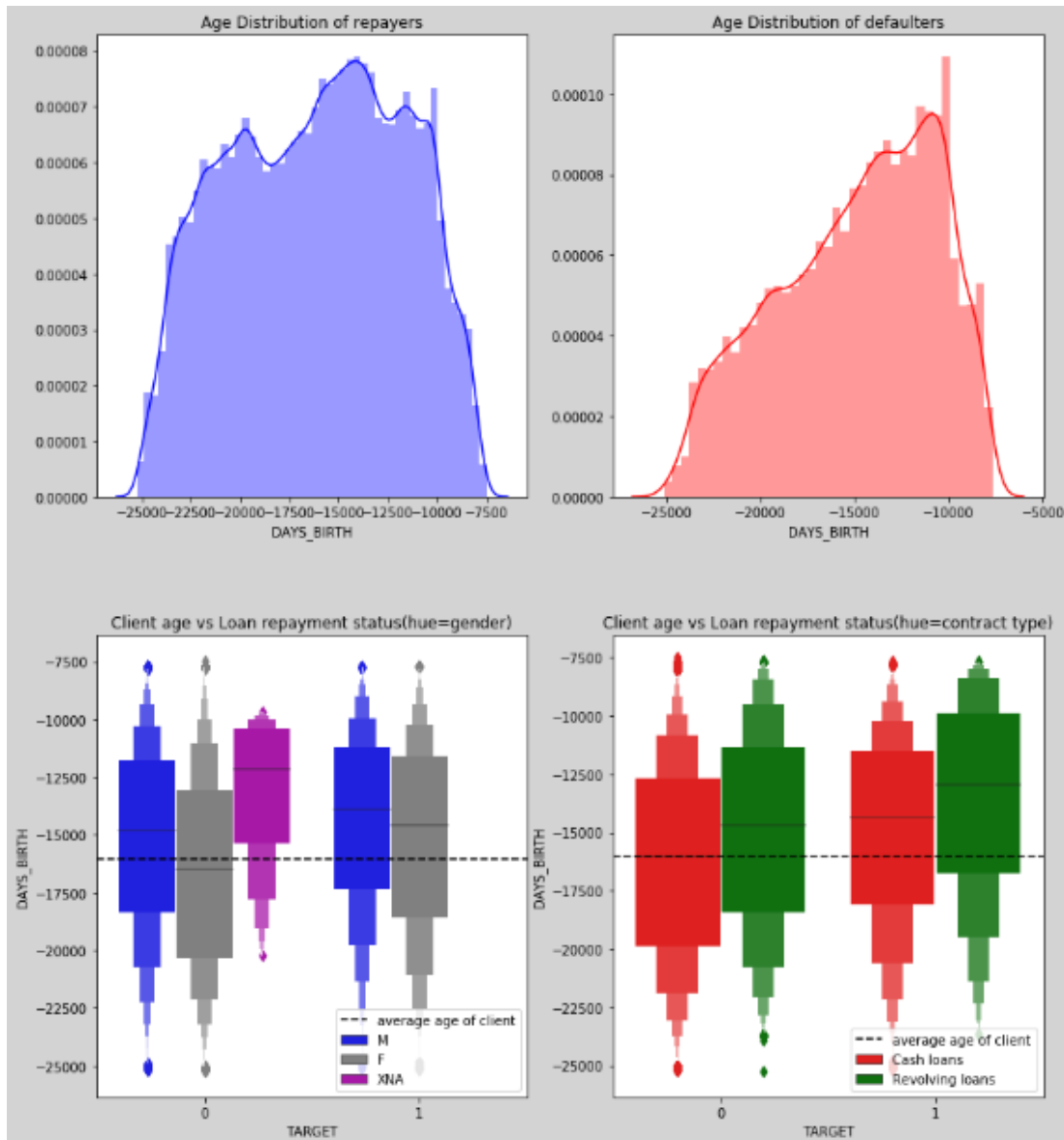


Fig 12: Age distribution overview

We can increase the knowledge about the problem looking to Bureau data Frame. This data frame can tell us about the client's previous credits provided by other financial institutions that were reported to Credit Bureau (for clients who have a loan in our sample). The first feature that we can observe is the number of previous credits by clients. Looking this information for repayers and defaulters we can see a difference in the mean of the distribution. Repayers used to have less previous credits while defaulters used to have more previous credit. This is a good sign because the population are different between each other and it can be useful in our classifier

