Credit Card Defaultee Analysis

ISOM3360 Group 23: Project Final Report

LAM, Ho Chit
SID: 20607878
hclamao@connect.ust.hk

LEE, Ho Wan Owen
SID: 20604852
hwolee@connect.ust.hk

LEE, Wai Chung
SID: 20702733
wcleeaj@connect.ust.hk

1. Introduction

Credit cards have been extremely vital to modern societies, due to its convenience of carrying out trivial to medium-sized transactions with a swipe of the card. However, due to its nature of "crediting" to cardholders, there have been instances where cardholders (defaultees) fail to repay their debt.

Credit card issuers might therefore suffer from a loss. Despite this, they play a crucial role in creating liquidity and boosting the overall economic welfare. To solve their pain point, potential measures may include reducing the loan size for high-risk customers or performing more meticulous due diligence on the client.

Below are the goals of this project:

- Help credit card issuers identify credit card defaultees
- Take extra precautions on people that are risky of delayed credit repayment

Due to the extremely high dimensionality of each data entry, it is tremendously difficult even for experienced bankers to properly predict possible defaultees at boundary cases. It is hence up to statistical approaches to help uncover potential hidden factors and improve prediction accuracy, hence minimizing economic loss for credit card issuers.

The credit card default prediction problem is a typical "A to B" mapping problem, where A (input) consists of information of a credit card applicant obtained from their application form, and B (output) represents whether or not this applicant will default. Via the use of supervised machine learning techniques, these algorithms can analyze the correlation between A and B and statistically determine a decision boundary beyond human precision.

2. Data Understanding

After rigorous selection, adhering to both the assigned criteria as well as an internal guideline to ascertain data quality, our group has decided to adopt the credit card approval/defaultee dataset, retrieved from https://www.kaggle.com/mishra5001/credit-card .

The data naturally skews significantly towards the non-target side, as it is rather intuitive that most people choose not to default. The skew is challenging for us as it will be difficult to beat the majority classifier, which assumes everyone will not

default. 24,825 applicants out of the whole universe (307,511 applicants) have defaulted, consisting of a mere amount of 8.07%. However, as a false negative classification incurs large cost for the issuer, it is still significant for us to try to maximise the true positive rate. The data quality is quite high, columns are generally meaningful and there are not a lot of missing values.

The data comprises of 112 columns, which is initially quite noisy. We cleaned the missing values with the mean for numerical attributes such as AMT_ANNUITY, CNT_FAM_MEMBERS. For categorical attributes such as OCCUPATION_TYPE, the mode is used to fill the missing values.

There are several attributes with over 10000 missing values, and our team has deemed these attributes to be of inferior quality. These columns were all dropped during the first stage of preprocessing.

The below is an excerpt from the main attributes that we used in our analysis:

- 'TARGET': Target variable (1 - client with payment difficulties: he/she had late payment more than X days on at least one of the first Y installments of the loan in our sample, 0 - all other cases)
- 'EXT_SOURCE_3': Normalized score from external data source (Credit score) - This is significant as this represents the credit score of the applicant. It is similar to the TransUnion score in Hong Kong.
- 'EXT_SOURCE_2': Normalized score from another external data source (Credit score) - This is significant as this represents the credit score of the applicant. It is similar to the TransUnion score in Hong Kong.
- 'DAYS_BIRTH': Client's age in days at the time of application
- 'REGION_RATING_CLIENT_W_CITY': Our rating of the region where client lives with taking city into account
- 'NAME_EDUCATION_TYPE_Higher education': Level of highest education the client achieved is Higher education
- 'NAME_INCOME_TYPE_Working': Clients income type (businessman, working, maternity leave,...) is working.
- 'DAYS_LAST_PHONE_CHANGE': How many days before application did client change phone
- 'NAME_EDUCATION_TYPE_Secondary/secondary special': Level of highest education the client achieved: Secondary/secondary special
- 'MALE_True': Gender of the client (1=Male, 0=Female)
- 'DAYS_ID_PUBLISH': How many days before the application did client change the identity document with which he applied for the loan
- 'NAME_INCOME_TYPE_Pensioner': Clients income type (businessman, working, maternity leave,...) is Pensioner.
- 'REG_CITY_NOT_WORK_CITY': Flag if client's permanent address does not match work address (1=different, 0=same, at city level)
- 'DAYS_REGISTRATION': How many days before the application did client change his registration

