Home Credit Default

Applied Data Science - Fall 2018
Carnegie Mellon University

Faculty Advisor
Prof Artur W. Dubrawski

TA Advisors
Karen Chen
Jieshi Chen
Keerthana Gopalakrishnan

Team
Rishab Narang (rishabn)
Karan Shah (kshah1)

1. Executive Summary

In 2008, the US economy crashed due to a large mortgage crisis that caused financial turmoil around the world. The main cause of the crisis was that many individuals were unable to pay back their loans due to bad credit. In hopes of not repeating history, the financial institution, Home Credit, has hired us to unlock new dimensions of their data and build models to make more efficient predictions. Therefore, our main priorities include using classification models to accurately determine loan defaulters and expand the context to only include new applicants with limited prior loan history.

Initially, we performed extensive data preparation and cleaning since the applicant information was dispersed across multiple CSVs. After merging multiple data files, and dropping features with too many missing values, the final dataset (23000 * 112) consisted of first-time applicants with 0 loan history or 1 previous loan. Next, we processed the categorical features by performing one-hot encoding and creating an initial train/test split (75/25). Next, we shortlisted models we wanted to use for this problem. Since we wanted to see the differences between using a parametric model, a non-parametric model, and a tree-based model, we included logistic regression, naive bayes, k nearest neighbors (instance based), and random forests. Once we determined the models we wanted to use, we performed hyperparameter tuning in order to determine the best k for KNN, and the max depth for random forests. After reviewing the problem, we were able to identify that false negatives were more costly than false positives and created a cost metric that finds an optimal balance between the opportunity cost (value) and cost of lending to a defaulter (risk). Then we trained each of these models on our train dataset, and using 5 cross validation process to depict an ROC curve of each model. Based on this initial ROC curve and cost metric, Naive Bayes performed the worst with an AUC of 0.53 and Logistic Regression performed the best with an AUC of 0.73. We repeated the model fitting and validation two more times, incrementally modifying the process. The second time we resolved class imbalance, and the third time, we ran the entire machine learning pipeline process after resolving class imbalance and including feature engineering techniques like removing correlated features and including domain specific features. After comparison, Random Forest was the best model with AUC and cost of 0.74 and $96,170. Finally, we built our best model, Random Forest, predicting on the test dataset. The AUC score on the test dataset was 0.66 which was approximately a 32% improvement over default classifier. We were able to determine the following features as the most impactful EXT_SOURCE_2 (normalized score), DAYS_BIRTH, LOAN_TERM, AMT_CREDIT, etc. Furthermore, we were able to determine the most optimal cutoff as 0.48 by determining that at this cutoff, the cost metric was at its lowest. Compared to the other models, at this point the cost was $39,530. Hence, using Random Forest with a max depth of 10, we were able to obtain a recall of 0.82 (cutoff = 0.48).

The information gain that Home Credit receives about new customers from our model can help mitigate some of the uncertainty in determining whether they new customers should receive a loan or not along with a more stable process of predicting repayment status for the current ones.

2. Goals

1. Accurately predict home loan repayment status for current customers using various classification data mining techniques. In other words, our main goal is to determine whether a client will be able to repay the home loan.
2. Build a credible model that expands the current solution to also predict defaulting for **new** applicants while minimizing credit loss to Home Credit.

3. Data Description

a. Application_test.csv/Application_train.csv: Contains information about loan application when it is submitted to HomeCredit. Static data for all applications. One row represents one loan in our data sample.
b. Bureau.csv: Our client's previous credits approved by other financial institutions
c. Bureau_balance.csv: History of client's monthly balances for the previous credits for other financial institutions
d. POS_CASH_balance.csv: Client's previous credits' monthly balance, point of sales, and details for the loans issued by HomeCredit.
e. Credit_Card_Balance.csv: Monthly balance and details of credit that client had with HomeCredit.
f. Previous_Application.csv: All the applications of the clients that were submitted to HomeCredit.
g. Installment_Payments.csv: Repayment history for the previously issued loans by HomeCredit. One row per month for completed/missed payment.

