Loan Repayment Prediction

Artificial Intelligence based Model

Table of Contents

[1 Introduction 2](#_Toc89469998)

[2 Data 2](#_Toc89469999)

[2.1 Insights 4](#_Toc89470000)

[2.2 Pre-processing 5](#_Toc89470001)

[3 Model 6](#_Toc89470002)

[3.1 Ensemble Model 6](#_Toc89470003)

[3.2 Proposed Method 7](#_Toc89470004)

[4 Results 8](#_Toc89470005)

[4.1 Curves 8](#_Toc89470006)

[4.2 Correlation and Confusion Matrix 8](#_Toc89470007)

[4.3 Partial Dependence Plots 10](#_Toc89470008)

[5 Conclusion 10](#_Toc89470009)

Loan Repayment Prediction – Ensemble Model

The field of artificial intelligence is conquering every aspect of our life. The world is advancing at a rapid pace and artificial intelligence is playing an important role in this pace. The businesses are becoming more and more dependent on the artificial intelligence and machine learning based algorithms. The two most critical questions in the lending industry are: 1) How risky is the borrower? 2) Given the borrower's risk, should we lend him/her? These risks involve great investments and sometimes even entire structure of the business. This report discusses the loan repayment prediction using the artificial intelligence.

# Introduction

The two questions being asked have the answers. The answer to the first question determines the interest rate the borrower would have. Interest rate measures among other things (such as time value of money) the riskiness of the borrower, i.e. the riskier the borrower, the higher the interest rate. With interest rate in mind, we can then determine if the borrower is eligible for the loan. Investors (lenders) provide loans to borrowers in exchange for the promise of repayment with interest. That means the lender only makes profit (interest) if the borrower pays off the loan. However, if he/she doesn't repay the loan, then the lender loses money.

The AI and ML have entered the business industry with its flying colors. All the business categories are using these advance algorithms in one or the other way and directly or indirectly.

# Data

We'll be using publicly available data from [lendingclub.com](https://www.lendingclub.com/). The data covers the 9,578 loans funded by the platform between May 2007 and February 2010. The interest rate is provided to us for each borrower. Therefore, the second question is address indirectly by trying to predict if the borrower will repay the loan by its mature date or not. Through this exercise we'll illustrate three modeling concepts:

* What to do with missing values
* Techniques used with imbalanced classification problems
* Illustrate how to build an ensemble model using two methods: blending and stacking, which most likely gives us a boost in performance

Following are the features of the dataset:

|  |  |
| --- | --- |
| **Features** | **Explanation** |
| credit\_policy | 1 if the customer meets the credit underwriting criteria of LendingClub.com, and 0 otherwise. |
| purpose | The purpose of the loan such as: credit\_card, debt\_consolidation, etc. |
| int\_rate | The interest rate of the loan (proportion). |
| installment | The monthly installments ($) owed by the borrower if the loan is funded. |
| log\_annual\_inc: | The natural log of the annual income of the borrower. |
| dti | The debt-to-income ratio of the borrower. |
| fico | The FICO credit score of the borrower. |
| days\_with\_cr\_line | The number of days the borrower has had a credit line. |
| revol\_bal | The borrower's revolving balance. |
| revol\_util | The borrower's revolving line utilization rate. |
| inq\_last\_6mths: | The borrower's number of inquiries by creditors in the last 6 months. |
| delinq\_2yrs | The number of times the borrower had been 30+ days past due on a payment in the past 2 years. |
| pub\_rec | The borrower's number of derogatory public records. |
| not\_fully\_paid | indicates whether the loan was not paid back in full (the borrower either defaulted or the borrower was deemed unlikely to pay it back). |

## Insights

Following are some insights into the dataset:

**Data types:**

**-----------**

**credit\_policy int64**

**purpose object**

**int\_rate float64**

**installment float64**

**log\_annual\_inc float64**

**dti float64**

**fico int64**

**days\_with\_cr\_line float64**

**revol\_bal int64**

**revol\_util float64**

**inq\_last\_6mths float64**

**delinq\_2yrs float64**

**pub\_rec float64**

**not\_fully\_paid int64**

**dtype: object**

**Sum of null values in each feature:**

**-----------------------------------**

**credit\_policy 0**

**purpose 0**

**int\_rate 0**

**installment 0**

**log\_annual\_inc 4**

**dti 0**

**fico 0**

**days\_with\_cr\_line 29**

**revol\_bal 0**

**revol\_util 62**

**inq\_last\_6mths 29**

**delinq\_2yrs 29**

**pub\_rec 29**

**not\_fully\_paid 0**

**dtype: int64**

It looks like only one categorical feature ("purpose") is present. Also, six features have missing values (no missing values in labels). Moreover, the data set is pretty imbalanced as expected where positive examples ("not paid fully") are only 19%. These need to be handled and the next section is explaining their handling through different techniques.

