LOAN UNDERWRITING

***SUMMER SEMESTER*** ***PROJECT REPORT***

***FOR THE DEGREE OF***

**BACHELOR OF TECHNOLOGY**

***IN***

**INFORMATION TECHNOLOGY**



***by***

Mudit Rathore: ICM2015502

Ayushi Asthana: ITM2015004

Ishani Mishra: IWM2015008

|  |
| --- |
|  |
|

|  |
| --- |
|  |
|  |
|
|  |
|  |
|  |

***UNDER THE SUPERVISION OF***

Dr. Shirshu Varma

Assoc. Professor

**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY,** **ALLAHABAD**

**Candidate’s Declaration & Certificate**

We hereby declare that the work presented in this project report entitled “Loan Underwriting”, submittedSummer Semester report of B.Tech. (Information Technology) at Indian Institute of Information Technology, Allahabad, is an authenticated record of our original work carried out from May 2018 to Sept 2018 under the guidance of ​**Dr. Shirshu Varma​**. Due acknowledgements have been made in the text to all other material used. The project was done in full compliance with the requirements and constraints of the prescribed curriculum.

Date : 24th Sept 2018 Mudit Rathore (ICM2015502)

Place : Allahabad Ayushi Asthana (ITM2015004)

Ishani Mishra (IWM2015008)



**CERTIFICATE FROM SUPERVISOR**

I do hereby recommend that the summer semester project report prepared under my supervision by Mudit Rathore (ICM2015502), Ayushi Asthana (ITM2015004), Ishani Mishra (IWM2015008) titled “Loan Underwriting” be accepted in the partial fulfilment of the requirements of the completion of summer semester of Bachelor of Technology in Information Technology for Examination.

Date: 24th Sept 2018 Dr. Shirshu Varma

Place: Allahabad IIIT-Allahabad

ACKNOWLEDGEMENT

This project consumed huge amount of work, research and dedication. The implementation would not have been possible without the support of certain individuals. Therefore we would like to extend our sincere gratitude to all of them.

We would like to thank our supervisor Prof. Shirshu Varma for giving us the opportunity to carry out the project, for giving us the expertise and endless support in implementing this project.

We are also thankful to our panel Dr. Vrijendra Singh and Dr. Triloki Pant for their support and guidance.

We would also like to thank Mr. Sanchit Tyagi, PhD scholar, IIIT Allahabad for his support throughout the project work. His knowledge and experience and huge support helped the project in quality of outcomes.

Table of Contents

1. Candidate’s Declaration & Certificate…………………………...…..…1
2. Certificate from the Supervisor……………………………………….…..1
3. Acknowledgement……………………………………………………..….…..2
4. Table of Contents……………………………………………………...…….….3
5. Abstract…………………………………………………………………………….4
6. Introduction………………………………………………………….…………...5
7. Motivation…………………………………………………………….………...…7
8. Problem Definition………………………………………………..…….…….. 8
9. Literature Review…………………………………………………...……..……9
10. Methodology……………………………………………….………………. 12
    1. Libraries Used…………………………………………..………….…. 12
    2. Parameters Used………………………………………….…………. 13
    3. Data Mining Techniques…………………………………….…..... 15
       1. Strategies to deal with missing data……………….. 15
       2. Strategies to deal with imbalanced data………….. 16
    4. Modelling…………………………………………………………..…… 17
    5. Build Ensemble Model……………………………………..……… 20
11. Result…………………………………………………………………………. 22
12. Conclusions…………………………………………………..……………. 25
13. Discussions and Future Work…………………….…………………28
14. References……………………………………………………….………… 29

Abstract  
A loan is money, property or other material goods that are given to another party in exchange for future repayment of the loan value along with interest or other finance charges. The massive clean-up of bad loans in India is turning out to be a prolonged battle for survival for the country’s banks [1]. Therefore before lending a loan, the lender ensures if the borrower's loan application is an acceptable risk. Investors (lenders) provide loans to borrowers in exchange for the promise of repayment with interest within an agreed period of time, that means the lender only makes a profit (interest) if the borrower pays off the loan completely. However, if he/she doesn’t repay the loan, then the lender loses money. The lenders only approve a personal loan or retail finance applications where it can be demonstrated that loan repayments are affordable, based on the applicant’s income and outgoings, for the term of the loan, and that the applicant has a strong record of managing credit in the past. Background checks are done for this purpose. Underwriters assess the borrower's ability to repay the loan based on an analysis of their credit, capacity, and collateral. The underwriter can make a final decision without giving equal weight to each consideration. Weaknesses in one area can be offset by another.