4. Exploratory Data Analysis

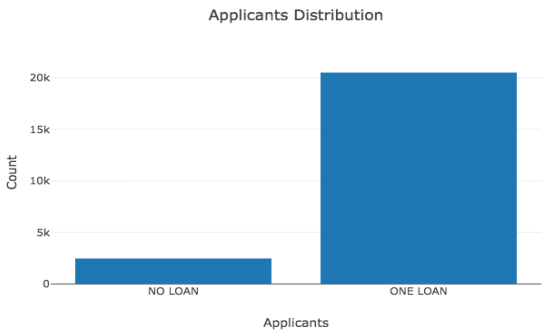
Data Preparation & Data Cleaning

Since we don't have any truth labels in our Application_test.csv, we decided to perform all of our analysis using Application_train.csv. Because we wanted to specifically target new customers, customers who recently took out a loan with Home Credit, but have very little previous credit history with Home Credit and other financial institutions, we needed to perform an extensive amount of data manipulation. We essentially had to perform multiple joins, set differences, and intersections between all the files in order to procure our specified dataset consisting of new customers. Next, the following three categorical features, WALLSMATERIAL_MODE, FONDKAPREMONT_MODE, EMERGENCYSTATE_MODE, were dropped from out dataset because there was no variance among the different categories in each feature. In addition, we dropped 6 additional features where the percentage of missing values were greater than 70%. The resulting dataset consisted of approximately 23000 records, and 112 features. 13 of the features were categorical (max arity: 58, min arity: 2), and the remaining 99 were numerical.

Data Visualization

The first thing we found when performing data visualization is that there is a class imbalance in our data. There is a much higher count of people that did not default (Target: 0), and a much lower count of people that did default (Target: 1). To resolve class imbalance, we randomly sampled the majority class to match the number of samples in the minority class in order to create a more balanced dataset. Specifically, we only fixed class imbalance on the training dataset during cross validation, and never on the validation dataset.

In addition, more applicants in our dataset have one previous loan, than no loans at all.



Other interesting findings:

a. 82.6% of loans are cash loans whereas 17.4% are revolving loans (no predetermined amount, usually credit card loans and overdrafts)
b. 13,307 applicants are married, 1321 are separated, 4902 are single/not married, 2 are unknown, 1071 are widows, and 2365 had a civil marriage.
c. 11,435 applicants (50.3%) are from the working class, 7 applicants are businessmen, 4304 applicants are pensioners, 1296 are state servants, 4 are students.
d. 80% of defaulters have an education level of secondary, 2.39% of defaulters have lower secondary education, 13.8% have higher education, and 3.89% have incomplete higher education.
e. Top defaulters (33%) are laborers and the second on that list is sales staff (17.6%)
f. Majority of our defaulters (32%) are in the age of 20-30. Also, the younger you are, more likely you are to default (from our analysis).

5. Model

The task is classification. The output column is **TARGET** with an arity of 2

0 represents non-defaulters

1 represents defaulters

Methodology

Step 1: One Hot Encoding

We processed the categorical data using one hot encoding. The categorical variables are converted into a binary vector form which can be used better by the model.

Shape of data before one hot encoding: 22968 rows * 112 columns
Shape of data after one hot encoding: 22968 rows * 195 columns

Step 2: Train/Test Split

We split the data set into 75%-25% train/test split. This split is done randomly to ensure any form of bias on splitting is taken care of. The test split is kept aside to later test the final model on this data set.

