

**Home Credit Default Risk**

**Data Science Project Protocol**

*Author(s):*

*Nimrod Ken Dror*

*Lev Solodkin*

# 

Table of Contents

[Introduction 3](#_Toc4368475)

[Methodology (Project design) 3](#_Toc4368476)

[**General** 3](#_Toc4368477)

[**Treating imbalance** 4](#_Toc4368478)

[**Data exploration strategy** 4](#_Toc4368479)

[**Treating the data** 4](#_Toc4368480)

[**Models** 4](#_Toc4368481)

[Notebook Detailed Explanation 6](#_Toc4368482)

[**Exploratory Data analysis:** 6](#_Toc4368483)

[**Data Encoding:** 10](#_Toc4368484)

[**Missing Values** 11](#_Toc4368485)

[**Outlier Handling** 13](#_Toc4368486)

[**Flat File Generation** 21](#_Toc4368487)

[**Feature Importance** 22](#_Toc4368488)

[**Model Selection** 23](#_Toc4368489)

[**Partition of the Dataset** 23](#_Toc4368490)

[**Balancing of the Dataset** 23](#_Toc4368491)

[**Hyper parameter fine-tuning** 24](#_Toc4368492)

[**Final model performance** 24](#_Toc4368493)

[**Conclusion** 25](#_Toc4368494)

# Introduction

In today’s loan industry, there is a huge gap between what financial institutes know about their loaning customers, their credit history and their ability to repay their debts. This leads to information gap that causes inefficiency in loan providing and high loan interest rates. This also prevents many adequate and potential customers from applying for a loan, thus causing financial loses to both sides.

The goal of this project is to create, based on historical loan data, a Machine Learning based model that will aid financial institutes in the process of screening and approving potential customers.

This model will try to predict whether a user will be able to repay his debt within the required timeframe or not.

Based on our current knowledge and assessment, we predict that the outcome will be influenced by some or all of the following factors:

* Employment status
* Gender
* Marital status & number of kids (if any)
* Age
* Past loans
* Annual income from salary and other sources
* Assets owned by the client

Apart from the above stated factors, we believe that there may be additional factors that have to be considered in order to gain an accurate prediction regarding the possibility that a loaner will default.

In order to answer this question we will perform Machine Learning based analysis on a dataset of actual users and loan history provided by Home Credit Group, as provided by [Kaggle](https://www.kaggle.com/c/home-credit-default-risk/).

# Methodology (Project design)

## **General**

The data used in our project is a csv file that contains the details of loans taken from Home Credit Group, which we obtained from a [Kaggle](https://www.kaggle.com/c/home-credit-default-risk/data) competition.

The shape of our dataset before any changes made is 307,511 rows and 122 columns.

In this dataset, we can see details regarding different aspects of each loan taker, such as - age, annual income, number of cars and many more. Additionally, the data tells us regarding each loan if this loan defaulted or was returned on time - this will be our final outcome variable, where 0 will mean that the loan was returned on time, and 1 will mean that it defauted.

## **Treating imbalance**

A major issue that we will be facing is the imbalance of the dataset. As expected, most of the bank’s data is comprised of entries of loans that were returned on time (otherwise the bank would go bankrupt…), therefore more than 90% of the dataset is comrpised of non-defaulted entries, and less than 10% shows defaulted loans. Such an imbalance may cause severe distortion to our results, and will require some handling through the data itself or through model generation.

## **Data exploration strategy**

We plan to explore variables that we believe are key variables that affect the outcome of whether the loan defaulted or not. Exploration will be done using Python’s matplotlib and Seaborn libraries, focusing on pie charts, histograms and boxplots to identify key trends in our data, possible outliers and also problematic values that might require our attention.

We will encapsulate some of the code with custom-made functions that will render a clean and reusable code.

## **Treating the data**

Our data contains multiple variables with missing values. We will be using techniques to identify and quantify these variables and the % of missing data in each of them. After finding out the proportion of missingness and also the way the missing data behaves - Missing at random, completely at random or Not at random - we will decide whether we delete the rows, the columns or perform some feature engineering to represent the missingness.

We will be using techniques such as OLS to find out the connection of missing data to non-missing data within the dataset to determine the missingness types.

Please refer to Github for the Data Retrieval Protocol for our dataset

## **Models**

Our data will be divided into train and test and dev, on a proportion of 80% train and 20% test.

And then divide the train Dataset to train and dev. The dev Dataset will be used for model selection and final model testing.