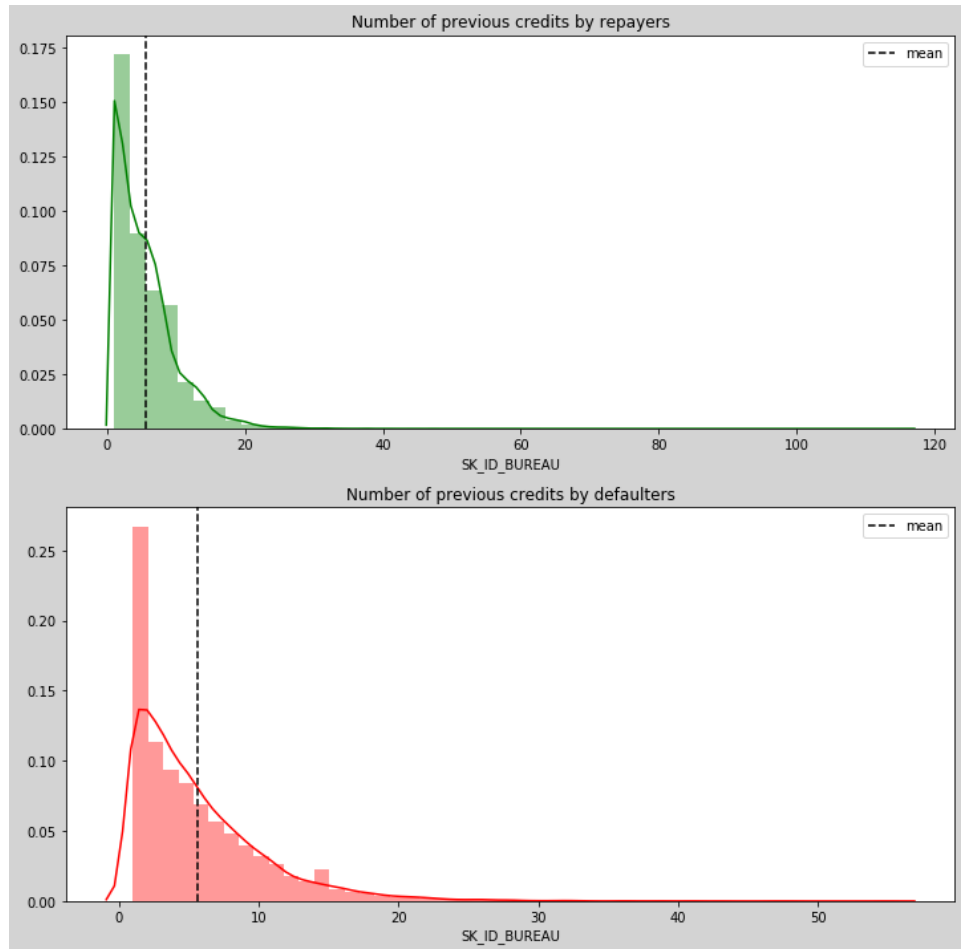


Fig 13: Distribution of the previous credit

As more feature we tried to understand clearer is the behavior of the clients. We can build some rules: more previous credits and younger is the client, high is the probability of no pay. We can deduce this looking for previous analysis. If I add a piece of new information, how many days since the previous credit does the client have? Fig 14 shows us that defaulter has smaller time between actual credit and the previous credit and the population as quite different

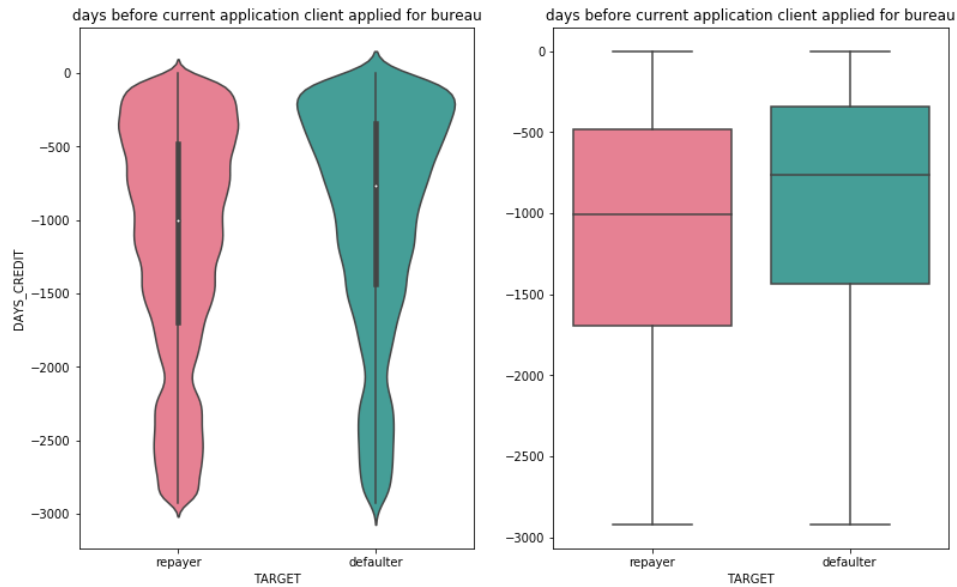


Fig 14: Distribution of days of previous application

Data is so important nowadays that we can infer the client's behavior and tries to use this information in order to profit our business, in this case, Home credit business. If we take a look in previous\_application data frame we can see for what propouse the clients were looking for a loan. In this graph Fig 15, we can see that loan are accepted by type. The most approved is to purchase electronic equipment, for Everyday expenses and Education and refused are for types with finality to pay others loan a to by a new car.