X_train: DataFrame of all the X features for training: 17226 rows * 194 columns
y_train: Training target: 17226 rows * 1 column
X_test: DataFrame of all the X features for testing: 5742 rows * 194 columns
y_test: Testing target: 5742 rows * 1 column

Models compared: Logistic Regression, Random Forest, Naive Bayes, and KNN

Step 3: Hyper Parameter Tuning using Cross Validation

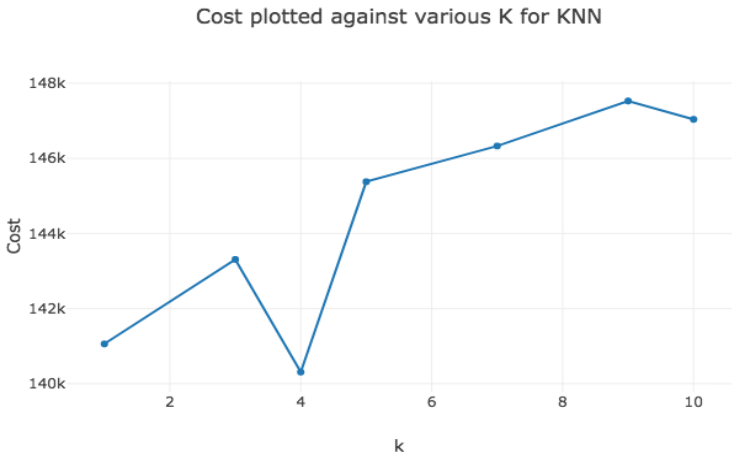
We perform hyper parameter tuning for the given models to explore the optimal setting for the classifier from a range of possibilities.

Description of CV Function

1. Takes in X_train data. Remove the SK_ID_CURR (Identifier for applicants) from the training set.
2. Shuffles the data randomly.
3. Splits the data into k folds.
4. For k folds:

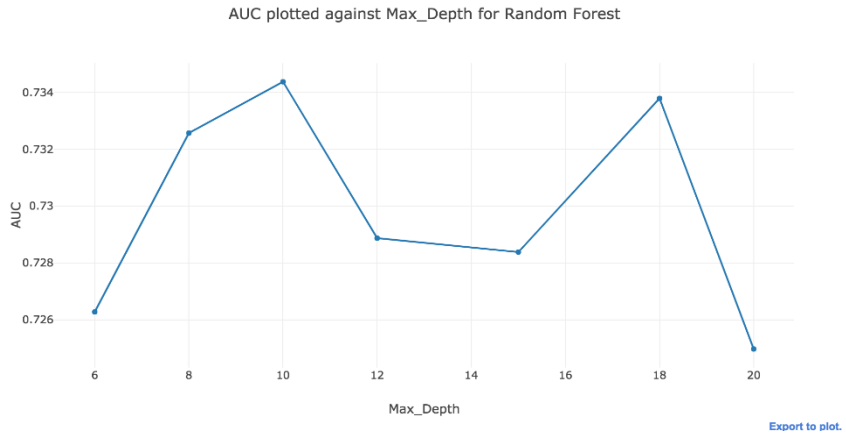
a. Hold the k-th set for testing.
b. Using the median strategy, impute the data to fix the missing values.
c. After imputation, the features need to be scaled between 0 and 1.
d. Once the imputation and scaling have been done, the corresponding model is called.

Here we run models for various K (Knn) and Max_Depth (Random Forest) to find the optimal parameters for the models.



Task: Find the best k for KNN using different neighbors

For KNN, at k = 4, we observe the lowest cost. The number of FN's is 1471 and FP's is 792.



Task: Find the best Depth for Random Forest using Cross Validation

Best Depth = 10 (Choose the best AUC at a low depth)

Step 4: Pass Models (Logistic Regression, Knn, Naive Bayes, and Random Forest) with the optimal parameter from Step 3 to a Cross Validation Class to perform model fitting and prediction

Return: The results are returned in a data frame with columns storing the actual labels, predicted probabilities, and the fold along with the SK_IDs.