- 'REG_CITY_NOT_LIVE_CITY': Flag if client's permanent address does not match contact address (1=different, 0=same, at city level)
- 'AMT_GOODS_PRICE': For consumer loans it is the price of the goods for which the loan is given
- 'REGION_POPULATION_RELATIVE': Normalized population of region where client lives (higher number means the client lives in more populated region)
- 'NAME_CONTRACT_TYPE_Revolving loans': Identification if loan is cash or revolving
- 'LIVE_CITY_NOT_WORK_CITY': Flag if client's contact address does not match work address (1=different, 0=same, at city level)
- 'DEF_30_CNT_SOCIAL_CIRCLE': How many observation of client's social surroundings defaulted on 30 DPD (days past due)
- 'DEF_60_CNT_SOCIAL_CIRCLE': How many observation of client's social surroundings defaulted on 60 (days past due) DPD

Below displays a rough data distribution description regarding these attributes, generated using the pd.DataFrame.describe() function:

	mean	std	min	max
EXT_SOURCE_3	0.51085291	0.1744642	0.00052727	0.89600955
EXT_SOURCE_2	0.51439267	0.19085501	8.17E-08	0.85499967
DAYS_BIRTH	-16036.995	4363.98863	-25229	-7489
REGION_RATING_CLIENT_W_CITY	2.03152082	0.50273703	1	3
NAME_EDUCATION_TYPE_Higher education	0.2434482	0.42916404	0	1
NAME_INCOME_TYPE_Working	0.51631974	0.49973441	0	1
DAYS_LAST_PHONE_CHANGE	-962.85879	826.807143	-4292	0
NAME_EDUCATION_TYPE_Secondary / secondary special	0.71018923	0.45367517	0	1
MALE_True	0.34164306	0.47426133	0	1
DAYS_ID_PUBLISH	-2994.2024	1509.45042	-7197	0
NAME_INCOME_TYPE_Pensioner	0.18003258	0.38421522	0	1
REG_CITY_NOT_WORK_CITY	0.23045354	0.42112384	0	1
DAYS_REGISTRATION	-4986.1203	3522.88632	-24672	0
REG_CITY_NOT_LIVE_CITY	0.07817281	0.26844377	0	1
AMT_GOODS_PRICE	538396.207	369279.426	40500	4050000
REGION_POPULATION_RELATIVE	0.02086811	0.01383128	0.00029	0.072508
NAME_CONTRACT_TYPE_Revolving loans	0.09521285	0.29350919	0	1
LIVE_CITY_NOT_WORK_CITY	0.17955455	0.38381662	0	1
DEF_30_CNT_SOCIAL_CIRCLE	0.14342067	0.44595624	0	34
DEF_60_CNT_SOCIAL_CIRCLE	0.10004894	0.36168886	0	24
OCCUPATION_TYPE_Drivers	0.0604954	0.23840279	0	1
FLAG_WORK_PHONE	0.19936848	0.39952623	0	1

ORGANIZATION_TYPE_Self-employed	0.1249126	0.33062033	0	1
NAME_HOUSING_TYPE_With parents	0.04825844	0.21431218	0	1
NAME_FAMILY_STATUS_Single / not married	0.14778008	0.35488243	0	1
NAME_HOUSING_TYPE_House / apartment	0.88734387	0.31617251	0	1
NAME_INCOME_TYPE_State servant	0.07057634	0.25611625	0	1
NAME_FAMILY_STATUS_Married	0.6387804	0.48035482	0	1
FLAG_OWN_CAR_Y	0.34010816	0.47374606	0	1
HOUR_APPR_PROCESS_START	12.0634189	3.26583226	0	23

3. Model Building

Since this problem clearly has a label of whether or not the applicant will choose to default, supervised machine learning algorithms should be applied. With consideration to model runtime and interpretability, our group has chosen 3 simple models: Decision Tree classifier, Logistic Regression and Naïve Bayes classifier.