Fig 15: Distribution of propose to get a loan

The home credit uses the own data to try to create good rules for decide news loan and for it, the company needs to know how is the behavior of the client that has already have a loan. We can see in pas\_cash\_balance data frame how is the age in months, the client is located. The high concentration of this distribution is around 10 months and there are clients with an active loan for 96 months.

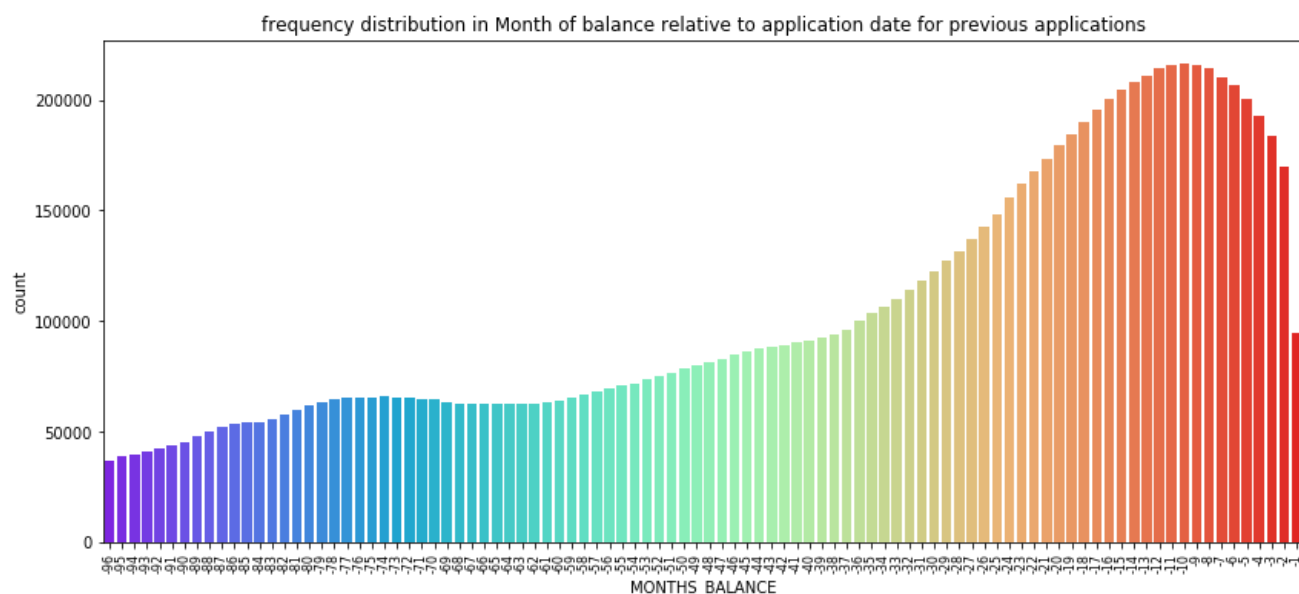


Fig 16: Frequency distribution in Months of balance relative to application date for previous application

## 2.3 Algorithms and Techniques

### 2.3.1 Light GBM

Light GBM is a gradient boosting framework that uses the tree-based learning algorithm, it grows tree vertically while another algorithm grows trees horizontally meaning that Light GBM grows tree leaf-wise while another algorithm grows level-wise. It will choose the leaf with max delta loss to grow. When growing the same leaf, Leaf-wise algorithm can reduce more loss than a level-wise algorithm.

Like XGBOOST, LightGBM is good with large datasets. However, LightGBM typically trains much faster than XGBOOST and also uses less memory.

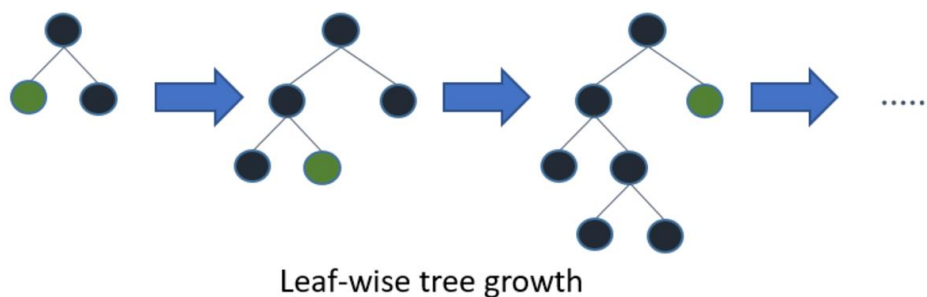


Fig 17: Example of leaf-wise tree growth

The size of data is increasing day by day and it is becoming difficult for traditional data science algorithms to give faster results. Light GBM is prefixed as 'Light' because of its high speed. Light GBM can handle the large size of data and takes lower memory to run. Another reason why Light GBM is popular because it focuses on the accuracy of results. LGBM also supports GPU learning and thus data scientists are widely using LGBM for data science application development.