Loan underwriting is the process that we undertake to analyze all of the information provided by each loan applicant and their credit file to assess whether or not that applicant meets our minimum loan criteria. As part of that process, all data is verified, analyzed and summarised to paint a picture of each applicant. Our system evaluates the current status of the loan and the financial position of the borrower and renders a programmatic loan decision regarding whether the loan will be repaid as expected during approval, the goal is to get this done without human intervention, the results, however, will be subject to verification by expert in case of doubt/discrepancy.

This system was developed to accurately predict the loan repayment for a particular borrower based on various parameters. The system receives selected information, debt-to-income ratio, public records, previous records etc for which the loan has been approved. It then predicts whether the loan will be charged off or fully paid.

Introduction

Loan Repayment is a major issue not only in Indian Financial Systems but across the globe. Due to bad loans, lenders need to face adverse economic conditions and restrictions. Depending upon the risk threshold and the Non Performing Asset(NPA), Financial Institutions with bad loans may be restricted [2] to carry out expansion in terms of branches, staff recruitment and the size of their loan book.

The process of loan starts with a borrower requesting an investor to lend. The borrower is asked to fill an application having as many as 500 data elements that helps the lender to correctly identify the borrower and accurately predict his/her economic condition. This information includes borrowers employment status, income, assets, liabilities, monthly expenses etc.. When the necessary information is collected, the stack of documents is an inch thick pile. At this point, the loan file is turned over to an underwriter, who evaluates the information, verifies the information against the supporting documents, and decides whether to approve the loan or deny the application.

A well-established bank approves 4-5 lakh loans every year [3], the number of application could be even higher, hence as far as the process of underwriting is concerned, the lending institution needs to be careful with the proceedings to minimize losses. The average turnaround time for a loan decision has come down from a few weeks with human underwriters to anywhere between 1 day to 10 seconds with automated systems!

The process of approval through a human underwriter is vulnerable to many shortcomings. The necessity of human intervention cannot be ignored in some subjective decisions, but it also opens the decision to bias, inaccuracy and delay, which we want to be reducing at all costs.

Through the years, several methods of automated underwriting have surfaced and have been commercially used. We discuss in this report one such method called Ensemble learning.

**Ensemble methods**[4] use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone.

They can be defined as combining several different models [5] (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 that 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.

Stacked Ensemble method[6] is supervised ensemble machine learning algorithm that finds the optimal combination of a collection of prediction algorithms using a process called stacking. It is able to support regression, binary classification and multi-class classification.

Stacking, also called Super Learning or Stacked Regression [6], is a class of algorithms that involves training a second-level “meta learner” to find the optimal combination of the base learners. Unlike bagging and boosting [7], the goal in stacking is to ensemble strong, diverse sets of learners together.

As base learners we are using - xgboost [8], random forest classifier [9] [10], and SVM models [11] to train our data. This would generate our level 1 data or the meta-features. These features would be used to train the meta-learner which is a simple logistic regression model [12].

The ensemble model is one of the optimum learning techniques, but we suffer another drawback with the data layout that is hard to overcome. The number of unpaid loans is as such much lower than the paid loans which makes the data highly skewed and introduces bias in the learning causing inaccurate results or false accuracy measures.

Motivation

Underwriting is difficult, expensive, and time consuming. It takes approximately 50 minutes for an underwriter to review a typical loan application. Non typical loans, such as those for FHA and VA mortgages, second homes, investment properties, and small apartment buildings, can take even longer. The typical underwriter is a highly skilled individual with many years of experience in mortgage banking. The underwriter has to be knowledgeable about hundreds of specific product and investor underwriting guidelines to determine if the loan profile satisfies the requirements of the lender and the ultimate investors. The sheer volume of information that has to be considered, along with the interdependence of this information, makes underwriting a challenging decision-making process.

The low interest rates of the last two years unleashed a substantial increase in the number of people applying for home loans [13], many to refinance under better terms. This increase has led to a serious shortage of qualified underwriters, creating major bottlenecks for all lenders. To cope with the increasing number of loans and the shortage of underwriter personnel, there arises a need of creating a system that would automate the underwriting process. Such a system would alleviate the critical shortage of underwriters; better use the current personnel; and, most importantly, allow lenders to increase the number of loans processed by each employee. Underwriters, freed from the need to process simple loans, could then concentrate on difficult loans, study the local markets more fully, and spend more time with customers.