Before training of each model, a general preprocessing procedure is applied to the data. We adopted a top-down approach towards data preprocessing, after inspecting the data, we filtered out features that have plenty of invalid entries and inspect the nature of it. For meaningful numerical features like the family members count, we filled the missing values with the mean. And as for categorical features like occupation type, we filled the missing values with the majority value “Laborers”. We also normalised total income amount by minmax_scalar.

In order to maximize the meaning of accuracy as an evaluation metric and to minimize the bias introduced by class imbalance, we have adopted the simplest method of performing downsampling of the negative class. We have decided to downsample the data to the extent where the class ratio between positive and negative labels are 1:1. While data with negative label is sacrificed, we believe that the more significant meaning brought by classification accuracy leads to models and corresponding results with higher interpretability.

Below describes the methodology, parameters and interpretations for each model:

A. Decision Tree

We have first started out with the simplest decision tree model, using all 177 columns of features with cleaned data. Then we came to notice that the model is significantly overfitted and provides no incremental insight to the majority classifier. After some time, we discovered the importance of feature selection and data balancing. So, we fixed the data-imbalance problem by down-scaling the excess data, and we removed some unnecessary features that would potentially cause multicollinearity problems.

After selecting our features and built our second and third decision tree model, we have implemented three methods to train the hyperparameters of the model,

including Manual hyperparameter tuning, Cross-validation, and Gridsearch CV. After comparing their resulted models, we then opted for using the Gridsearch CV method, as it is simple to use and would yield to a better result, despite a longer run-time. The hyperparameters being tuned are max_depth and min_samples_leaf. There are two reasons for tuning only two hyperparameters. First, we might over-tune the model and the model would overfit. Secondly, tuning many hyperparameters with such a big dataset would require enormous computational complexity, resulting in unacceptably long run-time. And, to further reduce run time, we opt for only three-fold cross-validation instead of five that we thought would be appropriate originally.

Below are our tuning results:

- max_depth, or the maximum depth of the tree, is set to 7.
- min_samples_leaf, or the minimum number of samples required to be at a leaf node, is set to 109.

With these hyperparameter values, we obtained a model that can perform with the highest accuracy as well as the largest AUC.

B. Logistic Regression

The exact same data preprocessing procedure for Decision Tree is used for Logistic Regression. To facilitate model comparison, the exact same features and data instances are used for analysis.

GridSearchCV with 10-fold cross-validation is used to tune hyperparameters for logistic regression, taking accuracy and AUC as evaluation metrics. The tuned hyperparameters were 'C', the regularization coefficient, as well as 'penalty', the type of penalty. Ultimately the model with highest AUC is chosen as our selected logistic regression model.

Below are our tuning results:

- C, the penalty factor, is set to 0.12589254117941673
- penalty, the type of penalty, is set to 'l2' (among 'none', 'l1' and 'l2')

The optimal penalty type being L2 is consistent with the general observation that Ridge Regression outperforms LASSO and naïve logistic regression with a noisy data set, due to its ability to penalize heavy parameter coefficients and hence reduce overfitting.

C. Naïve Bayes classifier

The original Naïve Bayes classifer has little predictive power. Upon inspection, we tried to use less features. However, it is not effective as using too little features

causes the model to behave no differently than a majority classifier. Therefore, we seek improvement from tuning the hyperparameter var_smoothing.

To improve the Naïve Bayes Classifier, we used GridSearchCV to tune the hyperparameter "var_smoothing". var_smoothing is a stability calculation factor to widen (or smoothen) the curve to account for samples that are far from the distribution mean. We achieved an optimal parameter of 1 through 10 folds of cross-validation. This means that minimal smoothening must be done.

With hyperparameter tuning, we are able to achieve slightly better results. However, we suspect that data multicollinearity causes this model to perform poorly, which we will elaborate later.

