CS 573 Final Project Report

Home Credit Default Prediction using Data Mining Techniques

Varad Satam
MS Industrial Engineering
Purdue University
West Lafayette, IN
vsatam@purdue.edu

Anas Patankar
MS Industrial Engineering
Purdue University
West Lafayette, IN
patankaa@purdue.edu

Kaushik Manchella
MS Industrial Engineering
Purdue University
West Lafayette, IN
kmanchel@purdue.edu

ABSTRACT

In utilizing loan applicant data provided by the Home Credit consumer finance group on Kaggle, the goal of this project is to make predictions on whether or not a given individual will default on loan payments.

The two stakeholders for which this problem was being tackled are 1) Lending Businesses and 2) Loan Applicants. In making binary class predictions, the project assumed the primary objective of minimizing financial risk of lending money to defaulters while maximizing the number of worthy applicants who are awarded a loan.

Over the course of this exercise, a unique problem of target imbalance was encountered. In dealing with this, it was found that the standard metrics and assumptions about the binary target variable that were used throughout the CS 573 assignments were ineffective. This played a crucial role in driving the approach to model building and evaluation. Consequently, a variety of different predictive modeling algorithms were implemented and tuned to maximize the metrics that support the core business objective of minimizing risk while maximizing profit.

This paper intends to describe the activities that were taken up by the team to address the stated objectives.

1. Introduction

1.1 Background

The two stakeholders of this project are lending businesses (such as banks) and the consumers who are applying for the loans (who we shall refer to as applicants through the remainder of this paper).

From the perspective of the applicants, obtaining a loan provides them with an opportunity to own a home. Owning a home has a significant impact on the applicant's quality of life. For this reason, it is important maximize the loans given to applicants who are capable of repaying their respective loans. Unfortunately, this is not always the case in today's procedures. For instance, applicants who are fully capable of repaying a loan, but lack credit score are usually not qualified for loans currently.

From the perspective of the lending businesses, it is important to maximize the profit while minimizing risk. On a high level, profit is maximized by giving out as many loans as possible. With a larger number of loans, the business can expect a larger sum of interest payments that contribute to profits. It is as important (if not more important) to minimize loss due to loan defaulters not paying back. For this reason, it is important for the business to predict accurately which applicants are unlikely to repay their loan.

1.2 Scope

This project intends to utilize the available data sources, implement data mining algorithms, and make predictions with the intent of minimizing risk on the businesses while maximizing the number of loans being given out to the right people. The target variable for this prediction is a binary class which describes whether or not a given applicant will default on a loan payment.

In addition to making the predictions based on these objectives, the team intends to gain insight on topics such as but not limited to: the most influential factors in identifying loan defaulters, the performance of different models in the specific scenario of imbalance sets, best methods to train the models to make imbalance methods, unique strengths and weaknesses of different data mining algorithms.

2. Data Sets

2.1 Overview

The data has been provided by a company by the name of "Home Credit". There are 7 different data sets available from different sources for use in this project. While the initial scope was to study and leverage data from all the different data sets to make prediction, the team had to constrain the number of data sets used to three due to inadequate resources for storage and processing.

Below is a visual that describes all the available data sets and how they are related to each other:

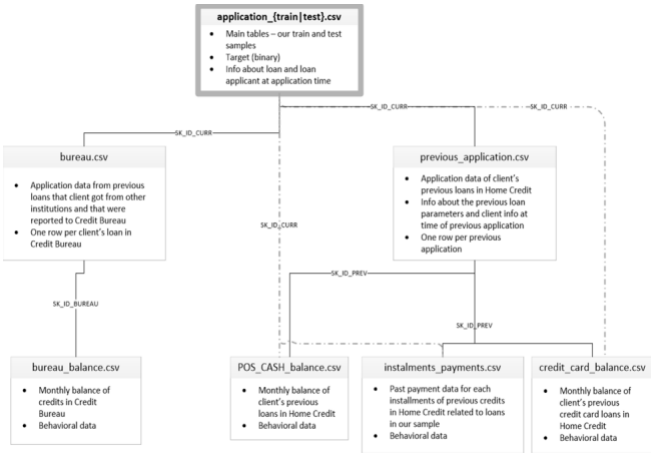

Figure 1: Table Schema

A larger version of this schema can be found in APPENDIX II.

2.2 Table Descriptions

Table: application.csv; Size: 308k Records, 122 Variables

This is the main table, which contains static data for all applications. One row represents one loan in the data sample.

Table: bureau.csv; Size: 1.72m Records, 17 Variables

All client's previous credits provided by other financial institutions that were reported to Credit Bureau (for applicants who have a loan in the sample).

For every loan in the sample, there are as many rows as number of credits the applicant had in Credit Bureau before the application date.

Table: bureau_balance.csv; Size: 27.3m Records, 3 Variables

Monthly balances of previous credits in Credit Bureau.

Table: POS_CASH_balance.csv; Size: 10m Records, 8 Variables

Monthly balance snapshots of previous POS (point of sales) and cash loans that the applicant had with Home Credit.

Table: credit_card_balance.csv; Size: 3.84m Records, 23 Variables

Monthly balance snapshots of previous credit cards that the applicant has with Home Credit.

Table: previous_application.csv; Size: 1.67m Records, 37 Variables

All previous applications for Home Credit loans of clients who have loans in the sample.