Problem Definition

Lending to the corporate sector, particularly small and medium enterprises, is becoming increasingly difficult with more than half the country’s public sector banks (PSBs) now under the RBI’s Prompt Corrective Action (PCA) framework [14], which restricts lending activities of the banks. Eleven out of the 21 public sector banks in India were already under the banking regulation supervision, and had to contend with restricted business so far.

Depending on the risk thresholds set in PCA rules, the banks are restricted from expanding the number of branches, staff recruitment and increasing the size of their loan book. Other restrictions include higher provisions for bad loans and disbursal only to those companies whose borrowing is above investment grades.

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? 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.

We’ll be using publicly available data from LendingClub .The data covers 40,000 loans funded by the platform in the previous financial year. The interest rate is provided to us for each borrower. Therefore, we’ll address the second question indirectly by trying to predict if the borrower will repay the loan by its mature date or not. Thus ensuring if the borrower will be able to pay off the loan or not. This will benefit the small and medium enterprises or startups, which usually get ruled out, post manual underwriting, from getting their loan requests approved due to RBI’s PCA.

Literature Review

While working through the project, there were a variety of preprocessing steps needed to make the data ready for use. Some of the issues encountered were due to the data imbalance, and the amount of missing data in application forms. This needed to be addressed in order to improve performance of the model and ensure consistency.

There are three main problems that missing data causes: can introduce a substantial amount of bias, make the handling and analysis of the data more arduous, and create reductions in efficiency. Missing data can be tackled by deleting the tuple in the data set which has missing values. But deletion of data has many pitfalls: like if a test case has missing data then the program will give errors or the size of data set gets reduced and proper learning is hindered. Because missing data can create problems in analysis, imputation is seen as a way to avoid pitfalls involved with listwise deletion of cases that have missing values. Imputation is the process of replacing missing data with substituted values. Imputation preserves all cases by replacing missing data with an estimated value based on other available information. Once all missing values have been imputed, the data set can then be analysed using standard techniques for complete data.

Data imbalance[15] is one of the major problems prevailing in real time anomaly detection datasets. A dataset is considered to be imbalanced if one of its classes plays a huge dominance over the rest of the classes. This property can be observed with most prominence in binary classification datasets, as most binary classification datasets are implicitly imbalanced. The major class entries are higher, while the minor class entries occupy a very small space in the dataset. This unequal distribution can be of the form 100:1, 1000:1, 10000:1 etc.

Classification, being a supervised learning process depends mainly on the training data. The level of training plays a major role in the resultant accuracy of the classifier. Imbalanced nature of the data sets acts as a huge downside in this scenario. Due to the minimal occurrence of the minor classes, the classifier gets insufficiently trained and hence provide inaccurate predictions which won’t be reflected in normal accuracy scores or general confusion matrices. Over sampling, under sampling and hybrid sampling can be used to counter imbalance.

Sampling is a process used in which a predetermined number of observations are taken from a larger population. The methodology used to sample from a larger population depends on the type of analysis being performed but may include simple random sampling or systematic sampling.

Advantages include ease of use and accuracy of representation. No easier method exists to extract a research sample from a larger population. There is no need to divide the population into subpopulations or take any steps further than plucking the number of research subjects needed at random from the larger group. Again, the only requirements are that randomness governs the selection process and that each member of the larger population has an equal probability of selection.

Random forest classifier[9] [10] creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object. A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement if bootstrap=True (default).

Gradient boosting[8] is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

In machine learning, support vector machines are supervised learning models [11] with associated learning algorithms that analyze data used for classification and regression analysis. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

Robust Scaler[16] removes the median and scales the data according to the quantile range (defaults to IQR: Interquartile Range). The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile). Centering and scaling happen independently on each feature (or each sample, depending on the axis argument) by computing the relevant statistics on the samples in the training set. Median and interquartile range are then stored to be used on later data using the transform method.

Ensemble methods [5] are meta-algorithms that combine several machine learning techniques into one predictive model in order to decrease variance (bagging), bias (boosting), or improve predictions (stacking). 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 the 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.