It is important to remember that use LGBM on small datasets is not suggested. Light GBM is sensitive to overfitting and can easily overfit small data. Implementation of Light GBM is easy, the only complicated thing is parameter tuning. Light GBM covers more than 100 parameters and runs a GRID to find the best parameters can cost a lot of memory. It is very important for an implementer to know at least some basic parameters of Light GBM.

Advantages of Light GBM:

1. Faster training speed and higher efficiency: Light GBM use histogram based algorithm i.e it buckets continuous feature values into discrete bins which fasten the training procedure.
2. Lower memory usage: Replaces continuous values to discrete bins which result in lower memory usage.
3. Better accuracy than any other boosting algorithm: It produces much more complex trees by following leaf wise split approach rather than a level-wise approach which is the main factor in achieving higher accuracy. However, it can sometimes lead to overfitting which can be avoided by setting the max\_depth parameter.
4. Compatibility with Large Datasets: It is capable of performing equally good with large datasets with a significant reduction in training time as compared to XGBOOST.
5. Parallel learning supported.

### 2.3.2 Logistic Regression classifier.

Since logistic regression is often used for binary classification problems, it was one of the most used learning algorithms that are used in financial companies. Logistic regression generates coefficients for each of a dataset's features and then constructs a linear equation composed of these features and their coefficients that can generate the best possible predictions. As the logistic regression classifier gets trained on each point in the dataset, the value of each feature's coefficient is continually modified so as to maximize the likelihood of observing the sample values. Specifically, logistical regression adjusts the feature

coefficients to maximize the logged odds of observing the sample values. I will use logistic regression to performance my benchmark classifier and try to get better results with light GBM.

Below, there is some example of each technique that was used to arrive at an ideal data frame to be able to be processed by the classifier

1. *Data engineering* (create new variables): The objective of data engineering is to create new features (also called explanatory variables or predictors) to represent as much information from an entire dataset in one table. Typically, this process is done by hand using operations such as group, aggregation. Moreover, manual feature engineering is limited both by human time constraints and imagination: we simply cannot conceive of every possible feature that will be useful. The importance of creating the proper features cannot be overstated because a machine learning model can only learn from the data we give to
2. *Aggregation*: To aggregate information, usually numeric, that is in different granularity than the original table
3. *Transformation*: Some negative numbers from the row data frame had to be transformed in positive (in the case of age).
4. *Label encoding*: Label encoding: assign each unique category in a categorical variable with an integer. No new columns are created.
5. *One hot encoding*: One-hot encoding: create a new column for each unique category in a categorical variable. Each observation receives a 1 in the column for its corresponding category and a 0 in all other new columns.
6. *Polynomial transformation*: here we have a combination of variables to be able to find and create new variables that may be interesting for the project
7. *Missing value*: Replace missing values for feature mean.
8. *Correlated features*: Drop features with a selected threshold

## 2.4 Benchmark

Each kaggle competition has a Leaderboard rank. The leaderboard is composed of teams that are ordered by results. Each result is the metric that was chosen to evaluate the challenge. As a benchmark model, we have the Kaggle competition own result. We have 4 benchmarks based on AUC metrics from users. (<https://www.kaggle.com/c/home-credit-default-risk/leaderboard>)

1. In the money: First 3 high AUC
2. Gold: From 4 to 24 high AUC
3. Silver: 25 to 559 high AUC
4. Bronze: 600 to 719 high AUC

Usually, the leaderboard is composed of experienced users and in this case, the difference between the first AUC and the last Bronze (719th) are in the decimal case. I expect to do a good job and get an AUC around the bronze rank.

As a second benchmark, we used only the information from the training and test tables to generate a classifier using the Logistic Regression method. This classifier will generate a result that would serve as a parameter for comparison to the last classifier. The AUC metric will be compared and the classifier with the AUC will be the one chosen to solve the problem.

## 3 Methodology

### 3.1 Data Processing

The methodology should follow a sequence of steps to transform the raw data in variable useful for our classifier give us a good probability of home credit client belong to class defaulters or non defaulters. In general is very difficult to find a data set ready to be used in a machine learning process, so it is necessary a very exhaustive step: EAD Analysis, Feature Engineering, Feature Selection, and classification.