There is one row for each previous application related to loans in the data sample.

Table: installments_payments.csv; Size: 13.6m Records, 8 Variables

Repayment history for the previously disbursed credits in Home Credit related to the loans in the sample.

2.3 Data Set Selection

As can be seen in the table descriptions, there are millions of records in combination with hundreds of variables. This poses a challenge with the dimensionality of the data. As the team was limited by processing and storage resources, a selection was made between all the available tables.

The primary table with the target variable (application.csv) had to be selected as the base set. In addition to this, bureau.csv and bureau_balance.csv were selected. The reasoning behind the selection was that credit bureau data is generally the key data source used for qualifying loan applicants in current practice. While variables from other tables such as previous_application.csv and installments_payments.csv may have an influence on the target variable, they were omitted on the assumption that they contain similar and correlated features as the bureau tables.

In summary, the following tables were used in training and testing models: **application.csv, bureau.csv, bureau_balance.csv**

3. Exploration

3.1 Target Variable

The first step in the exploratory data analysis of this project was to observe the target variable. As mentioned before, this is a binary target variable which represents whether or not a given applicant will repay the loan. Below is a histogram showing the distribution of this target variable:

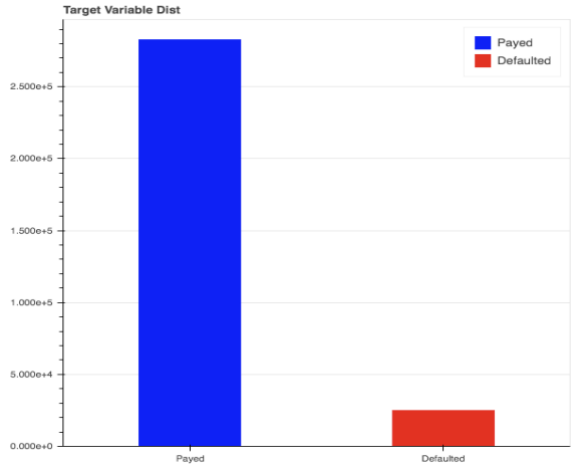

Figure 2: Target Variable Distribution

From the visualization above, it can be seen clearly that the target variable exhibits an imbalance distribution. Approximately 93% of the total applicants successfully payed on their loans, while only 7% of them defaulted. This observation of imbalance is very essential in determining the success criteria for models performance.

3.2 Missing Data

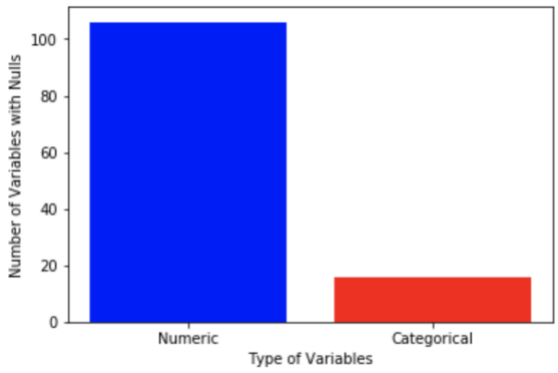

Figure 3: Counts of Variables with Missing Data

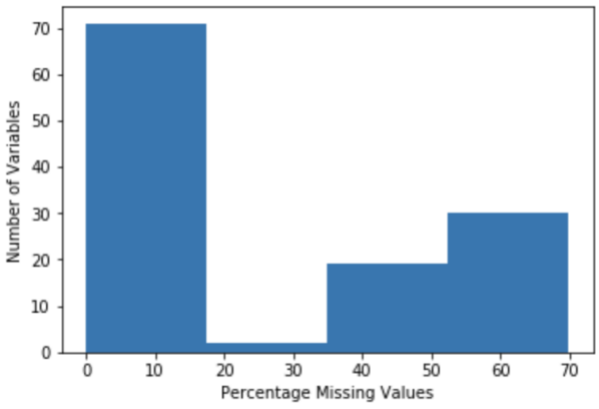

Figure 4: Histogram of Percentage Missing Values in Variables

For both numeric and categoric variables which contained less than 10% missing values, the rows were removed.

Numeric variables which were in the 20-50% range were median imputed, while Categoric variables in that range were mode imputed.

Cases for which numeric variables with more than 50% missing values were generally legitimately inapplicable. An example is with the variable in the application set is "Own Car Age" which was mostly null as a result of applicants not having a car. In such situations, numeric variables were imputed with a 0. Categorical variables in this band were one hot encoded, as a result, imputation was not required for inapplicable cases.

3.3 Highlights

This subsection shows some of the interesting observations as the variables were being explored.

3.3.1 Influence of Age

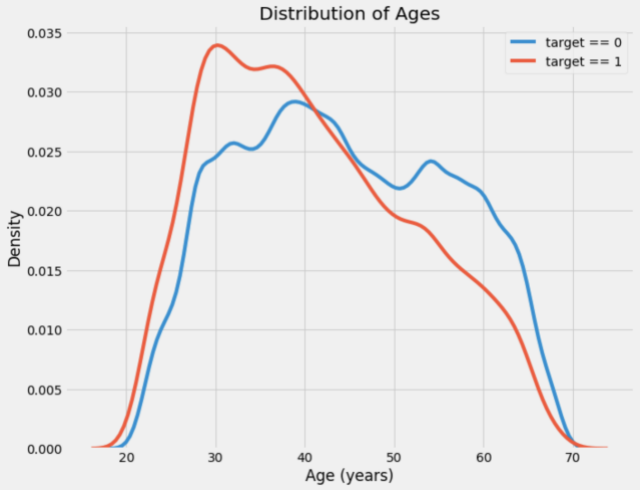

Figure 5: Density Plot - Age

The density plot in Figure 5 as well as the default rate bar plot in Figure 6 both suggest that younger applicants have a higher tendency to default on their loan payments.