We will keep our Target variable in the same proportions in the train/test/dev Datasets (stratified) in order to avoid imbalances in the data which will severely affect our models.

As our Target variable is 0 or 1, we will be using classification models such as XGBoost, LightGBM, Randomforest etc. The models will be trained using train sets that will be manipulated by balancing techniques such as oversampling, under sampling, SMOTE, and also by using some of the internal model features that deal with imbalanced data.

Our winning features will be chosen by using feature importance in Random Forest, XGBoost and etc.

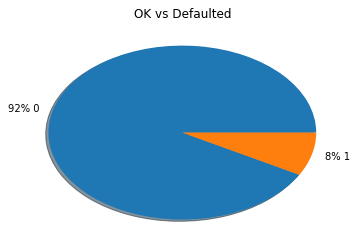
Each of our proposed models will then be run using basic parameters and the AUC (Area under Curve) for each model will be checked and compared. We will take the winning model and then continue tweaking it through its hyperparameters to receive a higher AUC result.

# Notebook Detailed Explanation

## **Exploratory Data analysis:**

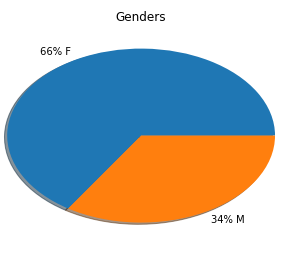
We start the notebook with plotting the following features in order to get a better understanding of our Dataset:

* **Ok VS defaulted clients**:



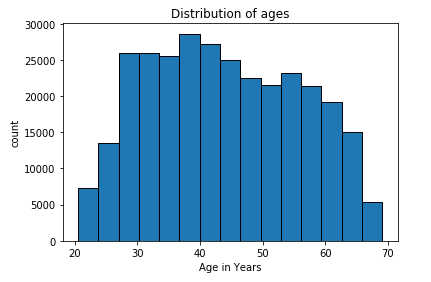
As we mentioned before and can see now, this Dataset is highly unbalanced and we will need to address that issue later on.

* **Clients gender**:

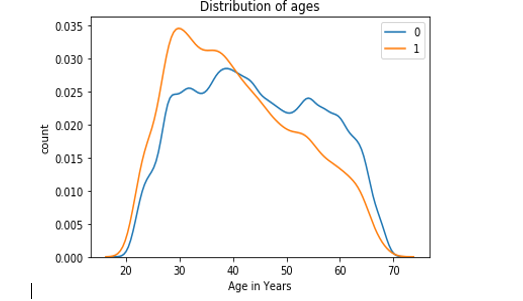


Interesting to see that there is a majority of women and not man as we expected.

* **Distribution of ages per count:**

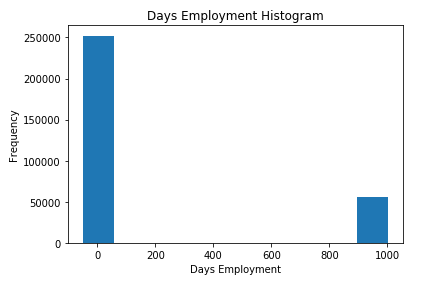


* **Distribution of ages per target**:



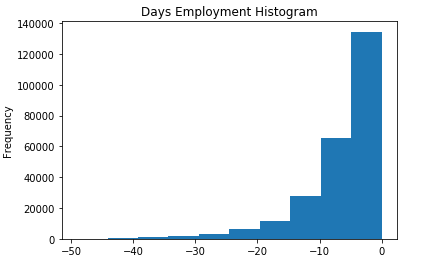
We can clearly see that the younger the client is, the more likely of him to default on a loan.

* **Distribution of days employment:**



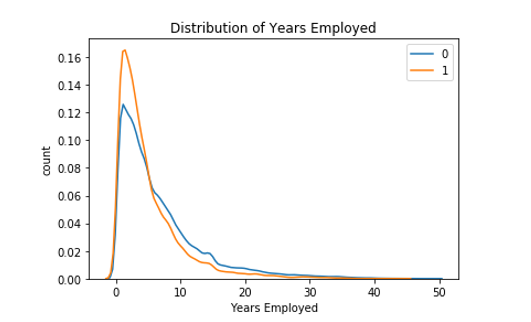
We can clearly see that there is an outlier in the data, after checking the data we noticed that there are some records that have 365243 days of employment (over a 1000 years), we decided to replace those values win null values and deal with them later on.