Apart from working on the above 3 models, there had also been certain feasible ideas that were eliminated to adhere to the time span and scope of the project:

- K-Nearest-Neighbour Classifier
    - During the idea proposition stage, we were looking through the course syllabus and proposed using a kNN classifier. However, this idea was dropped due to its long runtime and memory consumption, which is suboptimal for practical usage.
- Outlier detection via K-Means Clustering
    - Our group has debated applying k-Means clustering as a method to eliminate outliers. However, we concluded that we will not apply this method for the following reasons:
        1. We had no idea how to choose the value of k.
        2. There are too many features, hence curse of dimensionality will occur, leading to ineffective use of k-Means clustering.
        3. Since the dataset is extremely imbalanced, and the default cases are likely to be classified as outliers, this goes against our objective of finding these outliers instead of removing them.
- Feature selection using LASSO
    - With LASSO's ability to shrink less relevant coefficients to 0, theoretically LASSO can act as a feature selection method by adjustment of regularization factor. However, due to high multicollinearity in the dataset, LASSO does not effectively identify the best combination of coefficients to shrink. As a result, we ultimately opted to use a simple correlation measure to select the features.

4. Performance Evaluation

This problem of credit card defaultee prediction that we have chosen is of a strong business nature, and we were advised to focus not just on prediction accuracy, but also on cost analysis.

It is noteworthy that the dataset is severely imbalanced, with an extreme majority of credit card holders not defaulting. Hence, accuracy is not a quality measure of model performance, with the majority classifier having an accuracy of roughly 92%. However, with downsampling (explained in detail in Section 3) of negative label data, we can once again observe accuracy as an intuitive and quality measure of model performance.

We have adopted grid search with cross-validation to search for the optimal model hyperparameters in each model, and the ultimate optimal model evaluation were conducted on a separate hold-out test set.

Apart from the typical model performance indicators such as F1 score and AUC score, our group would like to tailor the models to this specific problem. A custom cost function was created, taking into account the estimated economic losses of each misclassified credit card user. Below details an explanation of how we derived and estimated the cost of misclassification through past statistics:

- Case 1: False Negative
    - False negative refers to credit card applicants that will default but is not detected by the model to be possible defaultees.
    - The economic loss of false negative is represented by the financial loss of credit due to default. There will be no further losses since credit card defaultees will typically expect their future applications to be rejected.
    - The amount of financial credit loss can be estimated by using the average annual amount of credit for each credit card user, which is approximately $6500 USD according to US statistics.
    - Hence the cost of false negative is set to be 6500 in our cost function.
- Case 2: False Positive
    - False positive refers to credit card applicants that will not default but is classified by the model to be possible defaultees.
    - The economic loss of false positive is represented by the amount of credit profit forgone due to not accepting the application.
    - The amount of credit profit can be estimated by accumulating the total credit profit over the span of credit card usage. According to data from JP Morgan, credit profit margin is roughly at 4.17% - 4.21% of credit balance (estimated to be $6500 USD). Assuming an average credit card usage span of 15 years and a discount rate of 2.8%, total accumulated credit profit is estimated to be $3306 USD.
    - Hence the cost of false positive is set to be 3306 in our cost function.

By applying various test decision boundaries and optimal model probability outputs to our custom economic cost function, we can obtain an optimal classification decision boundary for each model that results in minimized estimated economic loss. Note that the cost of false negative is much larger than that of false positive,

