# Machine Learning Engineer Nanodegree

## Capstone Project

## The Lending Club: Loan default predictions

Serge Bouschet
June 1st, 2017

## I. Definition

**Project Overview**
The growth of peer to peer lending business over the last decade has helped millions of people achieve financial goals through online marketplaces. The Lending Club connects borrowers and investors by providing qualified loans, leveraging technology to lower the cost from traditional banks.

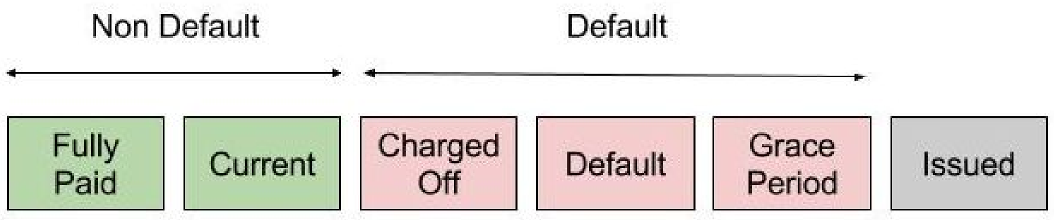
Loans offered through the platform are risk assessed to determine a credit rating and assign appropriate interest rates. However investors are still exposed to loan defaults. So what can data tell us about borrowers attitude to repayment and can we lower exposure to risk? The Lending Club's dataset offers a complete history of loan data through 2005-2017 with many features available to derive insight from this business.

**Problem Statement**

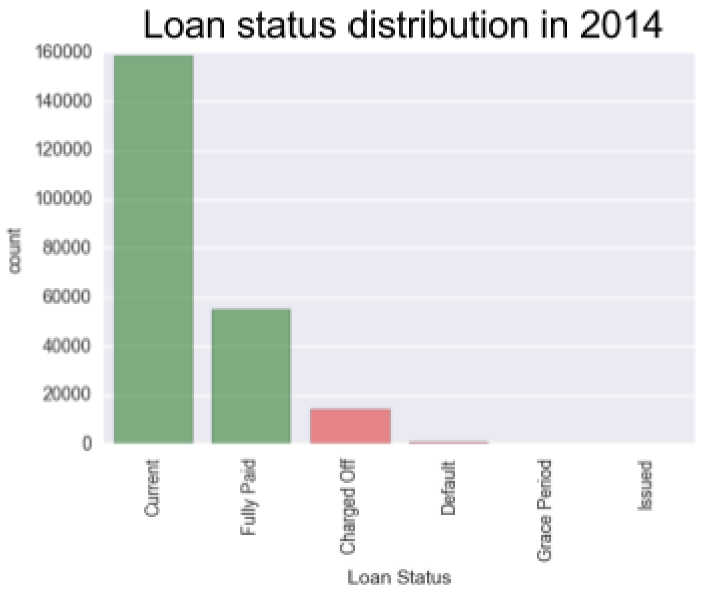
Measuring risk is a major concern in peer to peer lending, therefore the ability to predict Loan defaults from historical loan data available is the first objective we're going to address in this project. Predicting which loans are likely to default is beneficial in many situations.

For example, to assist investor's decision in selecting a loan. The Lending Club is a marketplace where investors make decision based the information available to them. Making accurate predictions based on this information could lower the risk associated with an investment, this is potentially a very useful tool.

The loan status in the dataset is a categorical variable which offers a very granular set of statuses, including statuses for Late repayments. This status can be either summarized to produce a binary label or can be used as continuous variable to explain the level of risk exposed in the transaction.



Default loan prediction on lending club's data presents characteristic of imbalanced dataset classes. Essentially, Current loan statuses account for a very large majority of all classes. A simple benchmark model gives a very high accuracy because most borrowers repay their loan on time and in full. This will need to be addressed when selecting the solution. An appropriate evaluation method is also required to address this.



**Metrics**

Our initial evaluation of loan default shows that we will be dealing with an imbalanced dataset (80 to 90% accuracy with benchmark model predicting that all loans are repaid).

With imbalanced asset classes, it is important to address the accuracy paradox from the outset and use other metrics beyond simple accuracy, such as precision, recall and F-beta score. The accuracy paradox effectively states that a model may have a higher predictive power with a lower accuracy level than another given model. This is explained by the fact that with loan default prediction, the value of predictions lies with finding true positives, therefore we no longer need to focus on accuracy (percentage of all correct predictions: true positive and true negative over number of observations). Since the data is imbalanced, a large number of true negatives isn't of value because our objective is to reduce risk and find loans that are likely to default. A high accuracy on true negatives will affect the overall accuracy score and may hide poor results on true positive rate. This is essentially why we need to study other measures such as precision and recall.

$Precision = TP \div (TP + FP)$
$Recall = TP \div (TP + FN)$

Recall is also known as Sensitivity or true positive rate. It gives a measure of the proportion of positives that are correctly identified, in other words it’s directly related to how accurate loan default prediction are.