After removing that outlier, we plotted again the distribution of days employment to see that that Dataset has a normal distribution:



\* Note that the values are negative because they describe the amount of days the client was employed before he or she took the loan

* **Distribution of Days employed per target:**



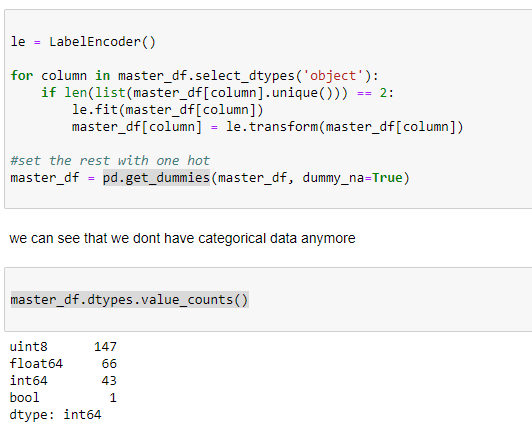
As expected- the less the loaner work, the more likely of him to default on a loan.

## **Data Encoding:**

Before we address the missing values phase, we need to treat all categorical data in the Dataset.

There are 16 features that comprised of categorical data that must be addressed.

We decided to use LabelEncoding for categorical data that have only two values such as male/female etc.. , and for those that have over two values we will use one hot encoding.

No categorical data, we can move on to missing values handling. 

## **Missing Values**

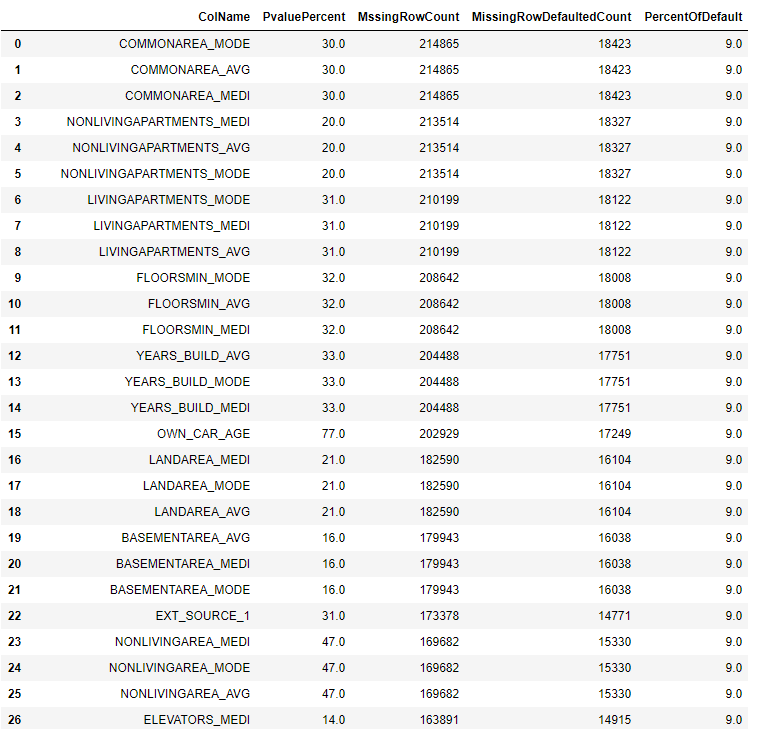
There are 62 columns with missing values in our Dataset.

