Credit Team Project Proposal
credit@chalearn.org - https://github.com/Didayolo/credit
Give me some credit

Eden Belouadah (eden.belouadah@yahoo.fr), Adrien Pavao (adrien.pavao@u-psud.fr),
Ghiles Sidi Said (ghiles.sidi-said@u-psud.fr), Taycir Yahmed (taycir.yahmed@telecom-paristech.fr)

October 2017

1 Background

In an economic and financial world full of risks and challenges, the task of detecting potential threats has become an urgence, especially in the banking sector. Machine learning is an awesome tool to analyze customer's data and make accurate decisions. This is why it is increasingly used in this field[1].

Figure 1: To which borrower should we give some credit ? Regarding what ?

The goal of this kaggle challenge (*Give me some credit - Sep 19, 2011*) is to be able to decide if a loan should be granted or not by taking into account some informations about the borrower. This project treats a serious real-world problem of fraud and risk detection. So, resolving such problems helps improve the security in the financial domain, and leads to more confidence between customers and companies.

[1] http://financeandriskblog.accenture.com/analytics/where-can-machine-learning-be-applied-to-improve-banking-performance

2 Material and method

The main task is to exploit informations about borrowers expressed as 10 features of different types (integer, pourcentage, real) to predict the class whose type is binary, it means that it can take the values *true* or *false*. The description of the different features is given in the file *Data Dectionary.xls*. the table 1 gives more information about the dataset :

Number of features	10
Number of examples	150000

Table 1: Number of features and examples

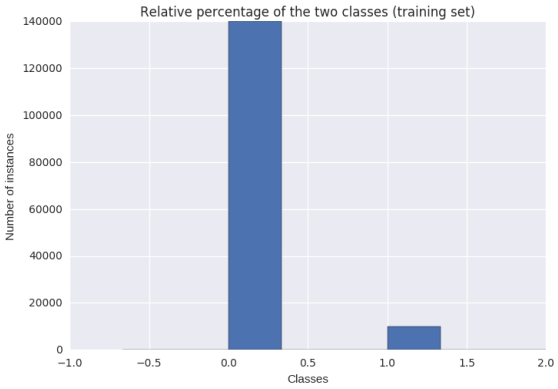

Figure 2: Classes distribution in dataset

We see that the most of time, the borrowers are

classified as granted ones since they will not experience financial distress in the next two years.

The split of dataset between training set, test set and validating set is given in Table 2.

Training set	Test set	Validating set
100000	25000	25000

Table 2: The split of the dataset

Knowing that we are face to unbalanced dataset, we used a *Stratified Shuffle Split* to ensure the same classes distribution for all data subsets.

Furthermore, we notice that the two features *MonthlyIncome* and *NumberOfDependents* have a considerable percentage of missing values (19.82% for *MonthlyIncome* and 2.62% for *NumberOfDependents*). To handle this, we can use various methods including statstical imputation and regression imputation.

Here we have the features importances ranking using Gini coefficient :

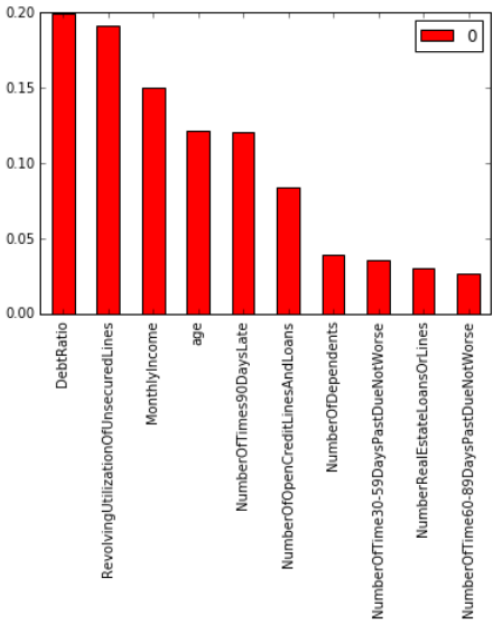

Figure 3: Features importances

2.1 Evaluation metric

Because the classes are not balanced, The *Area Under the ROC curve (AUC)* seems a very good metric to evaluate the model since this metric is insensitive to the inbalance of the classes. The most important is to avoid false positives, or in other words to attribute a credit to people that are going to experience financial distress in the next two years.

3 Preliminary results

In the Kaggle competition, the winners scored around 0.86 as of the AUC. In the pipeline described below, we reach over 0.85 on the Kaggle leaderboard. Besides, we observed that the score on our validation set and the one the leaderboard differ significantly. So, we make the assumption that the data used in the leaderboard (test set) have some specific distribution.

3.1 Pipeline

1. Load the data

2. Prepare the data : removing the target variable from the feature matrix and **imputing the missing values**. Note that for now, we use filling by zero but for the final approach, we intend to perform statistical and regression imputation.

3. Train/validation splitting: Since we have an imbalanced dataset, we used stratified splitting which returns **stratified randomized folds**. The folds are made by preserving the percentage of samples for each class.

4. Learning the model: We used a **soft voting classifier**, that predicts the class label based on the argmax of the sums of the predicted probabilities. The base estimators are non-tuned models including *LogisticRegression, RandomForestClassifier, GradientBoostingClassifier, ExtraTreesClassifier, MLPClassifier, AdaBoostClassifier, GaussianNB* and *QuadraticDiscriminantAnalysis*. Furthermore, the soft voting depends on different weights, corresponding to the f1-scores (we can also consider the AUC) calculated on the validation set. Thus, the contribution of each model in the final decision is weighted by its ability to predict the class labeled *'1'*.

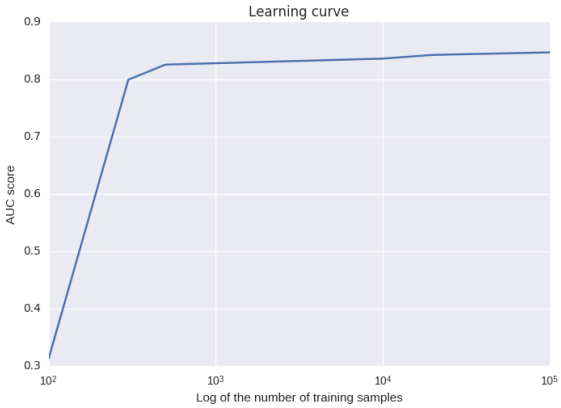

Figure 4: Learning curve