Final report - CS229: Dynamic loan default prediction

- Maxime Rivet; SUNetIDnumber: 06228390
- Marc Thibault; SUNetID number: 06227968
- Mael Trean; SUNetIDnumber: 06228438

I - Motivation

Introduction

Predicting the outcome of a loan is a recurrent, crucial and difficult issue in insurance and banking.

The objective of our project is to predict whether a loan will default or not based on objective financial data only.

We used a dataset provided by LendingClub concerning almost 1 million loans issued between 2008 and 2017.

Since a default is far more costly for a loan issuer than a missed loan, we focused on maximizing the F-score as an evaluation metric for our algorithm.

Using a very structured pipeline to load and test algorithm, we reviewed most of the classification Machine Learning strategies to extract information from the very noisy data provided by LendingClub.

Data

It consists into approximately 800,000 samples of loans granted by the company, with the full set of informations about the borrower, the history of payments and the outcome of the loan.

The dataset is quite clean and the figures can be considered as ground truth, but lots of columns are either irrelevant, very sparse or non informative.

Moreover, the dataset is very unbalanced, with approximately 17% of loans considered as defaulted.

Since the objective is to predict the outcome from the informations gathered at the signature of the loan, we cannot use the data concerning the history of payments or the current situation of a loan.

Excluding features for which the information is incomplete, or uninformative, we get a total of 19 features, that cover personal information (credit grade, income, housing status, ...) and credit information (amount, interest rates, ...).

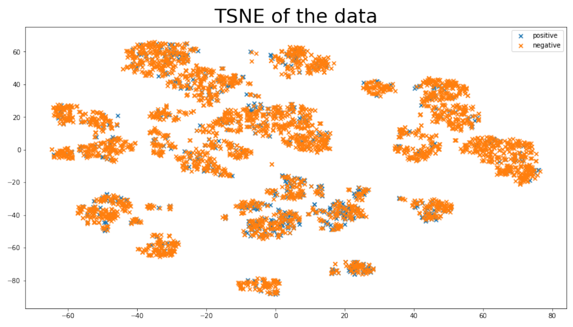
Accuracy is not well-suited for our problem. The unbalance of the classes would lead an algorithm to never predict a default. $F1$-score allows us to quantify a good prediction on both precision and recall.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

II - Method

Data exploration

The dataset is not well separated, as this TSNE denotes. Also, the class unbalance is significant, and no clear separator appears.



Models

The main models we used are:

- Regularized Logistic Regression, from `sklearn`
- Gradient Boosting Decision Trees, from `xgboost`

Our approach consists in taking class unbalance into account when training our learning algorithm, with the three following methods:

- Force a fixed class balance p in the training set by resampling,
- Change the limit α above which the algorithm outputs 1 from the `predict_proba`: $z_i = 1_{(y_i > \alpha)}$,
- Balance the loss function to penalize the midprediction of distinct classes differently: $L_i = \omega_1 \; y \; log(\hat{y}) + \omega_0 \; (1-y) \; log(1-\hat{y})$.

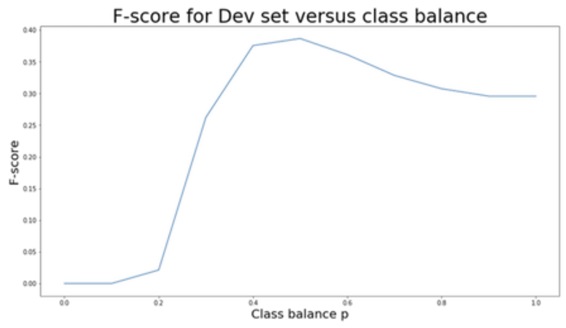
We then grid over the parameters of these methods:

- fixed class occurrence ratio p,
- limit α,
- penalization parameters ω_0, ω_1

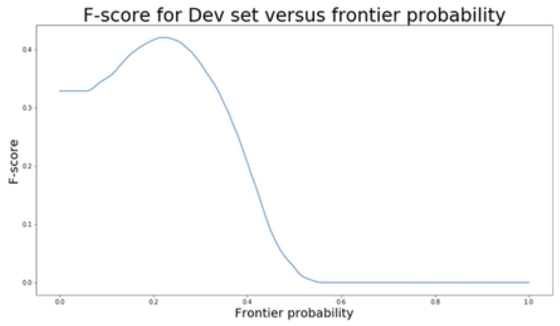
We then optimize the F-score we obtain on the dev set, in order to make sure that we do not overfit.

III - Results of individual classifiers

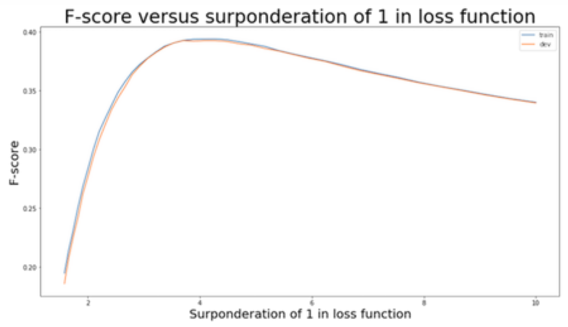
Our three cross-validation methods yield significantly good results, beating the random choice on this extremely intricated dataset.



We reach a F-score of 38% by balancing classes upon training.



We reach a F-score of 41% by shifting the decision frontier.



We reach a F-score of 39% by penalizing labels distinctively.

IV - Combining classifiers

We added a regression of regressions approach, which consists in:

- Train several classifiers
- Output `predict_proba` for each sample and each classifier
- Run a second classifier with the outputs of individuals classifiers as features

The stacking of different models is also quite promising, since it allows to take advantage of the best classification possibilities of each model.

The results are better on test data, as we reach 43% of F-score.

Accuracy matrix:

Reality vs. Predicted	**Fully paid**	**Defaulted**
Fully paid	84,944	37,294
Defaulted	12,032	18,092

V - Further enhancements

The exploration of the stacking of models sounds promising, and would probably be the best exploration axis if we had more time to spend on this project.

Dynamic predictions by updating them when payments are due might allow a more precise risk management for the issuer. Time and status of payments can bring important insight in order to spot default risk

A lot of the initial data has been rejected. It represents refused loans by LendingClub for which we did not have the final status because they have never been opened. By doing so, the data ended being quite uniform, knowing that only approved loans where in it. A possible approach would be to include all rejected loans as defaulted in the training set. It might be able to solve the imbalance of classes and to train better on rejected loans.

References

Introduction to stacking :

http://blog.kaggle.com/2016/12/27/a-kagglers-guide-to-model-stacking-in-practice/

Kaggle dataset and best preprocess :

https://www.kaggle.com/vincepota/predicting-customers-who-will-charge-off

Dealing with unbalanced datasets :

https://svds.com/learning-imbalanced-classes/

Contributions

For the first steps, Mael worked on the code architecture to deliver a clean pipeline and Maxime gathered and preprocessed the data. He gave a nice representation of the data with T-SNE algorithm. Marc tried several classifiers and identified with Maxime the class imbalance issue. Mael implemented the last classifier of the project, which uses the output of the other classifiers.