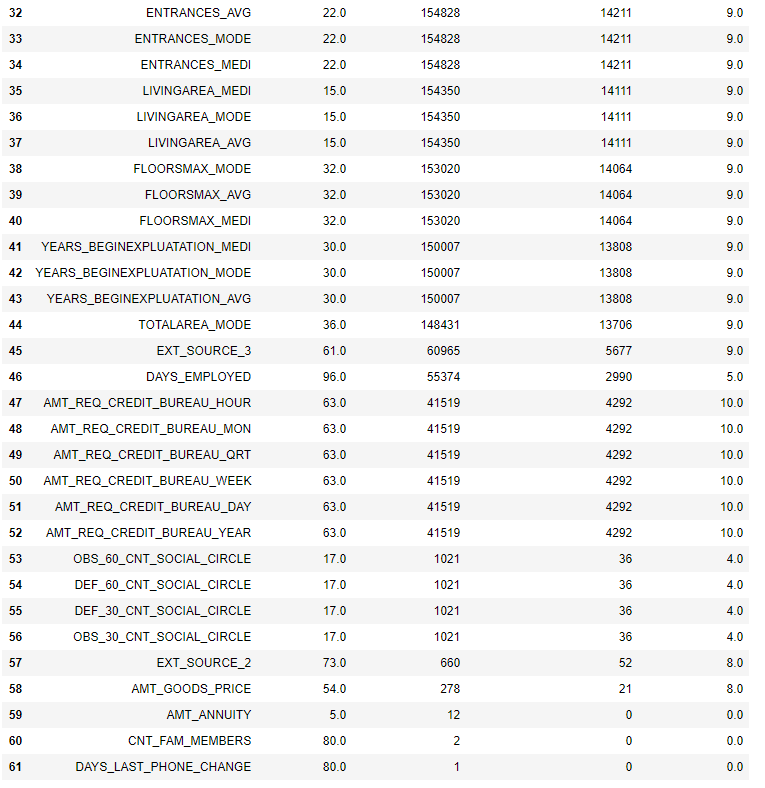
Before we decide which method will be best for missing values, we need to check the missingness type and amount of missing data.

For that usage we created a method we call **explore\_missing\_values**() that will test the missingness type using OLS and will return a Dataframe with the following columns:

* **ColName**.
* **PvaluePercent** – the percentage of columns with no missing values that indicate a strong relation (pvalue) to the missing column.
* **MssingRowCount**.
* **MissingRowDefaultedCount** – how many missing rows are defaulted, that is a very important piece of information because our Dataset is highly unbalanced.
* **PercentOfDefault** – the general percentage of all defaulted records.

As we can see The pvalue percent is high on all columns so we have to treat them as missing not at random.





After debating, we decided to remove rows that have less than 1022 missing values and to do one hot encoding for the rest of the missing columns.

The one hot encoding will be based on using one of the following methods

* **Quartile encode** – creating 5 columns that represent the 0-25%, 25-50%, 50-75%, 75-100% and is nan. For that usage we wrote the **encode\_quartile**() function.
* **Decision tree classification** - based on DecisionTreeClassifier algorithm. For that usage we wrote the **one\_hot\_by\_classify()** function that cuts the data to categories made by DecisionTreeClassifier.

Note that this function also decide which is the best max depth by checking the cross\_val\_score for each depth in the given depth range (default is 1-6).

We decided after several testing that the best way is to use the **one\_hot\_by\_classify()** function for create the one hot encoding groups.

## **Outlier Handling**

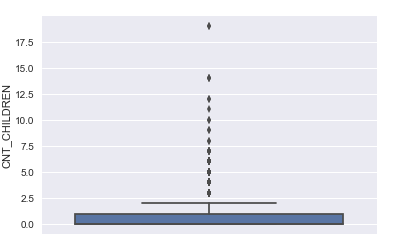
For the usage of detecting outliers, we ran the **zscore\_outlier()** function for checking if tour features containing outliers.

The following represents the outlier detection

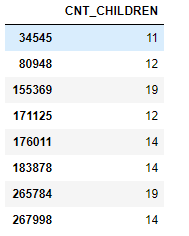


We found 13 features that contain outliers. Most outliers look like data that we cannot delete so let's use boxplot to take a closer examination at each feature.

* **CNT\_CHILDREN:**

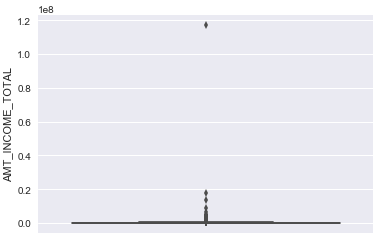


Let's see who has more then 15:



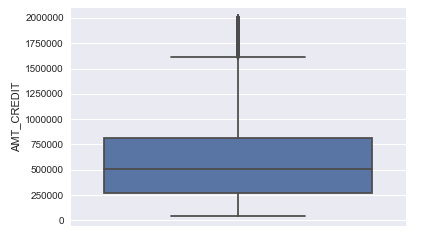
We decided to leave that one.. though 19 children sound farfetched.