F1 score is the weighted average of the precision and recall, where best value is 1 and worst 0. It is mathematically defined as follows:

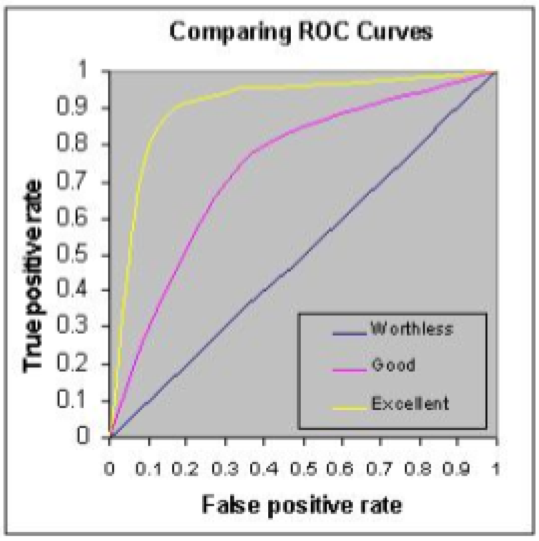
$$F1 = 2 \times (Precision \times Recall) \div (Precision + Recall)$$

We’ll use F1 score to understand if our model keeps a good balance between true positive and false positive predictions as we still want to retain the ability to select Fully Paid loans, which are good investments rather than reject them (False Negative) as well as reject loan that default.

ROC curve analysis is a useful tool to measure and understand performance of an algorithm with respect of precision and recall. In a ROC curve (receiver operating characteristic), the true positive rate (Sensitivity) is plotted as a function of the false positive rate (false alarms) for different cut-off points of a probability threshold. ROC analysis provides a tool to select optimal models and discard suboptimal ones. Performance is measured by evaluating the area under the ROC curve, considering that the optimal curve draws a right angle where 100% precision and %100 sensitivity is achieved ((0, 1) coordinates). The area measures discrimination which is the ability of our model to correctly classify default loans. The ideal area is 1 (surface of a 1:1 square), while 0.5 represents a diagonal line equivalent to random guess and therefore a worthless model.

When evaluating the ROC Curve, the steepness is also important since we want to maximise the the TPR (true positive rate, represented on y-axis) while minimising the FPR (false positive rate). A better curve should ideally hug the y-axis before the sharp bend.

Example of ROC Curves



Our goal for this project is to achieve 50% True Positive Rate (or Recall), predicting loan defaults correctly while retaining less than 10% False Positive. That means predicting at least one out of two loans default correctly. It's an arbitrary goal and this project will determine how challenging this target actually is.

# II. Analysis

### Data Exploration

Our raw dataset contains approved loans from 2007 until 2015, this represents 887,379 records with 74 features. The data is not evenly distributed over the timeline, it reflects the Lending Club's company growth over its short existence. More than two thirds of all observations have been captured for 2014-2015 period. In 2015, labels are not distributed as previous years, the proportion of Current status is much higher which indicates we're looking at data with many loan transactions still ongoing.

A data dictionary is provided with descriptions of all features which is very valuable for feature selection. Details about the loan in numerical and categorical formats are available: amounts, interest rate, installments, loan term, borrower's income and employment length as well as loan grade (a credit score), loan type, borrower's job title, loan purpose, are the main features. A couple of identifiers are available: id and member_id; they allow us to verify that each record corresponds to one loan rather than the history of loan statuses. Few descriptive features such as loan description and url can be used to derive new features and do text analytics. Geographical features are in this dataset too, zip code and address state. While time based features such as issue date, last payment date, next payment date and term can give some insight into the stage in the lifecycle the loan. It's a fairly exhaustive and diverse dataset that has been collected over a long period of time and as such can potentially reveal a lot of insight.

An important feature is obviously the loan status which classifies the state reached by a loan after its approval. It's important to understand that this status is not necessarily final, for example some "late" statuses classify the late repayment states at various points in time, these are time dependent and eventually progress into a "default" or "current" state. The full loan lifecycle is not given but fairly easy to derive. In this project our focus is to predict loan defaults so we will use this feature to derive a binary classifier (default, not default) from the set of status available. During preprocessing we will also decide which observations are kept to train our model based on this status.

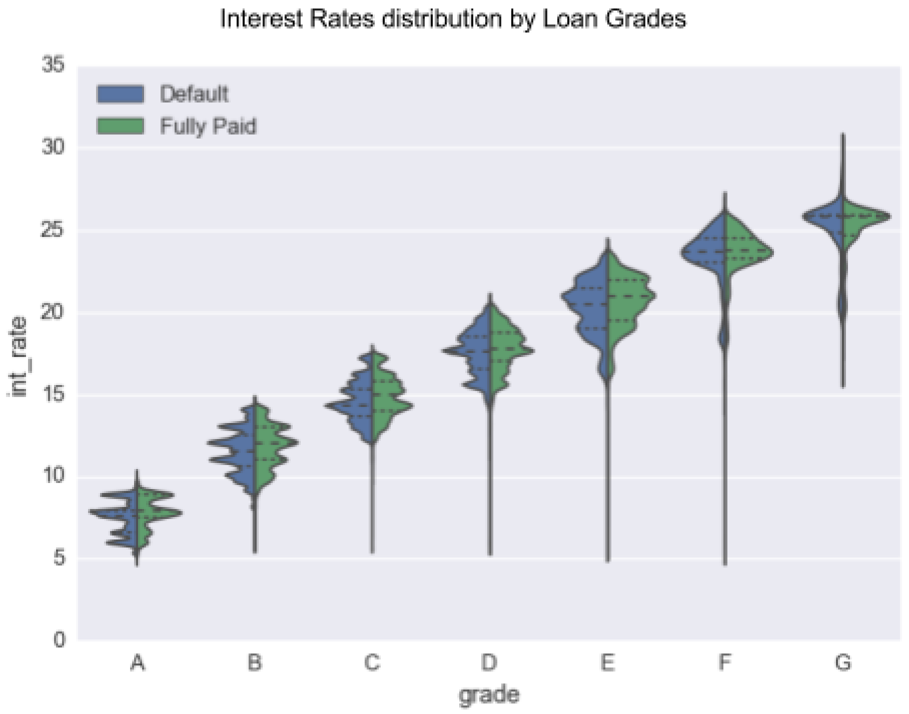
It is not specified however whether all of these features have been collected before submitting the loans on the marketplace of whether some of this information was added during or after repayment. We will pay attention to this during the feature selection process to ensure our model is not biased using data influenced by the outcome of the loan lifecycle.