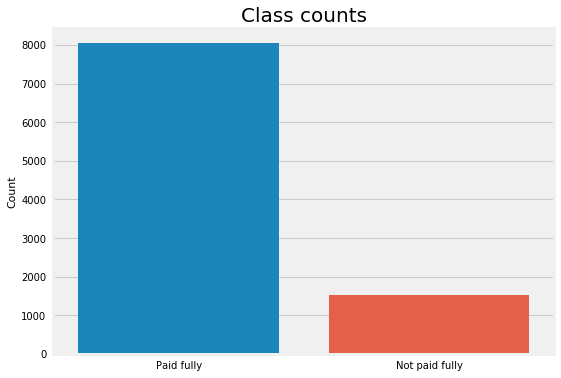


Figure 1: Categories in the data

## Pre-processing

Almost always real-world data sets have missing values. This can be due, for example, users didn't fill some part of the forms or some transformations happened while collecting and cleaning the data before they send it to you. Sometimes missing values are informative and weren't generated randomly. Therefore, it's a good practice to add binary features to check if there are missing values in each row for each feature that has missing values. In our case, six features have missing values so we would add six binary features one for each feature. Below are some of the most common strategies for dealing with missing values:

* Simply delete all examples that have any missing values. This is usually done if the missing values are very small compared to the size of the data set and the missing values were random. In other words, the added binary features did not improve the model. One disadvantage for this strategy is that the model will throw an error when test data has missing values at prediction
* Impute the missing values using the mean of each feature separately
* Use Multivariate Imputation by Chained Equations (MICE). The main disadvantage of MICE is that we can't use it as a transformer in sklearn pipelines and it requires to use the full data set when imputing the missing values. This means that there will be a risk of data leakage since we're using both training and test sets to impute the missing values

# Model

The modelling is the main phase of any development project. The architecture of the model is defined according to the nature of the problem and its solution. In this project, ensemble method is used to predict the loan repayment.

## Ensemble Model

Ensemble methods can be defined as combining several different models (base learners) into final model (meta learner) to reduce the generalization error. It relies on the assumption that each model would look at a different aspect of the data which yield to capturing part of the truth. Combining good performing models they were trained independently will capture more of the truth than a single model. Therefore, this would result in more accurate predictions and lower generalization errors. - Almost always ensemble model performance gets improved as we add more models. - Try to combine models that are as much different as possible. This will reduce the correlation between the models that will improve the performance of the ensemble model that will lead to significantly outperform the best model. In the worst case where all models are perfectly correlated, the ensemble would have the same performance as the best model and sometimes even lower if some models are very bad. As a result, pick models that are as good as possible.

There are different ensemble methods construct the models in different ways. Following are some methods:

1. Blending

Averaging the predictions of all the models.

1. Bagging

Build different models on different datasets and then take the majority vote from all of them.

1. Boosting

This method builds the models sequentially. Each model learns from the previous one.

1. Stacking

Build k models called base learners. Then fit a model to the output of the base learners to predict the final output.

## Proposed Method

This project will be using Random Forest (bagging) and Gradient Boosting (boosting) classifiers as base learners in the ensemble model, we'll illustrate only averaging and stacking ensemble methods. Therefore, modeling parts would be consisted of three parts:

* Strategies to deal with missing values
* Strategies to deal with imbalanced datasets
* Build ensemble models

The ensemble models will be built using two different methods:

1. Blending (average) ensemble model. Fits the base learners to the training data and then, at test time, average the predictions generated by all the base learners.

* Use VotingClassifier from sklearn that:
* fit all the base learners on the training data
* at test time, use all base learners to predict test data and then take the average of all predictions.

1. Stacked ensemble model: Fits the base learners to the training data. Next, use those trained base learners to generate predictions (meta-features) used by the meta-learner (assuming we have only one layer of base learners).

# Results

This section describes the results of the project in various aspects.

## Curves

As we can see from the chart below, stacked ensemble model didn't improve the performance as evident from figure below

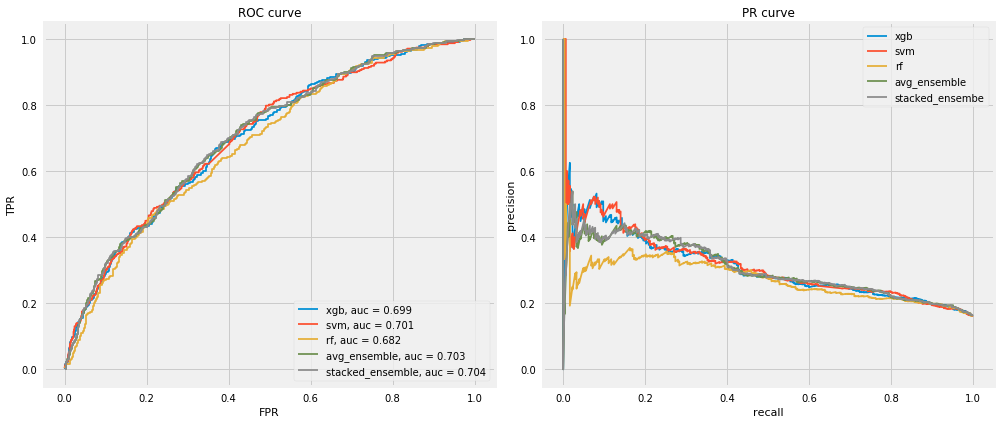


Figure 2: Curves of the model

## Correlation and Confusion Matrix

One of the major reasons of this unimproved performance are that the base learners are considerably highly correlated especially Random Forest and Gradient Boosting.