* **AMT\_INCOME\_TOTAL:**



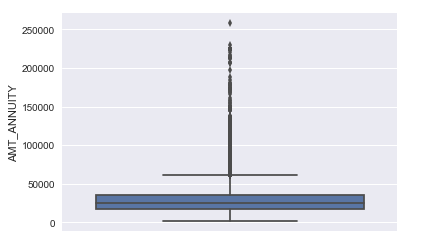
Only 224 records of people have an income over 1 million dollars a year and 13 of them defaulted on the loan. We will remove all 224 people

* **AMT\_CREDIT:**



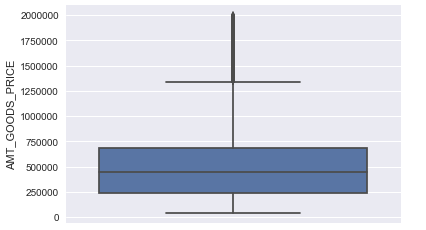
We will be removing 200 records of people that have amount of credit over 2000000.

* **AMT\_ANNUITY:**



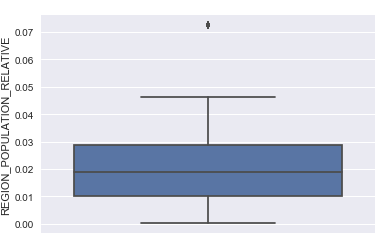
We will be removing 369 records of people with annuity over 10000.

* **AMT\_GOODS\_PRICE:**



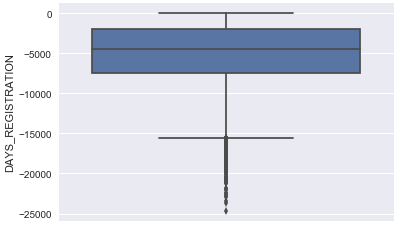
We found out that previous record removal also removed outliers from this column, so all records that have AMT\_GOODS\_PRICE over 1500000 do not exist anymore.

* **REGION\_POPULATION\_RELATIVE**



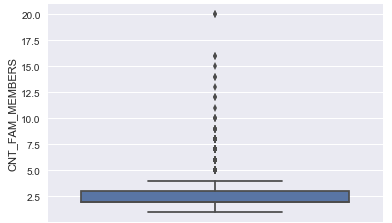
There are 7938 records with REGION\_POPULATION\_RELATIVE over 0.06 and 330 of them were defaulted. Not sure that it’s the right move but we decided to remove them.

* **DAYS\_REGISTRATION:**



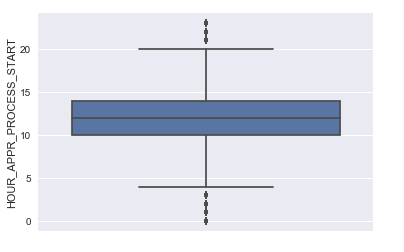
Not touching this one.

* **CNT\_FAM\_MEMBERS:**



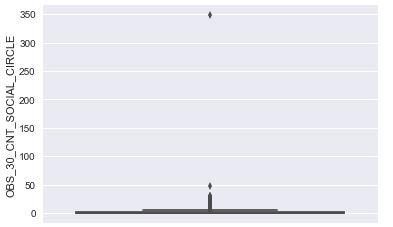
Not touching this one some families can be really big.

* **HOUR\_APPR\_PROCESS\_START:**



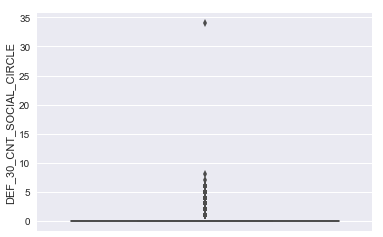
Not touching this one.

* **OBS\_30\_CNT\_SOCIAL\_CIRCLE:**



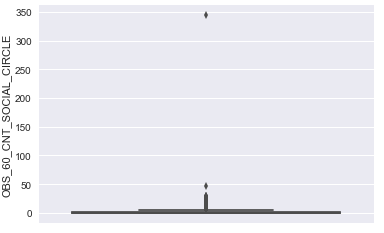
Just one record – remove.

* **DEF\_30\_CNT\_SOCIAL\_CIRCLE ASD:**



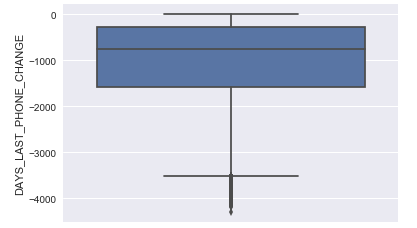
Previous record removal removed this outlier as well.

* **OBS\_60\_CNT\_SOCIAL\_CIRCLE ASD:**



Previous record removal removed this outlier as well.