Before running any machine learning classification with this, It is critical to understand

- Has the data for this feature been collected during or after loan repayment
- Does this data belong to the loan or the borrower's history
- Is there any strong correlation between some of these features
- What is the data quality for all features, missing values, outliers...

**Exploratory Visualization**

The goal of this investigation is to understand the distribution of loan status status over time and over the various loan purposes, understand data correlations and anomalies.

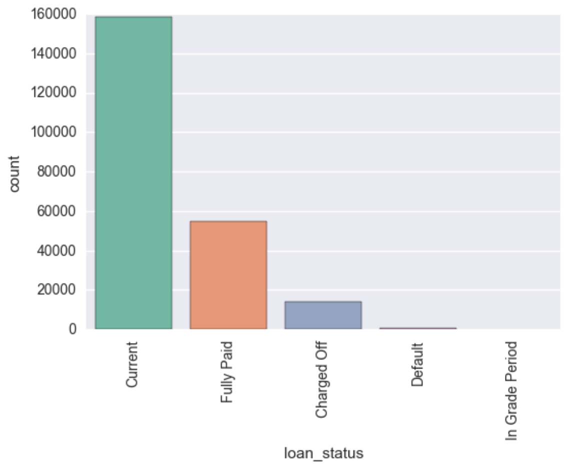


This chart shows the distribution of interest rates for loan each grade. We can analyse it by target class as well. This shows that the Lending Club is doing a lot of work on credit scoring to categorise loans and offer investors relevant categories of interest rates. There isn't a clear difference between loans fully paid and loans that defaults in terms of interest rates assigned; we might find a marginal tendency on riskier investments to default (i.e. those assigned a higher interest rate within the range offered under their grade).

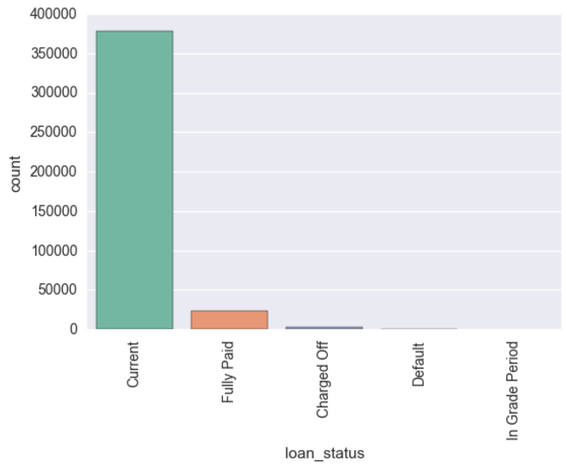
```
debt_consolidation    524215
credit_card           206182
home_improvement       51829
other                  42894
major_purchase         17277
small_business         10377
car                     8863
medical                 8540
moving                  5414
vacation                4736
house                   3707
wedding                 2347
renewable_energy         575
educational              423
```

The bar plot by loan purpose shows that few categories are more frequent than others however the proportion of bad loans seems consistent across all categories.