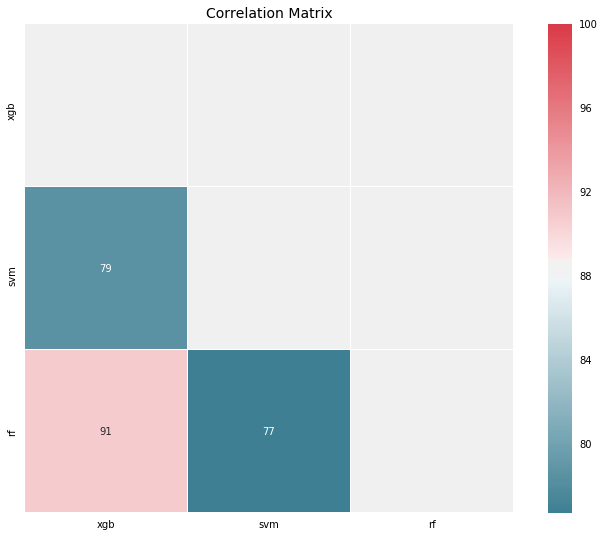


Figure 3: Correlation Matrix

In addition, with classification problems where False Negatives are a lot more expensive than False Positives, we may want to have a model with a high precision rather than high recall, i.e. the probability of the model to identify positive examples from randomly selected examples. Below is the confusion matrix:

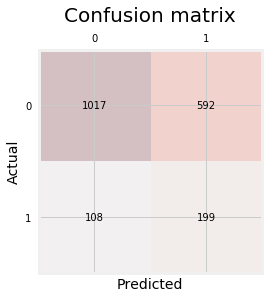


Figure 4: Confusion Matrix

## Partial Dependence Plots

Let's finally check the partial dependence plots to see what are the most important features and their relationships with whether the borrower will most likely pay the loan in full before mature data. The plot only the top 8 features to make it easier to read.

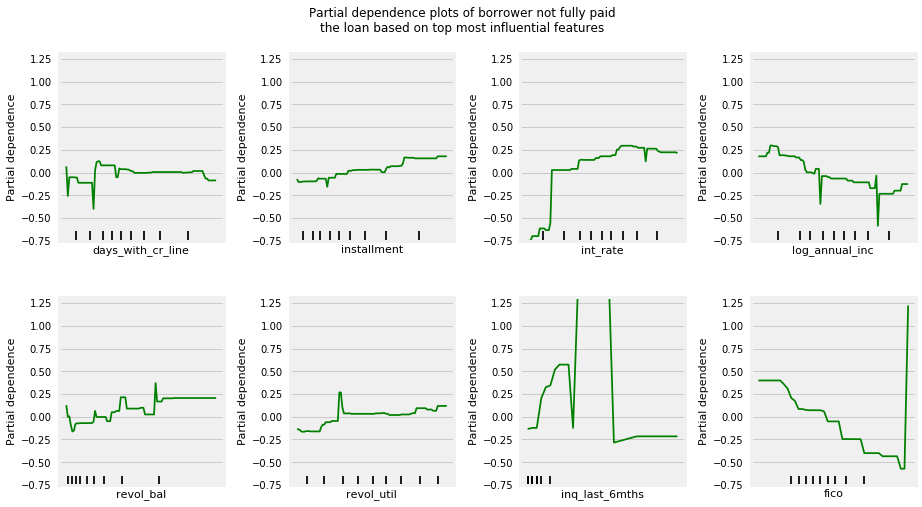


Figure 5: Partial Dependence Plots

As expected, borrowers with lower annual income and less FICO scores are less likely to pay the loan fully; however, borrowers with lower interest rates (riskier) and smaller installments are more likely to pay the loan fully.

# Conclusion

Most classification problems in the real world are imbalanced. Also, almost always data sets have missing values. In this project, we covered strategies to deal with both missing values and imbalanced data sets. We also explored different ways of building ensembles in sklearn. The ensemble models have performed quite fine but this performance can be improved by introducing dense neural layers in these models. Similarly certain work on pre-processing can also help to improve the model. In some classification problems, False Negatives are a lot more expensive than False Positives. Therefore, we can reduce cut-off points to reduce the False Negatives.