Methodology

# Libraries Used

* **Pandas:** Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Using Pandas, we can accomplish five typical steps - load, prepare, manipulate, model, and analyze, regardless of the origin of data.
* **Seaborn:** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
* **Matplotlib.pyplot:** matplotlib.pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.
* **NumPy:** NumPy is the fundamental package for scientific computing with Python. It contains among other things a powerful N-dimensional array object, useful linear algebra, Fourier transform, and random number capabilities.
* **sklearn.preprocessing.LabelEncoder:** Encode labels with value between 0 and n\_classes-1.
* **Sklearn.model\_selection.train\_test\_split:** Split arrays or matrices into random train and test subsets.
* **Sklearn.model\_selection.cross\_val\_score:**Evaluate a score by cross-validation.
* **Sklearn.model\_selection.cross\_val\_predict:** Generate cross-validated estimates for each input data point.
* **sklearn.ensemble.RandomForestClassifier:**  A random forest classifier.
* **sklearn.ensemble.GradientBoostingClassifier:** Gradient Boosting for classification.
* **Sklearn.pipeline.make\_pipeline:** Construct a Pipeline from the given estimators.
* **sklearn.preprocessing.RobustScaler:** Scale features using statistics that are robust to outliers.
* **sklearn.preprocessing.Imputer:** Imputation transformer for completing missing values.
* **Imblearn.pipeline.make\_pipeline:**  imbalanced-learn has a Pipeline which extends the scikit-learn Pipeline, to adapt for the fit\_sample() and sample() methods in addition to fit\_predict(), fit\_transform() and predict() methods of scikit-learn.
* **imblearn.under\_sampling.RandomUnderSampler:** Gives us random undersampling of the training data.
* **imblearn.over\_sampling.RandomOverSampler:** Gives us random undersampling of the training data.
* **imblearn.over\_sampling.SMOTE:** SMOTE with Imbalance Data using imblearn module.
* **imblearn.ensemble.BalancedBaggingClassifier:** Balanced Bagging for classification.
* **Xgboost:**XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable.
* **sklearn.svm.SVC:**C-Support Vector Classification (C: Float, Optimal).
* **sklearn.linear\_model.LogisticRegression:** Logistic Regression classifier.
* **sklearn.ensemble.VotingClassifier:** Soft Voting/Majority Rule classifier for unfitted estimators.

# Parameters used

* **Purpose:** the Category in which the loan is applied, this parameter is considered in order to determine the risk involved in investing the money by the investor. High Risk and Low risk lending have different criteria’s for the evaluation of a particular application.
* **Revol\_balance:**  This indicates the borrower’s revolving money
* **inq\_last\_6mths**: The borrower’s number of inquiries by creditors in the last 6 months. This parameter indicates the number of inquiries by the investor to the borrower because of violating one of the conditions of lending.
* **deling\_2yrs**: the number of times the borrower had been past due on a payment in the past 2 years.this parameter indicate the delay in the installments of the loan in previous two years we have considered two years because this precisely give us the best idea about the present economic condition of the borrower.
* **pub\_rec:**  The number of derogatory public records. This parameter refers to any criminal records/derogating records/trials/etc. on the borrower's end
* **annual\_inc:** The annual income of the borrower.This parameter is used to correctly access and precisely predict the present economic conditions of the borrower.
* **Int\_rate:** The interest rate of the loan (proportion).This indicates the rate of interest on which the investor have lend the money to borrower.
* **Installment:** This the monthly installments owed by the borrower that is repaid over time with a set number of scheduled payments; normally at least two payments are made towards the loan. The term of loan may be as little as a few months and as long as 30 years.
* **dti:** The debt-to-income ratio of the borrower. The debt-to-income ratio is one way lenders, including mortgage lenders, measure an individual's ability to manage monthly payment and repay debts. DTI is calculated by dividing total recurring monthly debt by gross monthly income, and it is expressed as a percentage.
* **revol\_util:** The borrower’s revolving line utilization rate. Revolving utilization, also known as your “debt-to-limit ratio” or “credit utilization,” measures the amount of your revolving credit limits that you are currently using.
* **Loan\_status:** The status of the Loan.
* **Emp\_status:** This parameter indicates the employment status of the borrower.
* **Real\_estate\_ownership:** This parameter indicates that when in the extreme opposite economic conditions of the borrower will the borrower be able to lead a economically stable status.