2014 Loans

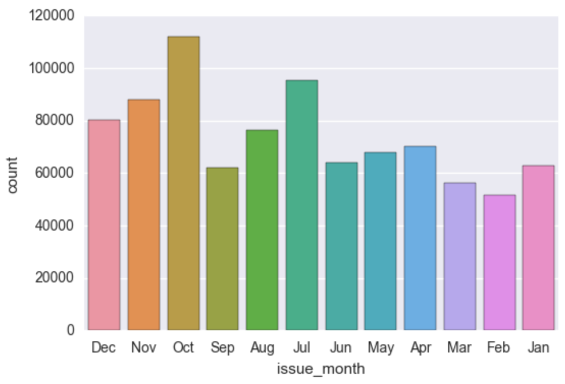


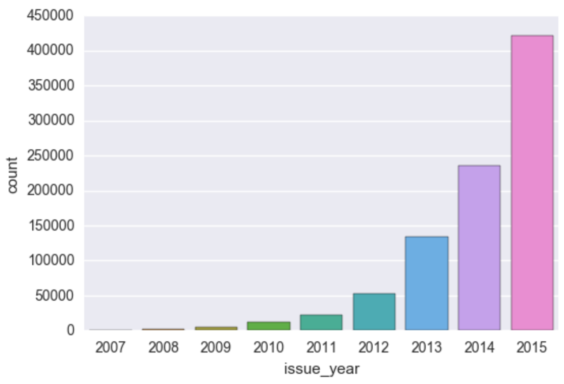
2015 Loans

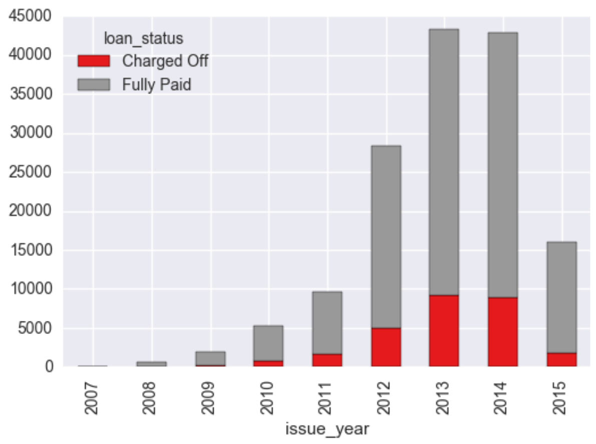


Issued by Month

Issued by Year





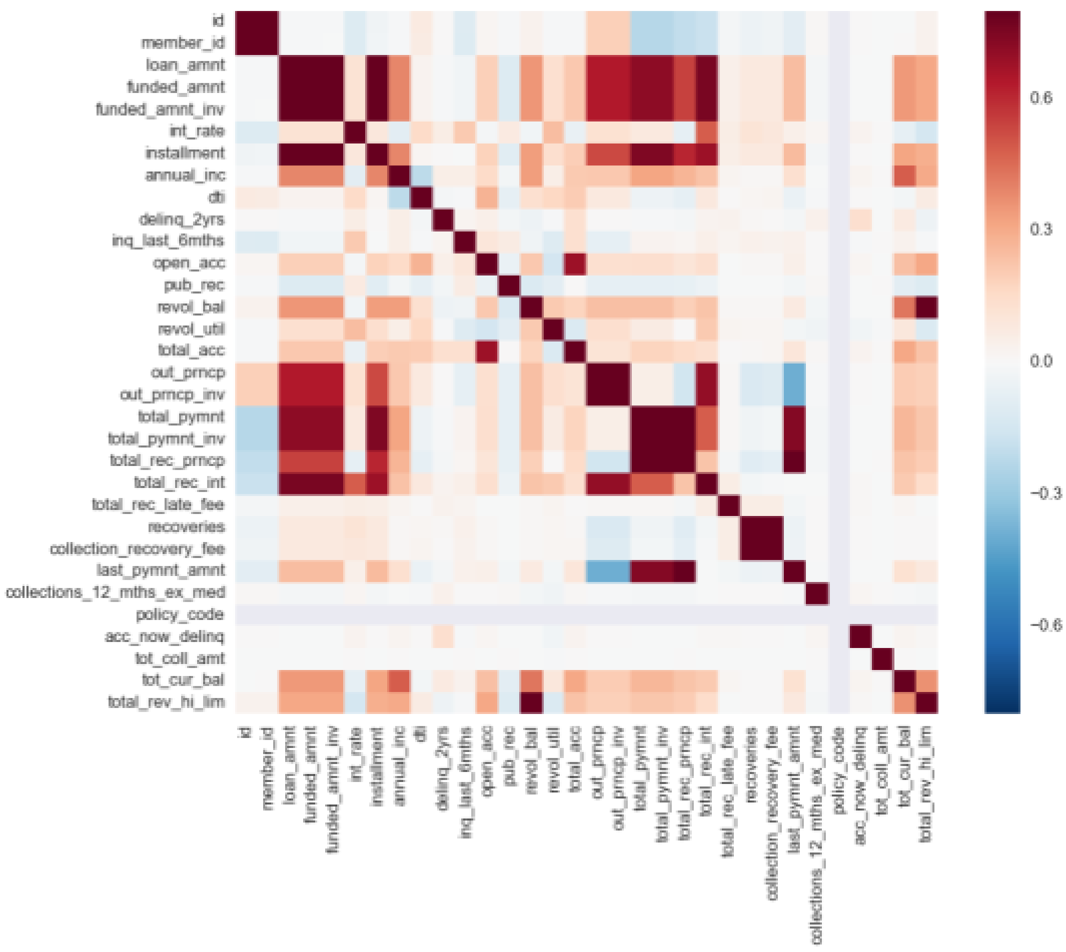


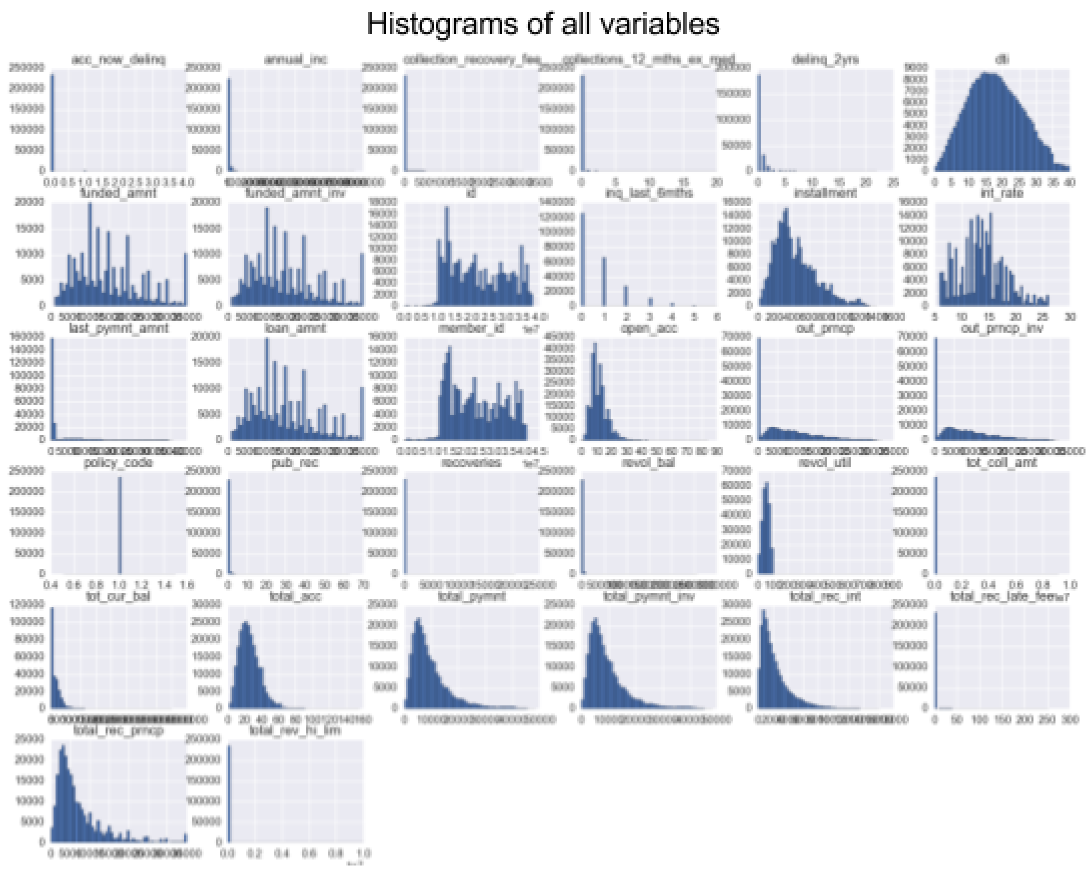
We can observe the important growth of accepted loans offered on the Lending Club marketplace while at the same time there is a significant reduction of loan default v fully paid ratio since 2013.

It’s also clear from the issue year plot that loans data from 2015 do not represent complete states for all loans. The proportion of current statuses vs completed (fully paid or charged off) loans is different from previous years. The assumption is that we’re actually looking at data with a higher number of in-flight transactions. These are not fit for machine learning because we need our algorithms to learn from a clear outcome. For this reason we’ll limit our scope to 2014 data, using a full calendar year in order to capture any seasonal variations and avoid overfitting by looking at one particular month or season.

Exploring correlations between attributes:

Correlation Matrix for all variables





**Algorithms and Techniques**

Our loan default prediction is a supervised learning classification problem dealing with an imbalanced dataset. Since it is one of the most common problem in machine learning, we have many algorithms at our disposal to handle this scenario. We have selected the following classifiers for our study; these provide different approaches to deal with our dataset, with varying level of performance and expressivity.

**Logistic Regression:** This is a linear model using a logistic function to classify data. Taking any real variables as input, it gives a probability estimate for the output. It is the appropriate regression algorithm to use when the output variable is binary. This model is relevant to our problem as it also supports penalised classification, it accepts a “class weight” parameter which penalises mistakes during classification.