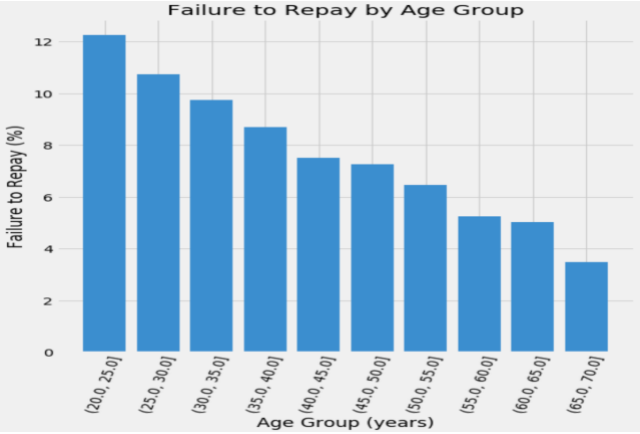

Figure 6: Default Rate - Age Group

3.3.2 Influence of Education

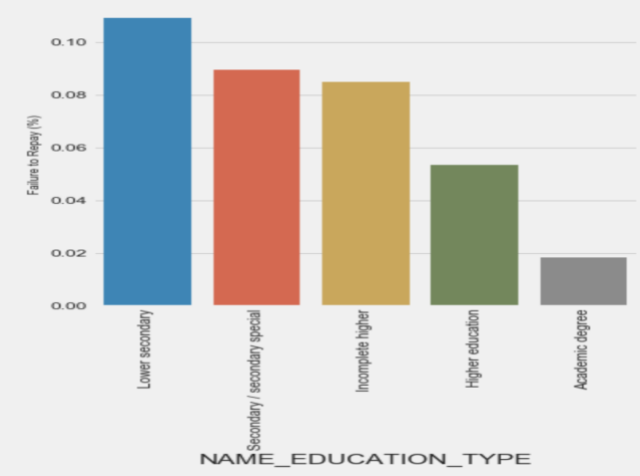

Figure 7: Default Rate - Education Type

Figure 7 corroborates with the intuitive hypothesis that the less educated applicants are more likely to default on payments.

3.3.3 Influence of Occupation

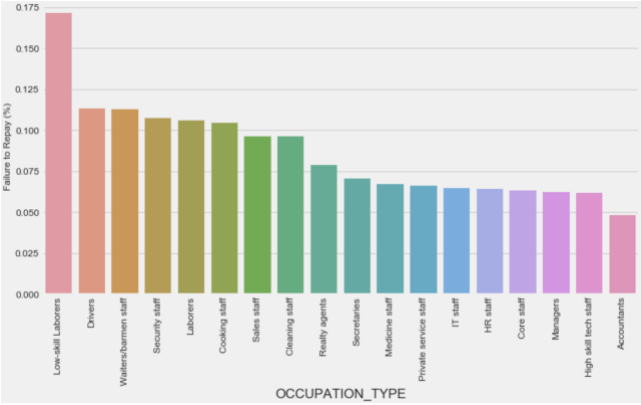

Figure 8: Default Rate - Occupation Type

Figure 8 suggests that Low-skill Laborers are most likely to default while Managers, Accounts, etc. are least likely.

3.3.4 Influence of Family size

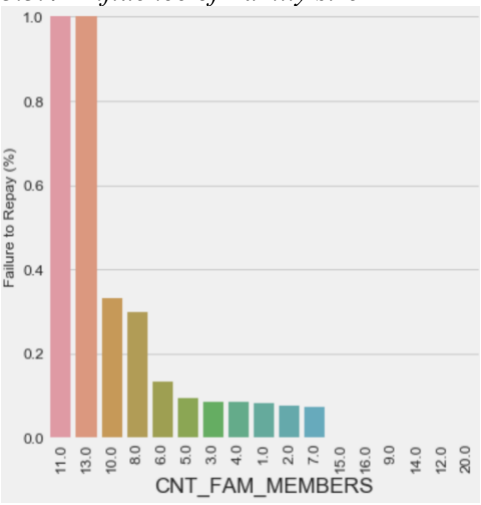

Figure 9: Default Rate - Family Size

Figure 9 suggests that applicants in larger households show a higher default rate (potentially due to higher dependency and uncertainty of the financial needs of the family)