* **DAYS\_LAST\_PHONE\_CHANGE:**



Previous record removal removed this outlier as well.

## **Flat File Generation**

We saved two version of flat files

* **/data/flat\_file.zip** – no outlier handling.
* **/data/flat\_file\_no\_outliers.zip** – with outlier handling.

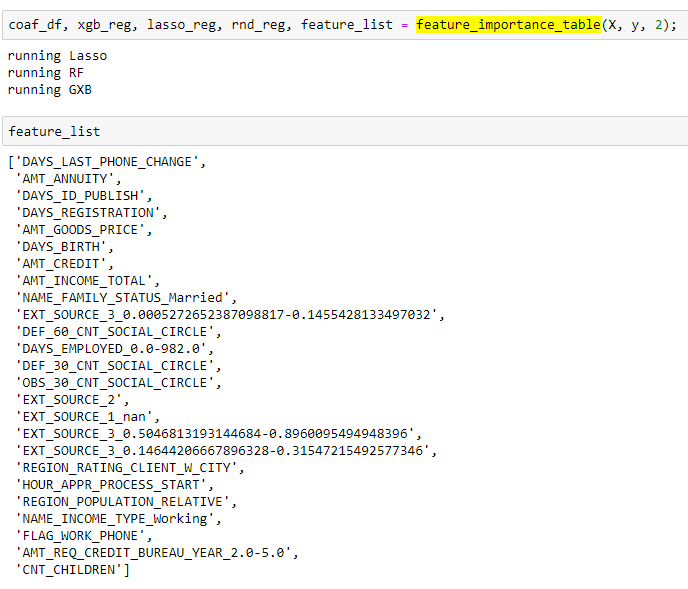
The next phases will be experimented with both files. Later on we will decide that the flat file with outlier handling yield better predictions.

## **Feature Importance**

For feature importance we created the **feature\_importance\_table()** function that runs the following models on the Dataset:

* Lasso.
* XGBClassifier.
* RandomForestClassifier

The function returns a Dataset that has the features selected minimum 2 of 3 models as the best features.

The following 25 features were selected:

## **Model Selection**

We will test the following classification models for our final predictions:

* XGBClassifier.
* RandomForestClassifier.
* LGBMClassifier.

But before we can start we need to partition the Dataset.

### **Partition of the Dataset**

The Dataset will be partitioned to train, dev and test.

We will be using only the train and dev Datasets for model selection and final model Hyper parameter fine-tuning and after the final model is complete we will use the test Dataset to check the final model accuracy.

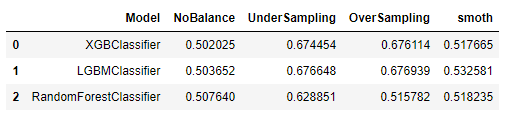
For Dataset partition we used **train\_test\_split**() function with test\_size=0.20 paramater.

### **Balancing of the Dataset**

We will use the following data balance technics:

* RandomUnderSampler
* RandomOverSampler
* SMOTEENN

After the data was partitioned and balanced we ran each Dataset with each model listed above and came out with the following results.

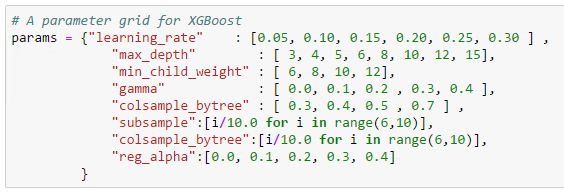


Although LGBMClassifieris the highest ranking model we decided to use GXBClassifier with OverSampling as our chosen Model.

## **Hyper parameter fine-tuning**

For hyper parameter fine tuning we will be using **RandomizedSearchCV** that will run 400 iterations (20 folds \* 20 param combinations).

After researching online, the following hyper parameters where selected with specific ranges for the test.



We configured the model to run with one thread, and the **RandomizedSearchCV** algorithem to run 16 concurrent jobs.

## **Final model performance**

## **Conclusion**

This project has demonstrated to us the importance of using an organized workflow and methodology when attending such data science projects.

We learned the importance of treating missing values and balancing of our Datasets, and also - how important it is to explore the data prior to modelling stage.

Considering the fact the Kaggle competition winners reached ~80%, we believe that if we had more computational resources, it would have enabled us to reach a better score than our current score.

We are looking forward for our next project and believe that many of the tools acquired on this project will serve us well in the projects to come.