**Decision Tree:** Classification and Regression Tree (CART) algorithm works top-down, by choosing a variable at each step that best splits the set of items. In the case of Decision Trees, which is the algorithm that fits our problem, each leaf node represents a target class (default or non-default). Information gain is used to decide which feature to split on at each step in building the tree, it’s a simple and efficient approach that also offer interpretability. Decision Trees however might not be as accurate as other algorithms and tend to overfit (not generalise well) so it’s important to ensure they don’t grow into an overly complex structure.

**Random Forest:** This is an ensemble learning method, able to solve classification problems. The classifier builds a set of decision trees at training time and outputs the mean prediction of the individual trees. This is also a very efficient algorithm on large datasets and gives an estimate of variables that are important for classification. Random Forest Classifiers also provide some protection against overfitting. Various methods (boosting, etc discuss) are available to RF behaviour.

Random Forests can overcome overfitting in Decision Trees and allow better handling on penalised classification for imbalanced data.

**SVM (Support Vector Machine):** SVM aims to achieve the best possible separation between classes of data. This algorithm is harder to overfit and able to handle large featureset, however it doesn't return confidence score (not explanatory). It can also be used with penalised model using class weight, giving more importance to certains classes. Achieving an optimised separation between fully paid and default loans is a desirable feature for our target model.

**Undersampling/Oversampling techniques**

In order to fine tune predictions on imbalanced data and improve performance of our classifier, sampling techniques have been evaluated. Both under or over sampling techniques are available and sometimes a combination of both in more sophisticated algorithms is used to optimise results on highly skewed datasets.