3.3.5 Loan Amounts

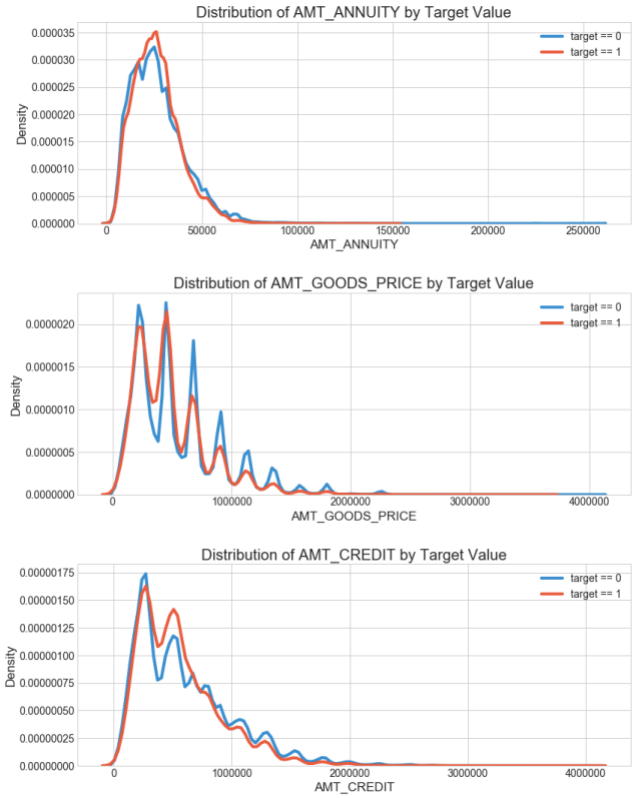

Figure 10: Density Plots - Loan Amounts

Figure 10 shows that there is no particular relationship between loan amount and default rates. ie. The payment amounts aren't the best indicators of the default risk.

3.3.6 Study of Correlated Variables

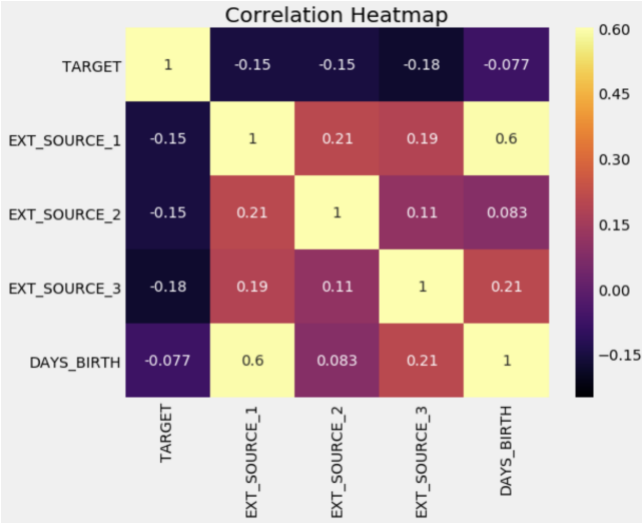

Figure 11: Correlation Heat Map

During exploration, some variables were seen to highly correlate with each other. The particular group that was studied was 3 external risk scores and age (DAYS_BIRTH). The relationship allowed to generate new polynomial features described in Section 4.1.

4. Feature Engineering

4.1 Set 1 (Null Set)

The first feature set consists of only the application data set which comprises of the static data of all applicants. Each applicant has a unique ID which acts as an identifier and the pathway to connect other datasets. For each applicant, their description regarding their demography, financial status and variables regarding their residential location are given. The detailed description is given in the Application Dataset column description. The modeling was initially done on this dataset.

4.2 Set 2

The feature engineering done for this set is the addition of Polynomial features. In this method, the strategy was to create new features that are powers of existing features as well as interaction terms between existing features. These features that are a combination of multiple individual variables are referred to as interaction terms because they capture the interactions between variables. In other words, while two variables by themselves may not have a strong influence on the target, combining them together into a single interaction variable might show a relationship with the target. Interaction terms were used to capture the effects of multiple variables. A select few interaction terms were aggregated to see if they might help our model to predict whether a client will repay a loan.

Polynomial features were created using the EXT_SOURCEs, DAYS_BIRTH variables. These features were specifically chosen as they have the highest correlation with the target variable (refer to Figure 11).

4.3 Set 3

In the previous two feature sets, only the application dataset was used for modeling. In order to better are recall and precision score, two more tables (burau.csv and bureau_balance.csv) were merged into Feature Set 2 to create this final feature set. The two other datasets include bureau dataset, which provides information about client's previous loans with other financial institutions reported to Home Credit where each previous loan has its own row and the bureau balance dataset which provides monthly information about the previous loans.

Since the team had limited domain knowledge about what makes a person likely to default, the focus was placed on getting as much information as possible into our final training dataset. Highly correlated variables were then removed in order to prevent the problem of multicollinearity.

The two bureau datasets were aggregated based on the basis of current customer ID and the aggregated datasets were joined with the application dataset in order to get our final training dataset.

5. Modeling

The imbalance problem was dealt with by sampling and the models were trained over the 3 feature sets. The models were evaluated based a combination of recall and precision – the F1 score.

5.1 Sampling

5.1.1 Strategy

Imbalance is usually dealt by using multiple methods like changing evaluation metric and resampling datasets. Oversampling involves

generating and adding copies of the minority class. This allows the algorithm to train a model using more samples of the minority class. Under-sampling is removing observations of the majority class. Another technique to deal with imbalance is generating synthetic samples. In our project we performed under-sampling on the training set using the 'imblearn' framework from scikit learn. Imblearn enabled efficient random sampling during model implementation. Under-sampling increased the overall sensitivity of the models and reduces noise by eliminating rows. There are two side effects of under-sampling. Firstly, under-sampling perturbs the a priori probability of the training set. This induces a warping in the posterior distribution. Secondly, the overall model accuracy gets affected.