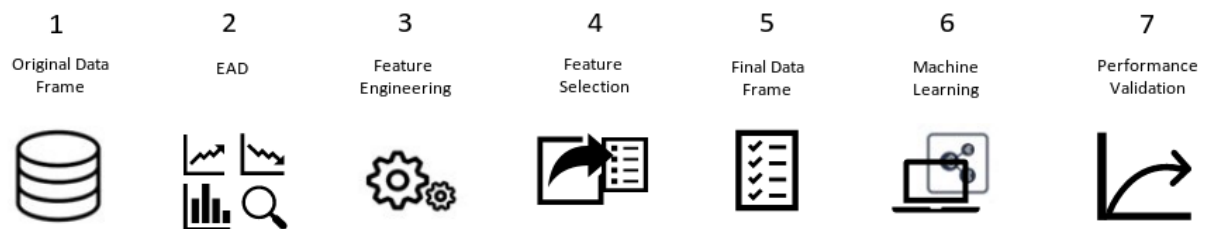


Fig 18: Solution process road map

1- Original data frame

Initially, the data was downloaded in a data frame format, along with the original tables there is a dictionary that explains all variable. There are 7 different data frame that can be connected to each other by a key according to the 'Fig 1' (Figure 1).

Because of memory limitations, steps 1 through 5 were made separate for three groups of tables:

Group 1: Train and Test data set.

Group 2: Bureau and Bureau\_balanced data set.

Group 3: Previous\_Application, Pos\_cash\_balance, Installment payment, credit card balance data set.

Inside each group, each data set was submitted through steps 1 through 5. In the end, the tables of the same group were grouped by the key generating a new data frame. This new data frame has also been submitted to steps 2, 3, 4 and 5 because the data can still have a correlation between variables. After all this step, all final data frame was grouped in a second final data frame.

*Group 1 exemplo:*

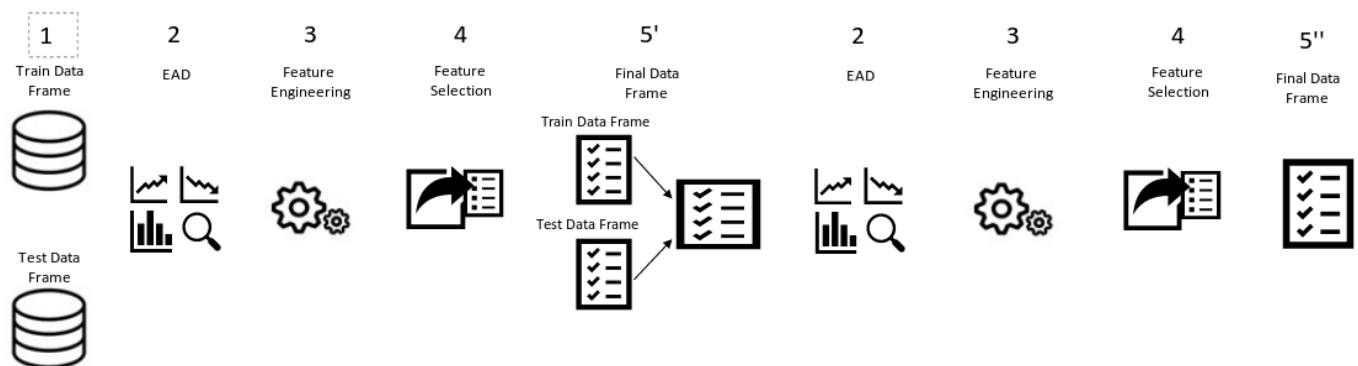


Fig 19: Solution process road map for group 1

This process was done for all three groups of tables. In the end, I have a table for each group that was grouped by a key and this table was submitted to the process of steps 2, 3, 4, 5, 6 and 7. Intermediate stages will be discussed in the next topics

## 2- EAD

To better understand the data we transform the tables into graphs where the information can be viewed in a different way. If some data showed anomalies, they were treated as described in the technique

step. Charts can be analyzed and some conclusions start to build form, for example, younger clients have a higher payment problem.

The EAD process is important in the machine learning process because it helps to observe the data in graphical format, identify anomalies and correlation between the tables and data. There is no better way to do EAD, there are several techniques that can be applied to find conclusions. In this case for continuous data, histogram and BoxPlot were used to identify the distribution of each variable, for categorical data and flag features were used bar graphs format

Some more significant features were worked with visualization where some anomalies could be found. Note: a machine learning process is cyclical, after arriving at the end of the process that is the result of the new classifier, new insights are generated. With these insights, we can go back in the initial part and intensify the analyzes in the variables more meaningfully. This process can be repeated several times.