- Cluster Centroid Undersampling: Apply K-Means clustering algorithm to undersample the majority class by replacing data with cluster centroids. The number of clusters used is aligned with the number of samples in the minority class in order to balance the dataset and obtain a 50:50 class ratio.
- SMOTE oversampling: Implementation of the Synthetic Minority Over-Sampling algorithm which uses k nearest neighbours (KNN) technique to create synthetic data points. This technique ensures synthetic data is created on the same scale, using feature vectors and random scaling factor between 0 and 1.

**Benchmark**

Our "naive" benchmark is defined such that all loans are repaid. This achieves over 80% accuracy, however this benchmark also yields a 0% true positive rate and is of little help when predicting loan defaults. It basically says that none of the default loans can be detected. Therefore we've set an arbitrary objective to achieve predictions with up to 50% recall, allowing us to exclude one out of two default loans from investor's consideration.

# III. Methodology

**Data Preprocessing**

The lending club dataset needs a series of preprocessing steps to feed its data into our classifiers. This dataset is not initially intended for machine processing purpose and we had to make the following transformations

- **Missing values:** As a default we considered that any attribute with more than 50% missing values should be dropped; the task to impute missing values would be too challenging or misleading our classifier.
- **Surrogate identifiers:** id and member_id are meaningless system generated identifiers that would only mislead our classifiers, these were removed from the feature set
- **Post loan transaction features** (leaking data into the future): In this dataset there are a number of features with data captured during the lifecycle of a loan, including post loan transaction. For example: total or accumulated fields. We've dropped these feature because they won't be available when loans are submitted on the market place, at the time of making our predictions.
- **High correlation and predictive power**: Few attributes with "amount" information were highly correlated and were ignored because of their poor predictive power
- **Descriptive fields:** there are few descriptive, free text attributes available which could be exploited through text analytics or joining other datasets, however we decided to remove them for our analysis and focus a simple classification from numerical and categorical data. These are relevant to our analysis, we have simply decided to limit our scope and include them in future enhancements.
- **Imputations and derivations:** None had to be performed
- **Reformatting:** A number of numerical features are encoded with symbols, such as employment length, we simply removed these to keep numbers. This was possible since all numbers are aligned to the same scale.
- **One hot encoding** of categorical attributes was performed on 7 features
- **Our target variable**, the loan status, was mapped to a subset including Fully Paid and Charged Off set of binary statuses
- **Normalization** was applied to numerical attributes, **rescaling** was not required since we did not intend to use Gaussian based algorithms
- **Feature engineering:** Features issue month, issue year were generated. Description length was derived from the description field (before being dropped from the list of features)

## Implementation

We followed three major steps in our implementation; our intention was to measure and progress through refinement of our model, allowing us to learn before applying new techniques:

1. Model evaluation from pre-processed training and test sets: this is to give us a standard metrics of our classifier's performance before refinements to address class imbalance
2. Evaluation of classifiers with penalty
3. Evaluation of resampling techniques

In each step we followed the same methodology in terms of validation and metrics measurement. Unless explicitly specified, our input was unchanged in each step; we used a 75% Train and Test split which gave us 40,973 observations in the majority class of the training set and 10,510 observation of loan defaults. This represents the full dataset for 2014 completed loan transactions, for all types of loans: purposes and grades.

A Stratified K-fold validation was used to preserve percentage of samples for each class, this is important in our imbalance data class scenario.

Classification and Metrics collection for all algorithms in scope (Logistic Regression, Decision Tree and Random Forest Classifier): precision, recall, False Positive Rate, G-mean score. We ran our algorithms using the standard default parameters and collected metrics relevant to evaluate performance. Metrics were collected using a custom function rather than running a standard sklearn report from metrics. Our function  simply computed all ratios defined in the metric section, using false positives (fp), true positives (tp), false negatives (fn) and true negatives (tn) to produce metrics such as sensitivity, precision as well as accuracy.

Our second training and evaluation loop focused on a penalised model, all other parameter and input data being equal: the only change was adding a “class weight” as input during classification. The ratio picked was not random, we chose 1:8 to reflect the imbalance of our default/non-default distribution.

Lastly resampling techniques were used to evaluate improvements and compare performances over penalised models. The *imblearn* Python library was used to select one under sampling and one over-sampling technique, these are discussed in more details in the Refinement section of this report.

ROC and PR Curves analysis: Receiver Operating Characteristic Curves required fitting our models using *predict_probas* functions in order to get a probabilistic output and compute metrics for different thresholds. While standard predict functions for  binary classification produce a list of 0s and 1s, predicting probabilities yields a score between [0,1] as output which allows ROC curves to be plotted. Aside from the visual plot the important metric used to evaluate performance is called Area Under the Curve (AUC) which literally calculates the area available under the curve that fits the 1:1 square representing the TPR and FPR coordinates, 1 being the maximum possible performance.

In terms of challenges, It became clear that given the number of observations in our dataset, SVM algorithm was going to outscale the compute power available to us, especially for the K-fold validation. Therefore we decided to drop the evaluation of this classifier and move this into the scope of future improvements.

**Refinement**

Our methodology included many refinements, from simple parameter changes such as class weight (penalised model) to a grid search method to compute optimum parameters values yielding the best performance. More noticeably, the inclusion of data resampling techniques as a last step in our analysis requires more explanations.