5.1.2 Under-Sampling ratio

Upon selecting under-sampling as our method, the next step was a study on which under-sampling ratio addresses the core objective of the predictive model. The core objective as mentioned in the Introduction is to minimize risk of approving defaulters. Maximizing approval of worthy applicants is a secondary objective.

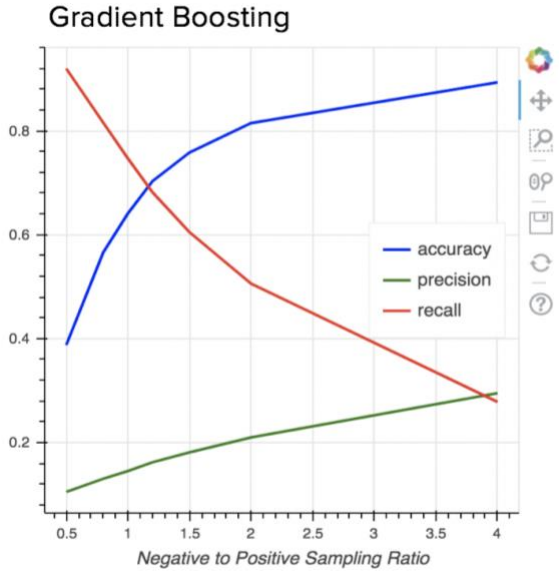

Figure 12: Varying Sampling Ratio on Gradient Boosting Model

From the relationship shown on Figure 12, a 1 to 1 ratio was chosen between Negative (0) cases and Positive (1) cases. This ratio **maximized Recall** which addresses the objective of minimizing risk (false negatives).

5.2 Evaluation Metrics

The models are evaluated based on the recall and precision scores. The objective is to predict the defaulters accurately. Accuracy is not a strong indicator of model performance as the high imbalance would always give a high accuracy. Higher accuracy would classify observations of label '1' as label '0'. This means that defaulters would be classified as potential customers. Issuing loans to defaulters would thereby increase the debt on the bank. Hence the focus was more on recall scores. The trade-off is as follows; a bank would afford to lose someone who is financially strong enough to pay back the loan rather than giving loans to defaulters. Losing customers would only reduce profits. It would not affect the amount of debt on the bank. On the contrary, giving loans to defaulters could potentially increase the debt of the bank. Thus, increasing recall score, i.e. reducing the number of false negatives is of more importance. At the same time, we will have to train the model in such a way that it prevents predicting all of the customers as defaulters just to get a high recall score. We will have to reduce both False Positives as well as False Negatives, thus giving us a high precision and recall. Considering both these metrics, we have another metric termed as F-1 score which is the harmonic mean of recall and precision. This was used while model selection.

5.3 Test-Train Split and CV Strategy

We decided to split the dataset as 70% for training and 30% for testing. To ensure consistency and lack of over/underfit in a model, we performed a 5 fold cross validation for every model. Validation sets were generated from the training dataset itself. Note that the sampling operations were only performed on the training dataset. The testing data was only used while calculating final recall and precision scores of a model.

5.4 Algorithm Roster

We implemented Logistic Regression, Random Forest, Gradient Boosting, SVM and Neural Network algorithms using sklearn package of Python. The developed models were evaluated based on recall and precision scores. The algorithm that gave the highest recall and precision scores for a particular feature set was selected as the final model.

5.5 Logistic Regression

5.5.1 Implementation

The scikit learn "LogisticRegression" function was utilized in for this implementation.

Logistic Regression is a probabilistic classifier which output probabilities of defaulting for each applicant. Using the standard cut off of 0.5, the final class labels were assigned.

Feature sets 1, 2 and 3 were all used and evaluated in this implementation.

Initially the default Logistic Regression was implemented for each of the feature sets for their respective sampled and unsampled versions.

Each Feature Set was also hyper-tuned for the L2 regularization parameter using scikit learn's GridSearchCV procedure. The scope for the L2 given as [0.1,0.01,0.001,0.0001].

5.5.2 Performance

After tuning of the regularization parameter, the following table summarizes the performance of all the different cases

Table 1: Logistic Regression Performance

Feature Set	Before Sampling (CV train)				After sampling (train)				After sampling (test)			
	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1
Set 1	0.9187	0.0000	0.0000	0.0000	0.5795	0.6080	0.1127	0.1902	0.5773	0.6769	0.1511	0.2482
Set 2	0.9190	0.0120	0.5738	0.0235	0.6832	0.6730	0.1594	0.2581	0.6823	0.6794	0.1561	0.2539
Set 3	0.9189	0.0170	0.5428	0.0330	0.6886	0.6838	0.1628	0.2630	0.6877	0.6752	0.1579	0.2560

NOTE: Large size copy of this table is attached in the Appendix

As expected, Feature Set 3 yielded the best metrics on accuracy, recall, precision, and F-1 score. Feature Set 3 provides an improvement from the null feature set of 3% improvement.

While 3% looks miniscule, for a business that is qualifying thousands of loans each day, this scales up in minimizing risk and maximizing profit. To quantify the extent to which a 3 % improvement in f1 score will benefit the business, a separate study will have to be done in the future beyond the scope of this project.