Step 5: Pass the results from Step 4 to a metrics calculation function.
The generic function can be used to perform two operations:
1. Plot the ROC Curve for the model passed.
2. Compute metrics like accuracy, precision, recall, True/False Negatives/Positives, and Cost at certain threshold

Return: This function returns a data frame for metrics or a plot for ROCs

Target Metric : Cost
After careful considerations, we realized that the **cost of making false negatives** is **higher** than the **cost of false positives**. This means if we predict that the applicant is not going to default whereas actually, the applicant does default - we end up losing a lot more money than missing on the opportunity. **We want to find the right balance between opportunity cost and risk cost.** Trying out different weights along with the underlying data, the following choice of weights describe a realistic picture of risk vs opportunity cost.

```
                    AVG COST PER CUSTOMER IN MATRIX FOR EVALUATION

                  ACTUAL CLASS     POSITIVE(DEFAULT)     NEGATIVE (NOT DEFAULT)
  PREDICTED CLASS
  POSITIVE(DEFAULT)                      0   (TP)                    10 (FP)
  NEGATIVE (NOT DEFAULT)                 90 (FN)                     0 (TN)


Total Cost = Finding the right balance between opportunity cost and cost of accepting a potential defaulter.
           = C(FN) * #FN + C(FP) * #FP
GOAL: To choose the model with lowest total cost
```

This metric comes in handy, especially for managers at Home Credit who may not be an expert in understanding the notions of False/True Positives/Negatives etc. We also review AUC from the ROC curves to validate our cost notion is correct.

Step 6: Step 4 and Step 5 are repeated three times:
1. Before fixing class imbalance or performing feature engineering:
   The train set is passed to the CV function without fixing the class balance or performing any sort of feature engineering. The results are shown in the A column of the figure below. We see Logistic Regression performs the best and Naive Bayes performs the worst on both AUC and Cost metrics.

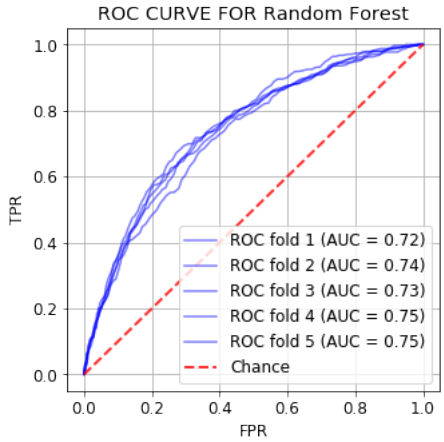
2. After fixing class imbalance but not performing feature engineering:
   The train set passed to the CV function has a logic in place which fixes the class imbalance on the training set, the hold out set for CV does not have class imbalance fixed. The results are shown in the B column of the figure below. We witness Random Forest performs the best and Naive Bayes performs the worst on both AUC and Cost metrics.

3. After fixing class imbalance and performing feature engineering:
   Added Features: We added a few domain features like loan percentage of income, annuity percentage of income, loan term, and percent of days employed to the data set (researched about these domain features from off kaggle).
   Removed Features: We calculated correlations between the features and removed features which had correlations above 70%. In total, 46 features were dropped after performing this operation.

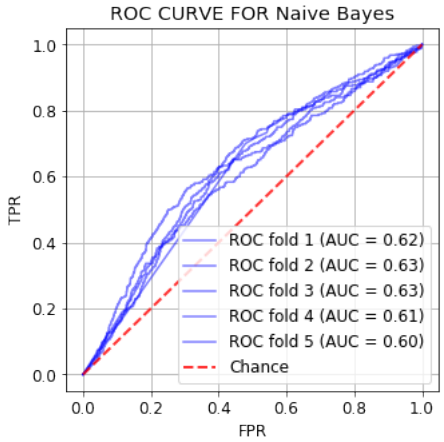
Model Comparison

	A		B		C			
MODEL	**Before fixing class imbalance or performing feature engineering**		**After fixing class imbalance but not performing feature engineering**		**After fixing class imbalance and performing feature engineering**		**Improvement over the Worst Model (For C)**	
	AUC	**COST**	**AUC**	**COST**	**AUC**	**COST**	**AUC %**	**COST %**
Logistic Regression	0.723	142,350	0.728	101,360	0.728	101,790	27.4956217	5.252817702
Naive Bayes	0.526	150,870	0.602	130,970	0.571	136,430	0	41.07124393
KNN	0.557	146,220	0.607	127,900	0.585	135,450	2.45183888	40.05790508
Random Forest	0.714	148,060	0.73	100,080	0.739	96,710	29.4220665	0
	BEST	WORST						

The best model here is Random Forest (column C) because the AUC for the model is the best and the cost is also minimum. The right column shows improvement of AUC and COST from the worst models.