*Anomalies:*

**1-DAYS\_BIRTH:** It was transformed to positive and divided by 365 to generate the variable age

**2-Age Working days:** Outliers were inputted as the median of the population

### **3-Feature engineering**

Feature engineering is a process that returns a very good gain for a classifier. It is made by combinations of variables generating new ones. In this project, we use a polynomial process of degree 3 to relate variables and create new ones.

For the numerical variables, some values were created as maximum, minimum, average the label encoding processes, one hot encoding, dummies described in the techniques section were used in the variables for the classifier to be able to understand the inputs and to process a solution

### **4-Feature Selecting**

Variables with missing: All variables with more then 60% missing were excluded. The calculation was done by adding all the missing records of each column and divided by the total of record

**5-Correlated variables:** All variables with more than 0.6 correlation indices were excluded.

After generating a rank with more explanatory variables by the process of feature select, I decided to keep all the variables to see if the classifier can still win with less explanatory variables

### 3.2 Implementation and refinement

After the final tables are finished, it means all the process of data engineering and data selection, the tables were submitted to the classifiers.

The first classifier to be evaluated, as previously discussed, was logistic regression. In Logistic regression was used only the data from the group 1 table (test and data frame test). In this process, the default parameters of the classifier were used and the roc curve was generated to serve as the benchmark result.

After obtaining the benchmark classifier, the other variables were adjusted according to the road map (Fig 14) and consolidated with the initial training and test table. This new table went through the validations and was submitted again to the logistic regression process and soon after the new Light GBM classifier. The two resulting roc curve, were evaluated and compared to decide which classifier will be used. It was decided to follow with the light gbm classifier.

During the process, some difficulties were encountered but all contrived to arrive at the objective of the project. Among the difficulties is working with all the data set together that consumes a lot of computational memory, for it was necessary to export the tables of each step and consolidate post but it not take a long time and can be replicated several times. In the process of feature engineering, it is necessary to be alert because the variables can go out of control depending on the degree of the polynomial. The point of greatest difficulty was the selection of light gbm parameters that can be made through a grid technique. The automation of this stage has consumed many hours of processing, which left unfeasible to wait for the final result due to my deadline for this project. For this, the parameters were manually tested until arriving at the best combination. All these difficulties are present in a machine learning project and depend of us to getting around the problems and at the end of this process, the classifier can easily be implanted on-line



## 4 Results

### 4.1 Model Evaluation and Validation

The validation phase of the model is very important because we are deciding which model we will propose for the home credit company to make its product offer strategy. The method adopted is simple but one of the most used in machine learning.

First of all we create a simple classifier that works on our data set and use the result as benchmark. After that, we need to try to find new features or new classifier that give us a better result, in this case, better AUC value. As we discussed in section 3.1, logistic regression was the first classifier (our benchmark ) with AUC = 0.66

Was used a random forest classifier with a method called 'feature importance' that show us what were the most relevant feature that the algorithm has used to get the best result. This random forest classifier has used the same data frame that was used in logistic regression. The top 15 feature in importance for this first classifier were:

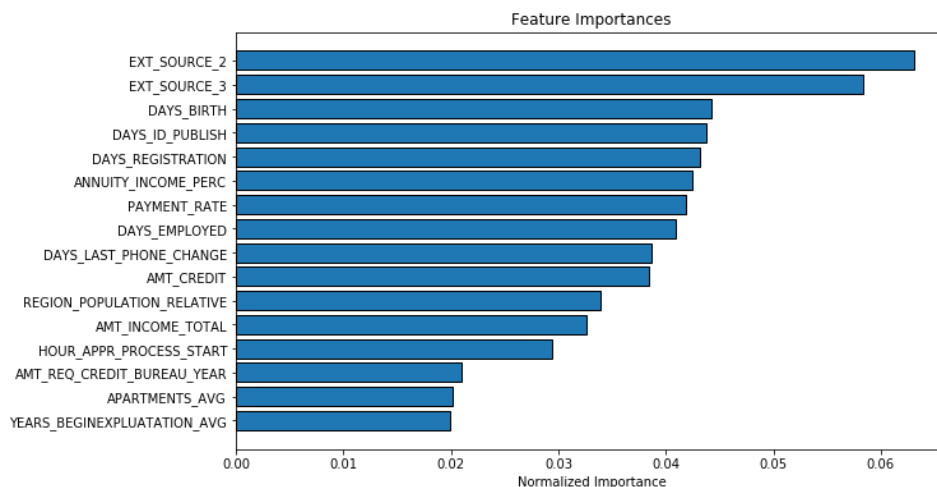


Fig 20: The top 15 feature (Exclude SK\_ID\_CURR)

Fig 20 shows us that external information (EXT\_SOURCE\_2, EXT\_SOURCE\_3) has great weight and shortly thereafter Days\_Birth shows great importance too. It is working as I expect.