5.6 Support Vector Machine

5.6.1 Implementation

The model was initially built on the training set. Once, the training set was resampled, the model was fitted using the resampled dataset. The hyper parameters that were tuned for SVM were the kernel type and C value using 10 fold cross validation. Kernel type governs the shape of the decision boundary. C values decides how much the model should avoid misclassification. It controls the tradeoff between achieving low error on training data and minimizing the norm of the weights.

Once the model was tuned with the best parameter values, it was used to calculate the metric scores of training and testing data. Table (X) states the values of SVM model on all the feature sets.

5.6.2 Performance

The model performance increases across all three feature sets. From the reported values, we can infer that feature set 3 gives the optimum values for the model.

Table 2: SVM Performance

Feature Set	Before Sampling (CV Train)				After Sampling(Train)				After Sampling(Test)			
	Acc.	Prec.	Recall	F1	Acc.	Prec.	Recall	F1	Acc.	Prec.	Recall	F1
Set 1	0.9204	0.0000	0.0000	0.0000	0.6595	0.1478	0.6859	0.2278	0.6540	0.1349	0.6309	0.2089
Set 2	0.9198	0.0000	0.0000	0.0000	0.6265	0.1578	0.7517	0.2414	0.6214	0.1440	0.7028	0.2385
Set 3	0.9201	0.0000	0.0000	0.0000	0.6877	0.1608	0.6975	0.2638	0.6827	0.1464	0.6296	0.2417

NOTE: Large size copy of this table is attached in the Appendix

The F1 score increases consistently across every feature set. There is a 15.6% increase in the F1 values from set 1 to set 3. Feature set 3 captures more variance as it is a combination of 3 datasets. It captures data regarding bureau balance and thereby feature set 3 could be explaining more variance and giving higher F1 scores.

5.7 Gradient Boosting

5.7.1 Implementation

The scikit learn "GradientBoostingClassifier" framework used for this implementation.

To start with, a default blanket Gradient Boost model was built for each of the feature sets. In order to assess how each of the feature sets fared, the accuracy, precision, recall and f1 score were extracted for the unsampled and sampled versions for each feature set.
Upon doing this, it was determined that feature set 3 performed the best based on the f1 value. As a result, the model was tuned for feature set 3 by utilizing scikit learn's 'GridSearchCV' function.

GridSearchCV allowed a 5 fold cross validation to tune for the following parameters:

1) Number of Estimators
2) Max Depth of Tree Estimators
3) Min Leaf Weight Fraction
4) Learning Rate

Other parameters such as loss function (deviance/exponential), minimum samples in leaf, and maximum samples for split were experimented with outside of the grid search cv procedure but their variation in the default values did not significantly affect metrics.

The number of parameters included in GridSearchCV had to be limited due to the unfavorable time complexity which resulted from adding more parameters which may not make that big of an impact.

These parameters were tuned using the f1 metric to determine the best set of parameters.

The grid search tuning procedure yielded the following parameters for maximizing f1 score:

1) Number of Estimators - 150
2) Max Depth of Tree Estimators - 5
3) Min Leaf Weight Fraction – 0.0
4) Learning Rate – 0.1

5.7.2 Performance

After tuning of the best models, the following table summarizes the performance of all the different cases

Table 3: GBM Performance

Feature Set	Before Sampling (CV train)				After sampling (train)				After sampling (test)			
	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1
Set 1	0.9219	0.0492	0.8153	0.0929	0.6932	0.7189	0.1706	0.2757	0.6944	0.7282	0.1695	0.2750
Set 2	0.9222	0.0498	0.8869	0.0942	0.6939	0.7247	0.1718	0.2778	0.6950	0.7324	0.1704	0.2765
Set 3	0.9229	0.0639	0.8279	0.1185	0.7004	0.7332	0.1765	0.2845	0.7010	0.7411	0.1747	0.2828

NOTE: Large size copy of this table is attached in the Appendix

While Table X shows the raw metrics, it is worth noticing that from each feature set gets progressively better than the previous feature set in all the 4 metrics for all cases. This suggests that the feature engineering played a role in improving predictive performance.

Amongst the 3 sets, Feature set 3 provided 3% improvement on the null set (Feature Set 1). While 3% looks miniscule, for a business that is qualifying thousands of loans each day, this scales up in minimizing risk and maximizing profit. To quantify the extent to which a 3 % improvement in f1 score will benefit the business, a separate study will have to be done in the future beyond the scope of this project.

5.8 Random Forest

5.8.1 Implementation

Initially, the random forest was fitted on the original training dataset. Since the dataset is unbalanced, we got a high accuracy but very low precision and recall. The numbers can be seen in the performance section. Then we fit the model on the under sampled data. The model was tuned using cross validation. The hyper parameters considered are the number of variables considered and the max depth of each tree. The total number of trees are taken as the default value, that is, 500 ensembles models. Further increase in trees has no positive effect on the performance of model. The fitted model was then used to predict the test set.

5.8.2 Performance

The table below shows the accuracy, recall, precision and the F-1 score for all the 3 feature sets.