Best Model: Random Forest

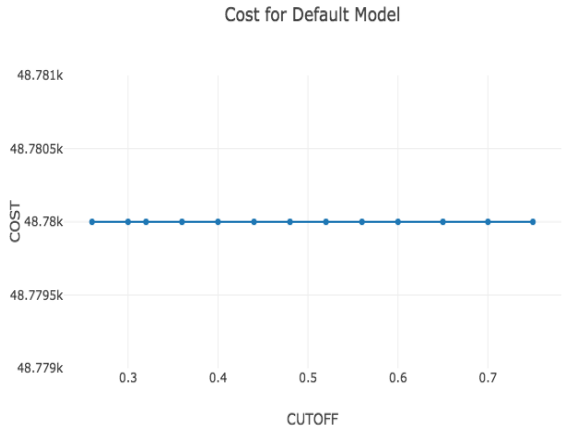


Worst Model: Naive Bayes

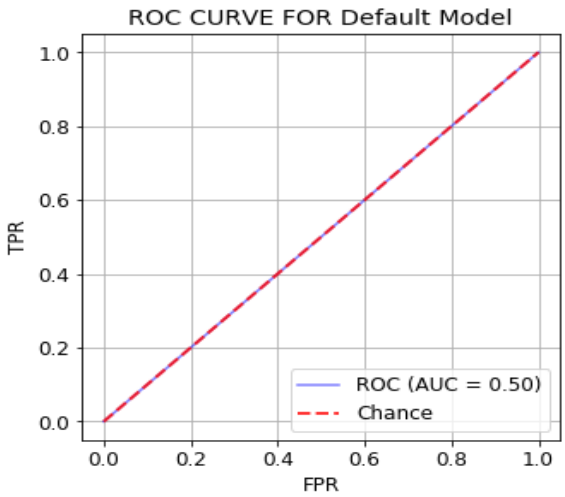
Above, we plotted ROC curve after performing a 5-fold cross validation on Random Forest and ROC curve after performing a 5-fold cross validation on Naive Bayes. Random Forest is the best model for our problem given AUC and Cost metrics whereas Naive Bayes is the worst model for our given problem. Our guess for Naïve Bayes bad performance is that it fails to capture the underlying correlations in the data as it considers features independently.

6. Default Model

Before building the final model, let us build the default model. The performance of final model should be better than the default model. The default model is where it predicts the most common value of the output. As we saw in EDA, label 0 is the most common value. The only error possible in this case is FN.



Cost at various cutoffs for the Default Model

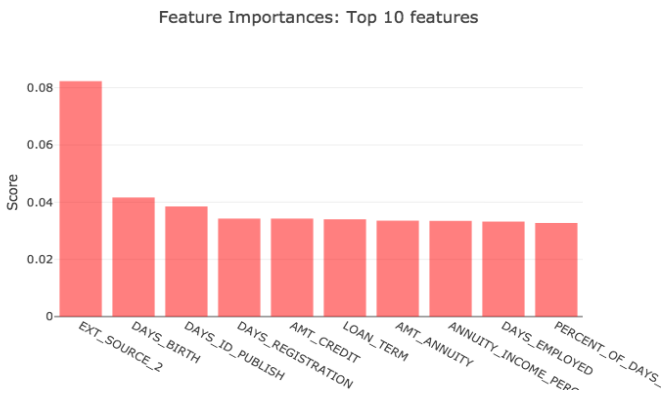


ROC for the Default Model

We see a flat cost line as the number of FN's and FP's stay the same for the given cutoffs. We have 542 FN, 5200TN, TP = 0, and FP = 0