- Penalty models: all classifiers chosen support a class weight parameter which “penalises” wrong classification of minority class, the input parameter is a ratio which tells the algorithm how much penalty is applied on each class.

- Under-sampling: Cluster Centroids, this undersampling method was used over a naive random undersampling or more sophisticated methods such as Near Miss

- Over-sampling: SMOTE was selected as it is often cited as an excellent over sampling method in terms of performance, we were curious to evaluate and compare this to other methods listed above

Lastly another refinement used was Precision/Recall Curve analysis which is a similar technique to ROC curves. The goal is to understand the trade-off between Precision and Recall by plotting their values against the threshold obtained from probabilistic predictions. It allows analysts to make decisions based on their goals, an optimal threshold can be picked when the goal is to minimise false positives over false negatives.

# IV. Results

**Model Evaluation and Validation**

Standard Predictions on our selected models are below. Penalised models showed significant improvements over standard parameters, these are the ones listed in this table. Remember that these were obtained using a K-fold stratified validation with 5 folds.

| **Model** | **FPR** | **Accuracy** | **Recall** | **Precision** | **G-means** | **AUC** |
|---|---|---|---|---|---|---|
| Linear Regression | 0.21 | 79.72 | 0.29 | 26.09 | 5.37 | 0.70 |
| Decision Tree | 21.66 | 66.73 | 20.77 | 19.51 | 40.33 | 0.68 |
| Random Forest Classifier | 7.81 | 75.09 | 7.43 | 19.39 | 26.18 | 0.81 |

Optimized results (using class weight penalty)

| Model | FPR | Accuracy | Recall | Precision | G-means | AUC |
|---|---|---|---|---|---|---|
| Linear Regression* | 74.91 | 35.18 | 75.11 | 20.22 | 43.41 | 0.71 |
| Decision Tree* | 22.01 | 66.55 | 21.29 | 19.64 | 40.74 | 0.68 |
| Random Forest Classifier* | 35.05 | 58.91 | 35.01 | 20.15 | 47.68 | 0.81 |

* results obtained with a class weight penalty (ratio 1:8)

We finally tested sampling methods, SMOTE oversampling performing marginally better than other methods

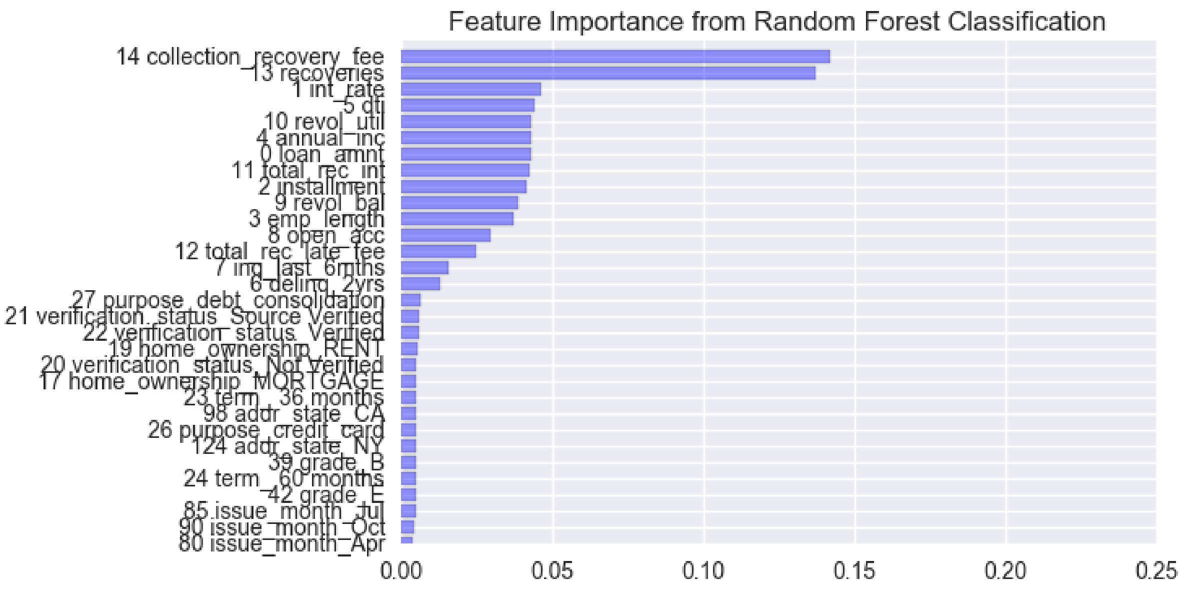
| Model | FPR | Accuracy | Recall | Precision | G-means | AUC |
|---|---|---|---|---|---|---|
| Linear Regression | 55.45 | 39.17 | 55.43 | 20.34 | 49.69 | 0.71 |
| Decision Tree | 43.14 | 45.38 | 43.66 | 20.54 | 49.82 | 0.83 |
| Random Forest Classifier | 31.64 | 51.11 | 32.24 | 20.65 | 46.94 | 0.97 |

We will discuss in the conclusion whether these results are align with our expectation, however for now, the results show similar performance overall between all classifiers selected. As expected random forest is showing improvements from a simple decision tree which gives us confidence about these results.

The sampling method is also boosting performance, in particular the recall and G-means which makes perfect sense. Recall in particular doubles to a very decent 44, this is however at the expense of accuracy.