The second classifier was done with all the variables already adjusted and consolidated described previously but there was no adjustment of parameters. The parameters tuning process will be part of the next step. The only difference here is the number of variables we are using, now we have the information of group 2 and group 3 of variables. We can see information as DAYS\_CREDIT in fig 21 that was not present in benchmark classifier and here this feature is in top 15 more important for our classifier. The value of the AUC metric from logistic regression is 0.70

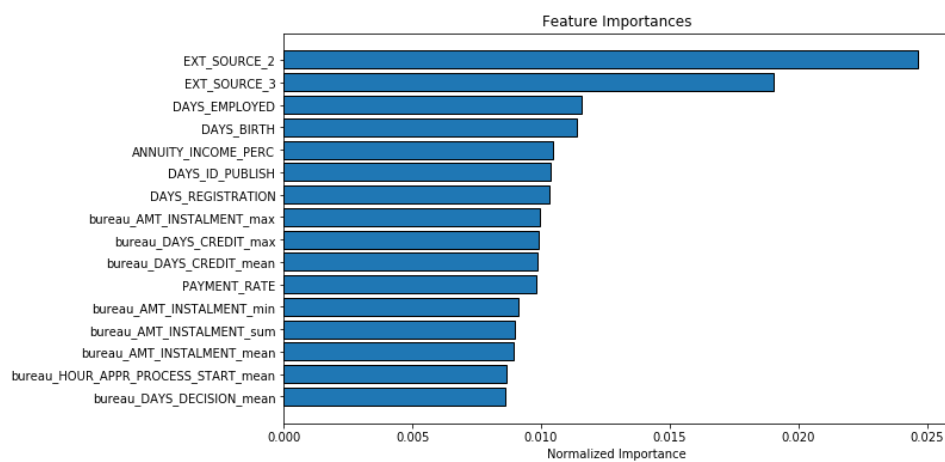


Fig 21: Feature importance with full data

The third step we changed the classifier to the GBM Light without parameter tuning and the value of AUC was 0.765. The order of the feature importance change a little bit, for example, Light GBM identifies AMT\_CREDIT(amount of previous credit) as an important feature.

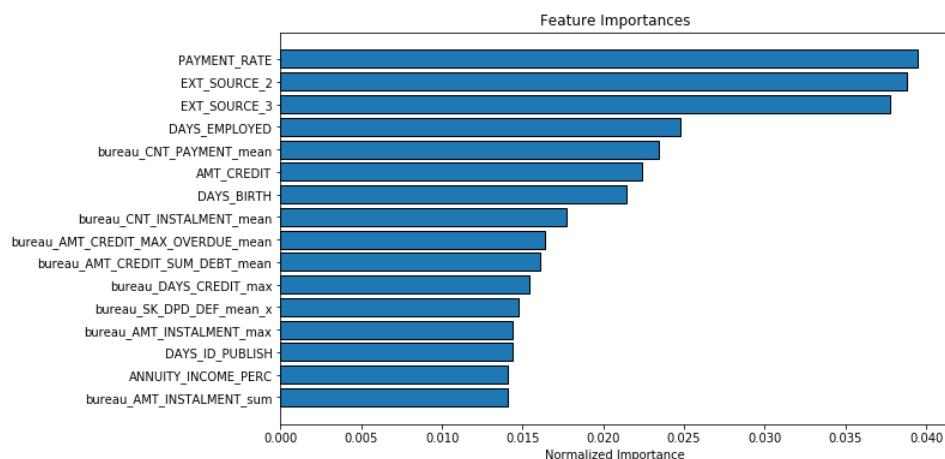


Fig 22: Feature importance with full data with Ligth GBM default parameters

Finally, as we have seen that the Light GBM classifier improved the performance compared to the logistic regression classifier I tried several combination until get the best AUC

#### Tried Parameters

- 1) n\_estimators: 500,1000,2000,5000
- 2) Objective: 'binary'
- 3) class\_weight : balanced,
- 4) learning\_rate: [0.05,0.6],
- 5) reg\_alpha: [0.1,0.2,0.3],
- 6) eg\_lambda:[0.05,0.1,0.5],
- 7) subsample:[0.8,0.7,0.5],
- 8) n\_jobs : [-1,2,3],
- 9) random\_state: [50]

The selected paramns were:

- 10) n\_estimators: 5000
- 11) Objective: 'binary'
- 12) class\_weight : balanced,
- 13) learning\_rate: [0.05],
- 14) reg\_alpha: [0.1],
- 15) eg\_lambda:[0.1],
- 16) subsample:[0.8],
- 17) n\_jobs : [-1],
- 18) random\_state: [50]

After this selection of parameters, the model was trained and validated again using the curve AUC and the value was 0.768. The importance of the features has changed just a little bit but in general, all features that were important with without parameters are present in features that in a process with tunned parameters