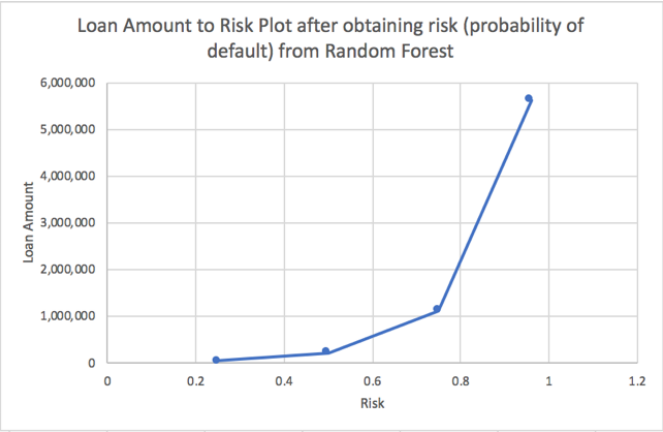
7. Final Model: Random Forest and Results

The final model used is Random Forest. We used 200 estimators for the tree to ensure the model has substantial number of estimators to give us accurate results given the 195 features in the train/test set.



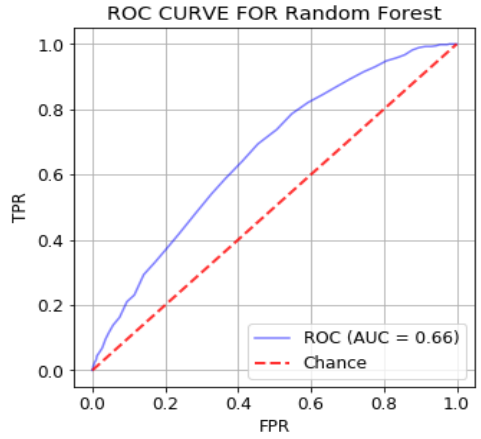
Column Descriptions
DAYS_BIRTH: Client's age in days at the time of application
DAYS_ID_PUBLISH: No. of days before application the client changed the identity document with which loan was applied

Comparing the results from our RF model, we saw feature **DAYS_ID_PUBLISH** for instance have contrasting results for defaulters and non-defaulters. For defaulters, on an average, the people changed the documents 7 days **after** submitting the application. For non-defaulters, on an average, the people changed the documents 2766 days **before** the application. For DAYS_EMPLOYED, defaulters were employed **2042 days** on an average whereas non-defaulters were employed **2265 days** on an average. These discrepancies between the results for the features help us distinguish the two types of applicants.



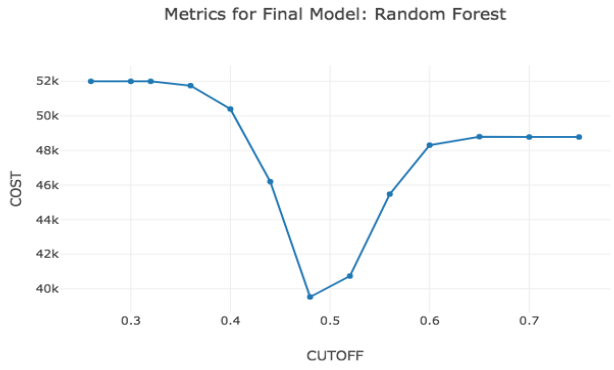
A sanity check to validate the results from the model are correct. We have plotted the loan amount to risk obtained from the model. A synthetic data set with varying amount given all the other parameters as same was passed. We see there is an increase in risk as the loan amount increases.

Model Performance and Metrics

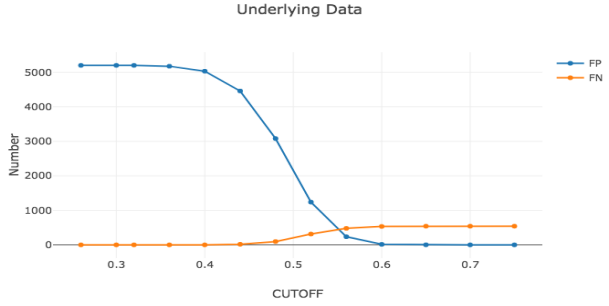


The AUC for the final model is **0.66,** which is a 32% improvement over the default.