In evaluating our model, feature importance can return valuable insight from our data and is can be used to make business decisions or tune the model further. How can we select good features to help our model perform? The plot below was created by extracting feature importance from a random Forest Classifier estimator

**Feature Importance**



**Justification**

Given the results outlined previously, a random Forest classifier looks like the best choice as it gives the best performance in the key metrics that matter to our problem: AUC. Logistic Regression however obtained better Recall and G-means but its AUC was inferior. This classifier is easy to train, offers good interpretability and can accept a class weight penalty to address class imbalance.

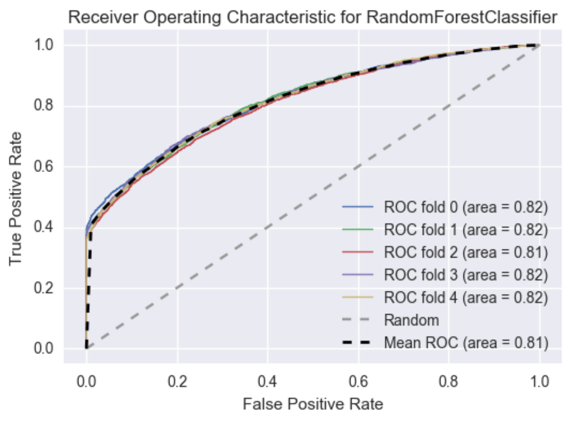
These results easily outperforms our benchmark which was achieving good accuracy but 0 recall (true positive rate). We have achieved our target, although more tuning is required in order to hit over 50% of true positive rate consistently with our choice of classifier and offer a solution that gives investors a good level of confidence in their investment. In order to achieve this however, we can start by selecting a higher threshold in the prediction probability and increase the true positive rate.

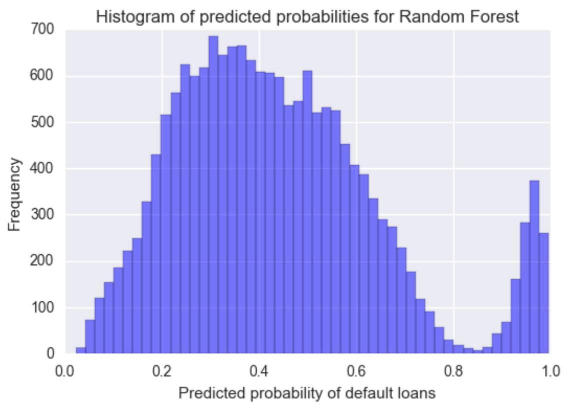
# V. Conclusion

**Free-Form Visualization**

Below a ROC Curves on a Random Forest Classifier with K-fold validation is displayed. This plot provides good insight into the performance of a Random Forest Classifier. Beyond the overall performance measured by AUC, it shows the tradeoff between False Positive and True Positive rate when moving the threshold of probability obtained from the classifier. Since our objective requires a greater focus on Loan default detection, we can decide to select a higher threshold and improve the true positive rate (TPR) at the expense of higher false positive, the curve makes this decision easier to assess.

We can combine ROC Curve analysis with an histogram of predicted probabilities to understand where threshold will have most variations. For example, we can see that there are a number of loans labelled with around 0.9 probability of default, moving the threshold from 0.5 to 0.6 is unlikely to have any significant changes, however pushing it further between 0.9 and 1.0 will ensure and a higher TPR and lower risk significantly.





**Reflection**

Two aspects of this project were challenging. The first one was dealing with a dataset of a fairly large size, covering many years of observations with many features in various formats and quality. An in-depth understanding of all features and data selection was key to produce a training set of good quality. For example, understanding features actually used in each part of the loan lifecycle, analysing distribution of loan statuses over time, correlation and poor quality input were time consuming activities required prior to the machine learning process.

The second one was to evaluate and gather metrics suitable to our problem, dealing with predictions on imbalance data class. In this scenario, refinement isn't just about finding a the extra percentage to boost final performance but an important step in the process to find meaningful and usable solution that truly lowers the exposure to risk for investors.

We managed to achieve a pretty good result with X and X … predictions of default loan and this result easily outperforms our benchmark and surpasses our expectation.

**Improvement**

There is a number of possible improvements to consider, some of them require more compute power, gathering more data, invest more time in study and investigations, or a combination of all.

More work could to be done on feature Engineering and enriching the dataset with localisation, demographic, macroeconomics data.

Understanding seasonal trends, for example holiday period, christmas could be added as input and help our classifier to understand patterns.
Adding Time series features. In a fast changing market such as P2P lending, we need to attach more importance to recent transactions in order to capture new trends in borrower's attitude and financial context.

Evaluating sampling methods requires more time and tuning that was allowed in this project, there are many strategies available that have not been evaluated.

Lastly, given the size of this growing dataset, we could run a model pipeline on Cloud computing with enough power available to evaluate and cross validate classifiers such as SVM, Neural Networks and further tune performance.

*References*

An introduction to ROC analysis - Tom Fawcett
http://people.inf.elte.hu/kiss/13dwhdm/roc.pdf
Learning from Imbalanced Data - Haibo He, Member, IEEE, and Edwardo A. Garcia
http://www.ele.uri.edu/faculty/he/PDFfiles/ImbalancedLearning.pdf