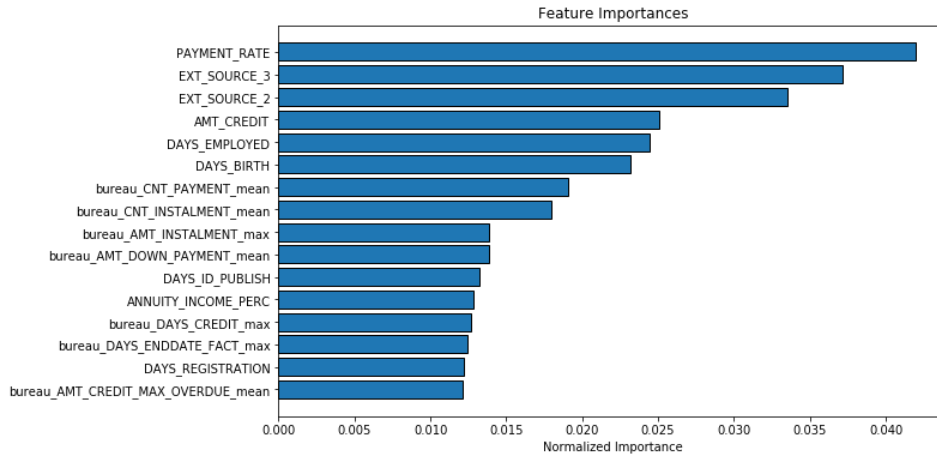


Fig 23: Feature importance with full data with Ligth GBM default parameters

To avoid overfitting and to generalizing over unseen data all processes were trained in 10 random samples of data. This kind of process has positive points like letting the solution and classification generalize the rules on all data but also has the negative part which can greatly increase the processing time.

To make the model more robust and reliable, a technique of data separation in training and testing was used, this process serves to validate if the rule that was constructed can be reliably attributed to a new data source (that belongs to the same population)

## 5 Conclusion

### 5.1 Free Form Visualization

To exemplify the result we need to compare the AUC area with the solution of the 4 models generated with different condition

0 Random Sorter: AUC = 0.50

1 Logistic Regression with training variable only: AUC = 0.665

2 Logistic regression with all tabs: AUC = 0.707

3 GBM lighth without parameter setting: AUC = 0.765

4 GBM lighth with parameter setting : AUC = 0.768

The best result in the main features was obtained by the LightGBM classifier used with parameters that was tuned and is clearly acceptable compared to Kaggle's solutions.

## 5.2 Reflection

Currently, machine learning methods have helped companies to improve profits and discover new opportunities in the market. The problems faced are not always easy to solve, in the case of the Home Credit company that requested a thorough analysis, that resulted in a practical solution to be implemented. Understanding the problem and how it should be solved is one of the utmost importance to complete the project

The initial process involves the acquisition of the company information that is necessary to analyze, so the solution can be optimized. The available dictionary helped to understand the comportment of the information and how is the relationship between them. After acquiring the information it was necessary to understand the behavior between the variables and the variable response.

To facilitate the understanding of the data the EAD process is very important, it can transform the information from the data tables into graphs. Along with the understanding, we can observe some anomalies that need to be adjusted to improve understanding of the classifier.

Having all the variables ready for use, we can create new variables using Data Engineering and be able to add information in the information already available. This process can bring great gains for the performance of the model.

Variables need to be cleaned because the correlation between them can disrupt the performance of the algorithm as well as variables with a lot of missing value can bring wrong conclusion. The Feature reduces process can handle this situation by deleting missing and correlated variables.

After all process of data engineering and data select, a final table is generated with all the information available together to be used by the classifier. Initially, the classifier was trained with few variables to be our benchmark, after that, it was added information and new techniques to get the best roc curve.

When applying several techniques together of machine learning we can find problems like computational capacity necessary to process all the data and make all the possible adjustments but the final result fit my expectation. We can see in conclusion section the applied techniques makes the performance increase above the random classifier and the benchmark classifier.

### 5.3 Improvement

Today the machine learning field is very vast and offers numerous techniques to make a classification. Use new techniques can improve the solution found in performance and processing technique. In real problem solutions, we have to work with the variable time, which is the time available to give a solution to a problem and also have another very important variable that is processing power. Both problems are often faced by professionals from several companies when they want to get an optimized result with machine learning.

If we use a higher degree of polynomial in feature engineering process it is possible to obtain more information to improve the power of the classifier as well to use more parameters to find the optimum parameter for the solution. I think it is possible to improve the result because we can see right AUC values in the rank of best results in kaggle. Overall, the best results come from teams with computational power and more mature knowledge on the subject (it is not a rule), probably the next project I will have a bigger maturation and better solution.