Table 4: Random Forest Performance

Set	Before Sampling (CV train)				After sampling(train)				After sampling (test)			
	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1
Set 1	0.9187	0.0000	0.0000	0.0000	0.6604	0.6918	0.1570	0.2417	0.6576	0.6728	0.1447	0.2382
Set 2	0.9192	0.0081	0.7987	0.0161	0.6888	0.6668	0.1599	0.2580	0.6838	0.6509	0.1522	0.2468
Set 3	0.9187	0.0000	0.0000	0.0000	0.6745	0.6871	0.1568	0.2591	0.6707	0.6514	0.1581	0.2469

NOTE: Large size copy of this table is attached in the Appendix

Form the table, we can see that there is an increase in the F-1 score as we change the feature sets and is the highest when we include 3 different datasets in feature set 3. This validates our claim that adding different datasets provides more information while predicting the target variable.

5.9 Artificial Neural Network

5.9.1 Implementation

The Keras framework was utilized in implementing this neural network.

We tried our hands on using Neural Networks to predict the target variable. Our goal was to analyze the power of a neural network. Hence, we did not go into minute details while modeling. For our artificial neural network, we used one input and output layer, and two hidden layers (two was chosen to minimize computational complexity). The number of nodes in the hidden layer is taken as the average of the input and the output nodes (experimental). The weights were initialized using the uniform function. The activation function for the hidden layer is the Rectifier Activation Function (Relu). Since we want the probabilities of the customer to default, we took sigmoid function as the activation function for the output layer. Logarithmic loss was used as the loss function to find the optimal weights.

The ANN was tuned using Grid Search. The parameters tuned were the batch size, number of epochs and the type of optimizer used for modeling. Also, to minimize overfitting, 5 – fold cross validation was used while tuning.

5.9.2 Performance

We used the ANN only on the first feature set due to computational constraints. Our goal was to test the power of a neural network.

Table 5: Neural Network Performance

ANN	Training Set (CV train)				Testing Set			
	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1
Set 1	0.9434	0.7590	0.2193	0.3402	0.8993	0.7118	0.1785	0.2854

From the table we can see that we got an F-1 score of 0.3769 which is the highest compared to any other model. The value that we got is without using the under-sampling technique. This tells a lot about the power of a ANN. But since our focus was both, prediction and inference and as we all know that the ANN acts as a black box, we will not go ahead with it as our final model.

5.10 Overall Performance Comparison

Table 6: Model Performance Comparison

Model	Logistic Regression	Support Vector Machine	Random Forest	Gradient Boosting	Artificial Neural Network (Set 1)
F-1 score	0.2560	0.2417	0.2469	0.2828	0.2854

The table shows the accuracy, precision, recall and F-1 scores for all models before and after sampling, and the testing set. We have the different testing set F-1 scores for different models. This metric can be used to compare different models as it is the harmonic mean of recall and precision, which are the two important metrics we considered while tuning the models. Based on the F-1 scores, we can see that Gradient Boosting is the best Model for predicting the customers that could default. The result is convincing as Boosting is an ensemble method which combines multiple weak learners to get one strong learner. The aggregation reduces the variance of the model thus giving the highest F-1 score. Random Forest is also an ensemble method but is lower than Gradient Boosting as it gives equal weightage to all trees unlike Gradient Boosting. Hence, for prediction purposes, we will select Gradient Boosting as our final model.

Artificial Neural Network gave the best F-1 score for the first feature set. It predicts the defaulters much better than gradient boosting. However, it is not chosen as the final model as it acts as a black box making inference difficult, and high computational complexity.

6. Insights

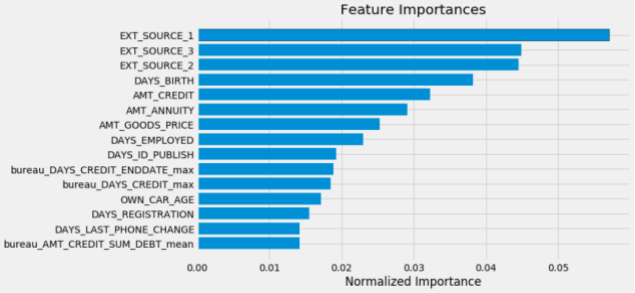

Figure 13: Variable Importance Plot

Credit approval and evaluation is the major process that banks perform to evaluate the strength of an applicant's profile while issuing loans. Granting the loan depends on 2 factors; the willingness of the bank to lend money in the current economy and the applicant's ability to pay back the loan. The state of the economy from a lender's point of view is governed by macroeconomic factors which influence interest rates of the banks. Macroeconomic factors are influential fiscal values or fiscal events monitored by governments and businesses. Factors like inflation and especially fiscal policies govern various rates like cash reserve ratios, statutory liquidity ratios, repo rate and reserve repo rates. These rates directly impact the interest rates determined by the banks while issuing loans. However, that data wasn't provided so we couldn't draw inferences regarding the impact of interest rates and fiscal rates on loan issuances.

The second major factor is the candidate's profile. There are 5 C's; credit, capacity, collateral, capital and conditions that are considered by banks for evaluating a candidate. Based on the given data, our model has successfully selected the important features.

Based on the variable importance plot, 'EXT_SOURCE_1', 'EXT_SOURCE_2' and 'EXT_SOURCE_3' describes the credit history score ratings of the candidate. Naturally these variables are of higher importance as these scores record an individual's ability to manage credit and make payments on time. These scores serve as a risk indicator for the banks. These scores are calculated on a proprietary basis based on the firm. Hence, there importance is varying. The importance might be governed by the company's reputation.