AUC	FN	FP	TP	TN	RECALL	MODEL
0.662079	97	3080	445	2120	0.821033	Random Forest



Since recall is more important for us, we want a lower cutoff to make sure we don't give away loans to defaulters. We ran metric evaluations for cutoffs ranging from 0.26 to 0.75 and decided to choose a cutoff of **0.48** because it minimizes the cost at this cutoff. This cost basically ensures an optimal balance between false negatives and false positives.



Our underlying data is imbalanced (majority non-defaulters). We have plotted the number of FP's and FN's at various cutoffs to get a sense of how FP's and FN's change. At 0.48 cutoff, we see there is a slight rise in FN's but the FP's have significantly reduced.

8. Business Implications

First, with our predictive information, the decision maker, Home Credit, now has a credible way of assessing relatively new applicants (customers with 0 or 1 loan). The reliable information we present would allow Home Credit to take on more risk and credit individuals that they have little information about. In turn, the decision maker's risk tolerance parameter would increase because they are more tolerant to risk and much less risk averse. Ultimately, this would result in an increase of their certainty equivalent (the guaranteed amount the decision maker would take in place of the risk). Hence, Home Credit would need a much higher return from the safer option (crediting individuals with good credit history) in order for them to not take the risk (credit individuals with little credit history). On the other hand, the risk premium would decrease. The risk premium is the return associated with the minimum number of new customers that can't default if Home Credit were to take on the risk of crediting new customers. For example, before our assessment, let us assume that Home Credit gives out loans to 10 new customers in hopes of an expected return of $1,000,000. Because they don't have our assessment, Home Credit would be much more risk averse, and they would only take on this risk if they knew they would receive at least $700,000 in returns (7 customers can't default). However, because we can alleviate some of the uncertainty using our results from above, Home Credit (now less risk averse) can take on a smaller minimum return, $500,000 (only 5 customers can't default) in order for them to take the gamble of loaning to 10 new customers in hopes of getting the maximum return of $1,000,000 (all 10 customers don't default). In the previous example, I did not include interest to keep things relatively simple. Finally, because Home Credit would be able to take on bigger gambles without losing as much, this would open up a pool of new opportunities. Primarily, their customer base would increase. However, since we are using a lower threshold than 0.5, we're flagging more applicants as "potential defaulters." These candidates can be further assessed by Home Credit's management team.

9. Risks and Mitigation

Cost of risk varies across the applicants. For instance, an applicant defaulting on a loan worth $10M has a different cost associated in comparison to an applicant defaulting on a $10,000 loan. Not accurately weighing these risks is a risk. This risk can be mitigated by working with Actuarial science team at Home Credit to identify what sort of weights can be used in the cost sensitive modeling process.

10. Recommendations for Continuation Work

Macroeconomic factors: Our model does not take macroeconomic factors like inflation, financial crisis etc. into account, which could be important factors of determining default. In future, Home Credit can include these features to foresee defaults as these factors affect the default behavior.
Application: An user-friendly application can be built for managers to navigate through the results easily, given a set of parameters.
Cost Metrics: We have weighed FN's 9 times to be more costly than the FP's (looking at class imbalance). We want to get a better understanding of the weights, which might help us with more realistic cost metrics. Also, each default would technically have a different cost associated with it, so the costs can be bucketed into different ranges.
LTV Value and Time Series Analysis: Lifetime value of an applicant can be included in the model. This value should be also based upon the updated market value of the mortgaged property. Then using this LTV, segmented loan rate of interest can be offered to reward the loyal customers. Also, a time series analysis can be done on the data to know when would an applicant will default.
Clustering Information: We implemented clustering to understand our applicants better (EDA). In future, the aggregation results from the clusters can be used as features to improve the model.
Dataset Balancing Techniques: Various class balancing strategies like over-sampling, SMOTE etc. can be employed to get a better data sets for modelling.

11. References

Dataset: https://www.kaggle.com/c/home-credit-default-risk