# Data Mining Techniques:

* **Data Selection:** We may not use all the data we have collected in the first step. In this step, we sample the dataset for training our model due to computational constraints that were encountered.
* **Data Cleaning:** The data we have collected is not clean and may contain errors, missing values, noisy or inconsistent data. So we need to apply different techniques to get rid of such anomalies. In our model, we cleaned the data on the basis of missing values, and inconsistent data entries. Since we would use different samples for different learners, this step is applied to the entire dataset instead of the sample only.
* **Data Transformation:** The data even after cleaning are not ready for mining as we need to transform them into forms appropriate for mining. The techniques used to accomplish this are smoothing, aggregation, normalization etc.
* **Data Mining:** Now we are ready to apply data mining techniques on the data to discover the interesting patterns. Techniques like clustering and association analysis are among the many different techniques used for data mining.

### Strategies to deal with missing values :-

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 how significant the values entered individually are. In our case, six features have missing values so we would add six binary features one for each feature. This, in combination with the feature importance check tells us if the missing values have any significance in the decision making or they are randomly generated due to some faults. Good thing is that the missing values are in the predictors only and not the labels. 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. The disadvantage of this model is that the contribution of other features to the model is also lost.
* Impute the missing values using the mean of each feature separately.
* Impute the missing values using the median of each feature separately.

### Strategies to deal with imbalanced datasets :-

Classification problems in most real world applications have imbalanced data sets. In other words, the positive examples (minority class) are a lot less than negative examples (majority class). In our case, the positive examples ( **Charged off** ) were only 17.81% and the negative examples (**Fully Paid**) were 82.19% of the total examples. Therefore, accuracy is no longer a good measure of performance for different models because if we simply predict all examples to belong to the negative class, we achieve 81% accuracy. Better metrics for imbalanced data sets are AUC (area under the ROC curve). However, that’s not enough because class imbalance influences a learning algorithm during training by making the decision rule biased towards the majority class by implicitly optimizing predictions based on the majority class in the dataset. As a result, we’ll explore different methods to overcome class imbalance problem.

* **Under-Sample:** Under-sample the majority class with or without replacement by making the number of positive and negative examples equal. One of the drawbacks of under-sampling is that it ignores a good portion of training data that has valuable information. In our example, it would loose around 32,876 examples. However, it’s very fast to train.
* **Over-Sample:** Over-sample the minority class with or without replacement by making the number of positive and negative examples equal. We’ll add around 32,876 samples from the training data set with this strategy. It’s a lot more computationally expensive than under-sampling. Also, it’s more prone to overfitting due to repeated examples.
* **EasyEnsemble:** Sample several subsets from the majority class, build a classifier on top of each sampled data, and combine the output of all classifiers.
* **Synthetic Minority Oversampling Technique (SMOTE):** It over-samples the minority class but using synthesized examples. It operates on feature space not the data space. Here how it works:
  + Compute the k-nearest neighbors for all minority samples.
  + Randomly choose number between 1-k.
    - For each feature:  
      a. Compute the difference between minority sample and its randomly chosen neighbor (from previous step).  
      b. Multiply the difference by random number between 0 and 1.  
      c. Add the obtained feature to the synthesized sample attributes.
  + Repeat the above until we get the number of synthesized samples needed.

# Modelling

**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 the 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. 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.

Diﬀerent ensemble methods construct the ensemble of models in diﬀerent ways. Below are the most common methods:

* **Blending:** Averaging the predictions of all models.
* **Bagging:** Build different models on different datasets and then take the majority vote from all the models. Given the original dataset, we sample with replacement to get the same size of the original dataset. Therefore, each dataset will include, on average, 2/3 of the original data and the rest 1/3 will be duplicates. Since each model will be built on a different dataset, it can be seen as a different model. *Random Forest* improves default bagging trees by reducing the likelihood of strong features to picked on every split. In other word, it reduces the number of features available at each split from *n* features to, for example, *n/2* or *log(n)*features. This will reduce correlation and hence, reduce variance.
* **Boosting:** Build models sequentially. That means each model learns from the residuals of the previous model. The output will be all output of each single model weighted by the learning rate λ. It reduces the bias resulted from bagging by learning sequentially from residuals of previous trees (models).
* **Stacking:** Build k models called base learners. Then fit a model to the output of the base learners to predict the final output.