'DAYS_BIRTH'(the age of the applicant), 'DAYS_EMPLOYED', 'AMT_CREDIT', 'AMT_ANNUITY', 'bureau_DAYS_CREDIT' & 'bureau_AMT_CREDIT_SUM_DEBT' describe the capacity and conditions of an individual in terms of their ability to pay back the loan and annuity amounts on time.

Variables like 'OWN_CAR_AGE' and 'AMT_GOODS_PRICE' provide data necessary to select viable collateral items. The dataset does not capture any information regarding investments of the individuals. Data related to an applicant's investments in stock markets or properties would further help the bank to evaluate a profile properly.

7. Project Outcome Evaluation

The project gave the team valuable experience in terms of working with a real world dataset which involved significant data cleaning, processing, exploration and implementation.

The major issue was dealing with the imbalance of the dataset. Working with this dataset didn't only introduce us to solving imbalance; an omnipresent issue in business datasets, but also towards selecting which solution out of the possible ones is the more appropriate solution.

Before the project, the course assignments focused on writing a code from scratch and also having the right computational time and space. The use of libraries like scikit and sklearn allowed that focus to shift from implementation of algorithms to tuning a model and inferring the model. We learnt a great deal in terms of selecting the right hyper parameter to tune and evaluating how the model behaves post tuning.

Lastly, the course focused on the science behind Artificial Neural Networks (ANN) but we never implemented the model practically. Implementing the ANN provided another point of view to understand the model. The ANN model provided better values without any sampling methods albeit that came at a cost of higher computational time.

8. Member Contributions

The project was a cumulative effort of each team member. It was team effort, and everyone was equally enthusiastic towards reaching our result. In the initial weeks, all of us met and brainstormed for ideas until we got an idea to submit in the proposal. In the following weeks, we coordinated closely to formulate a proper plan of attack to address the problem statement.

The Exploratory Data Analysis and Data Preprocessing was divided equally between the three of us. Since we had over 250 variables, each was assigned a set of variables to work with. Regarding the modeling techniques, Anas Patankar trained the Random Forest and the Artificial Neural Network, Kaushik Manchella performed the Logistic Regression and Gradient Boosting and Varad Satam trained The Support Vector Machine. After the modeling techniques, inference and model selection was done by all of us together for coherent reasoning. The highlight of the project was the coordination we had throughout the project.

APPENDIX – RESULTS TABLE

Logistic Regression

Feature Set	Before Sampling (CV train)				After sampling (train)				After sampling (test)			
	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1
Set 1	0.9187	0.0000	0.0000	0.0000	0.5795	0.6080	0.1127	0.1902	0.5773	0.6769	0.1511	0.2482
Set 2	0.9190	0.0120	0.5738	0.0235	0.6832	0.6730	0.1594	0.2581	0.6823	0.6794	0.1561	0.2539
Set 3	0.9189	0.0170	0.5428	0.0330	0.6886	0.6838	0.1628	0.2630	0.6877	0.6752	0.1579	0.2560

Gradient Boosting

Feature Set	Before Sampling (CV train)				After sampling (train)				After sampling (test)			
	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1
Set 1	0.9219	0.0492	0.8153	0.0929	0.6932	0.7189	0.1706	0.2757	0.6944	0.7282	0.1695	0.2750
Set 2	0.9222	0.0498	0.8869	0.0942	0.6939	0.7247	0.1718	0.2778	0.6950	0.7324	0.1704	0.2765
Set 3	0.9229	0.0639	0.8279	0.1185	0.7004	0.7332	0.1765	0.2845	0.7010	0.7411	0.1747	0.2828

Random Forest

Feature Set	Before Sampling (CV train)				After sampling(train)				After sampling (test)			
	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1
Set 1	0.9187	0.0000	0.0000	0.0000	0.6604	0.6918	0.1570	0.2417	0.6576	0.6728	0.1447	0.2382
Set 2	0.9192	0.0081	0.7987	0.0161	0.6888	0.6668	0.1599	0.2580	0.6838	0.6509	0.1522	0.2468
Set 3	0.9187	0.0000	0.0000	0.0000	0.6745	0.6871	0.1568	0.2591	0.6707	0.6514	0.1581	0.2469

Artificial Neural Net.

ANN	Training Set (CV train)				Testing Set			
	Acc.	Recall	Prec.	F-1	Acc.	Recall	Prec.	F-1
Set 1	0.9434	0.7590	0.2193	0.3402	0.8993	0.7118	0.1785	0.2854

Support Vector Machine:

Feature Set	Before Sampling (CV Train)				After Sampling(Train)				After Sampling(Test)			
	Acc.	Prec.	Recall	F1	Acc.	Prec.	Recall	F1	Acc.	Prec.	Recall	F1
Set 1	0.9204	0.0000	0.0000	0.0000	0.6595	0.1478	0.6859	0.2278	0.6540	0.1349	0.6309	0.2089
Set 2	0.9198	0.0000	0.0000	0.0000	0.6265	0.1578	0.7517	0.2414	0.6214	0.1440	0.7028	0.2385

Set 3	0.9201	0.0000	0.0000	0.0000	0.6877	0.1608	0.6975	0.2638	0.6827	0.1464	0.6296	0.2417

PERFORMANCE COMPARISON:

Model	**Logistic Regression**	**Support Vector Machine**	**Random Forest**	**Gradient Boosting**	**Artificial Neural Network (Set 1)**
F-1 score	0.2560	0.2417	0.2469	0.2828	0.2854

APPENDIX II – Data Source Schema