Since we’ll be using Random Forest classifiers (bagging technique) and Gradient Boosting (boosting technique) classifiers as base learners in the ensemble model, we’ll illustrate only averaging and stacking ensemble methods. Therefore, modeling would consist of three parts:

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

Before going further, the following data preprocessing steps will be applicable to all models:

1. Create dummy variables from the feature “purpose” since its nominal (not ordinal) categorical variable. It’s also a good practice to drop the first one to avoid linear dependency between the resulted features since some algorithms may struggle with this issue.
2. Split the data into training set (70%), and test set (30%). Training set will be used to fit the model, and test set will be to evaluate the best model to get an estimation of generalization error. Instead of having validation set to tune hyperparameters and evaluate different models, we’ll use 10-folds cross validation because it’s more reliable estimate of generalization error.
3. Standardize the data. We’ll be using RobustScaler [16] so that the standardization will be less influenced by the outliers, i.e. more robust. It centers the data around the median and scale it using *interquartile range (IQR)*. This step will be included in the pipelines for each model as a transformer so we will not do it separately.

# Build Ensemble Models:

We’ll build ensemble models using three different models as base learners:

* Gradient Boosting
* Support Vector Classifier
* Random Forest

The ensemble models will be built using two different methods:

* 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 [17] from sklearn that:

1. Fits all the base learners on the training data
2. At test time, use all base learners to predict test data and then take the average of all predictions.

* 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). There are few different ways of training stacked ensemble model:

1. Fitting the base learners to all training data and then generate predictions using the same training data it was used to fit those learners. This method is more prone to overfitting because the meta learner will give more weights to the base learner who memorized the training data better, i.e. meta-learner won’t generate well and would overfit.
2. Split the training data into 2 to 3 different parts that will be used for training, validation, and generate predictions. It’s a suboptimal method because held out sets usually have higher variance and different splits give different results as well as learning algorithms would have fewer data to train.
3. Use k-folds cross validation where we split the data into k-folds. We fit the base learners to the (k -1) folds and use the fitted models to generate predictions of the held out fold. We repeat the process until we generate the predictions for all the k-folds. When done, refit the base learners to the full training data. This method is more reliable and will give models that memorize the data less weight. Therefore, it generalizes better on future data.

We’ll use logistic regression as the meta-learner for the stacked model. Note that we can use k-folds cross validation to validate and tune the hyperparameters of the meta learner. We will not tune the hyperparameters of any of the base learners or the meta-learner; however, we will use some of the values recommended by the [Pennsylvania Benchmarking Paper](https://arxiv.org/pdf/1708.05070.pdf). Additionally, we won’t use EasyEnsemble in training because, after some experimentation, it didn’t improve the AUC of the ensemble model more than 2% on average and it was computationally very expensive. In practice, we sometimes are willing to give up small improvements if the model would become a lot more complex computationally. Therefore, we will use RandomUnderSampler. Also, we’ll impute the missing values and standardize the data beforehand so that it would shorten the code of the ensemble models and allows us to avoid using Pipeline. Additionally, we will plot ROC and PR curves using test data and evaluate the performance of all models.

Result











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. Below are some takeaway points:

* There is no definitive guide of which algorithms to use given any situation. What may work on some data sets may not necessarily work on others. For example, for a specific set of loan types, relatively simpler models also gave significantly accurate results.
* Sometimes we may be willing to give up some improvements to the model if that would increase the complexity much more than the percentage change in the improvement to the evaluation metrics.  
   Using a relatively complicated sampling technique like SMOTE or EasyEnsemble significantly increased the complexity of the program while causing as little as a 2% increase in accuracy.
* 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. In the context of lending, losing money by lending to a risky borrower who is more likely to not fully pay the loan back is a lot more costly than missing the opportunity of lending to trust-worthy borrower (less risky). As a result, we can use class\_weight that changes the weight of misclassifying positive example in the loss function. Also, we can use different cut-offs assign examples to classes. By default, 0.5 is the cut-off; however, we see more often in applications such as lending that the cut-off is less than 0.5. Note that changing the cut-off from the default 0.5 reduce the overall accuracy but may improve the accuracy of predicting positive/negative examples.
* When building ensemble models, try to use good models that are as different as possible to reduce correlation between the base learners.
* EasyEnsemble usually performs better than any other resampling methods.
* Missing values sometimes add more information to the model than we might expect. One way of capturing it is to add binary features for each feature that has missing values to check if each example is missing or not.

Future Work

With the growing need of automation and speed of task processing, automated underwriting and repayment predictions can help save resources and time. This also means institutions can direct their efforts towards efficient measures to minimize losses. The current automated underwriting technique has a lot of scope of improvement. This technique can be modified with the help of AI, to develop recommender systems that will be able to not only predict the decision, but also highlight areas of ambiguity for human review. Also, the model can be constructed with various sets of more advanced base learners to improve performance, for eg. we could’ve enhanced our stacked ensemble model by adding *Dense Neural Network* and some other kind of base learners as well as adding more layers to the stacked model.

The current models all suffer from bias due to data imbalance, while there are various techniques to handle this, there is no hard and fast rule. More research in the field would deal with data handling and optimized usage to maximize output accuracy. The minimization of Type - 1 error in predictions is necessary due to the overly costly false positives in this particular application.

The system of automated underwriting is largely a server based application, but with improving accuracies and incorporating AI, there is a scope of extending this as a client based application in the market to facilitate easy review and feedback on a loan application directly to the customer.

The current system is therefore, accurate, and robust but has scope of further improvements in terms of implementation and application

References

[1]. Nupur Anand. “*India’s bad-loan crisis is so dire that one bank is now banned from lending money*”, [https://qz.com/india/1276803/ dena-bank-isnt-alone-half-of-indias- government-owned-banks-are-in-trouble/](https://qz.com/india/1276803/dena-bank-isnt-alone-half-of-indias-government-owned-banks-are-in-trouble/), May 14, 2018.

[2]. Sunny Verma, Sandeep Singh. “*Checking NPAs: 11 public sector banks now on RBI watchlist, small firms face credit crunch*” <https://indianexpress.com/article/business/banking-and-finance/checking-npas-11-public-sector-banks-now-on-rbi-watchlist-small-firms-face-credit-crunch-5129389/>, April 9, 2018.

[3]. State Bank of India. "*Annual Report [2017-18]*". Internet: <https://www.sbi.co.in/AR1718/>, March 2018 [June 29, 2018].

[4]. Opitz, David, and Richard Maclin. *"Popular ensemble methods: An empirical study."* Journal of artificial intelligence research 11 (1999): 169-198.

[5]. Zhou, Zhi-Hua. *“Ensemble methods: foundations and algorithms.”* Chapman and Hall/CRC, 2012.

[6]. H2O.ai. "*Stacked Ensembles*" Internet: [http://docs.h2o.ai/h2o/latest-stable/ h2o-docs /data-science/stacked-ensembles.html](http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/stacked-ensembles.html). Sep 21, 2018.

[7]. Kumari, GT Prasanna. "*A Study Of Bagging And Boosting Approaches To Develop Meta-Classifier.*" IRACST – Engineering Science and Technology: An International Journal (ESTIJ), ISSN: 2250-3498,

Vol.2, No. 5, October 2012

[8]. Chen, Tianqi, Tong He, and Michael Benesty. "*Xgboost: extreme gradient boosting*." R package version 0.4-2 (2015): 1-4.

[9]. Liaw, Andy, and Matthew Wiener. "*Classification and regression by randomForest.*" R news 2, no. 3 (2002): 18-22.

[10]. Kartasheva, Anastasia V., and Mikhail Traskin. "*Insurers’ insolvency prediction using random forest classification.*" The Wharton School, University of Pennsylvania, Philadelphia, USA (2011).

[11]. Chaudhuri, Arindam. "*Modified fuzzy support vector machine for credit approval classification.*" AI Communications 27, no. 2 (2014): 189-211.

[12]. Hosmer, David W., Trina Hosmer, Saskia Le Cessie, and Stanley Lemeshow. "*A comparison of goodness‐of‐fit tests for the logistic regression model.*" Statistics in medicine 16, no. 9 (1997): 965-980.

[13]. H. Talebzadeh, S. Mandutianu, and C. F. Winner "*Countrywide Loan Underwriting*

*Expert System*" AI Magazine Volume 16 Number 1 (1995)

[14]. Press Information Bureau Government of India Ministry of Finance. “*Prompt Corrective Action (PCA) of RBI on Public Sector Banks (PSBs).*”, March 16, 2018.

[15]. Somasundaram, Akila & Reddy, U. Srinivasulu. “*Data Imbalance: Effects and Solutions for Classification of Large and Highly Imbalanced Data*.” International Conference on Research in Engineering, Computers and Technology (ICRECT 2016)

[16]. scikit-learn developers "*sklearn.preprocessing.RobustScaler*". Internet: <http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html>[June 30, 2018]

[17]. scikit-learn developers "*sklearn.ensemble.VotingClassifier*". Internet: <http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html>[July 5, 2018]