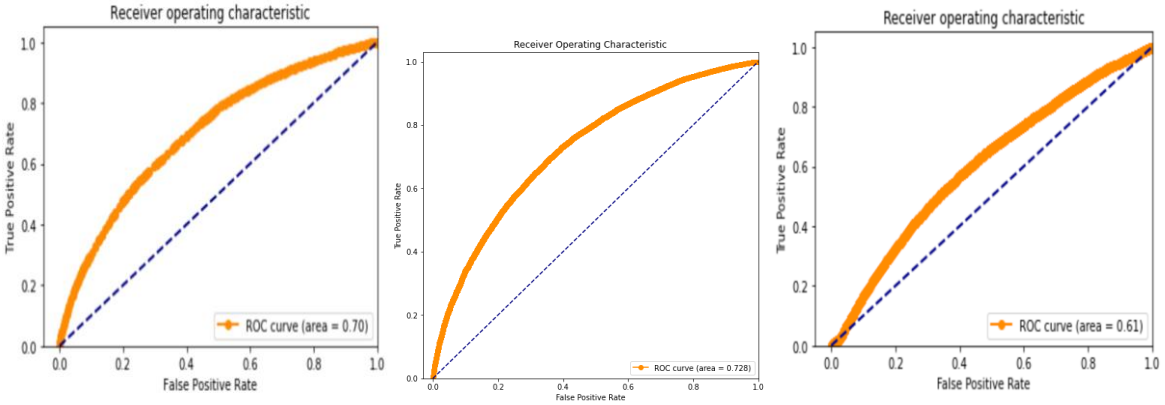
so we can expect the model decision threshold to be skewed towards predicting positive outcomes, i.e. likely to be lower than 0.5.

The performance metrics of each optimized model, evaluated on a balanced test set of 9930 data instances, is summarized in this table:

	Accuracy	F1 score	AUC	Threshold	Economic cost
Decision Tree	0.657603	0.65	0.704368	0.325	$14944844
Logistic Regression	0.663646	0.659878	0.727863	0.35	$14306508
Naïve Bayes classifier	0.64501	0.65	0.68977	0.25	$15794626

We also include the ROC curves of each optimized models:

Decision Tree Logistic Regression Naïve Bayes



From the result summarization above, Logistic Regression, a linear model, has the best performance in all chosen metrics, while a more statistical and probabilistic model like Naïve Bayes has inferior performance. Our group proposes a possible explanation:

- Logistic Regression only performs marginally more superiorly to Decision Tree. Since Logistic Regression has a linear decision boundary and Decision Tree is not likely to be similar, it is possible that the true decision boundary is more linear than nonlinear. However, it is also possible that Decision Tree suffers from underperformance due to its volatile nature.
- The Naïve Bayes model is constructed on the foundation of independence assumption among attributes. Given that the features are intuitively highly correlated (annual income vs how many cars they own vs how big is their house), the independence assumption cannot hold true, and can even be described as severely violated. Therefore, the model is expected to underperform against other models.

The models all appear to have subsatisfactory performance, with accuracy lower than 70%. There are 2 possible reasons for this:

- High correlation among features leads to model confusion.
- Default has an unpredictable nature, since default may arise from highly random events (from credit card issuer perspective) such as sudden lay-offs and shifts in macroeconomic environments. In other words, personal attributes may not be good predictors.

5. Conclusion

We identified credit card defaultee prediction as a very viable data mining problem to work on due to the complexity of data represented by massive numbers of predictors. We have selected 3 different supervised machine learning models: Decision Tree, Logistic Regression and Naïve Bayes classifier. Each model is then tuned to fit to training data while minimizing a custom economic loss estimator function.

Overall, these models do not perform up to our original expectations, with accuracy under 70% and AUC barely over 0.7 in our best model. Several possible explanations have been provided to explain the substandard performance, but we believe the most likely reasons are that the models by nature are not sophisticated enough to tackle this problem, and that the credit card defaultee problem may not be very explainable by personal attributes. Nevertheless, these models can still give insights into the decision boundary of the credit card defaultee prediction problem, potentially uncovering useful insights with a more domain-specific analytical approach when analyzing model parameters.

Since the models are evaluated and validated based on a custom economic loss function, the results generated from using these models already take profit maximization (or rather, loss reduction) into consideration, which means that credit card issuers can directly base their credit card application approval decision on the model results obtained using the optimized model parameters.

There are certain extensions to our current methodology that, within the span of the project, are either too time consuming or beyond the scope of this course. Below details several possible future development directions for this project:

- It is understood that these classifiers in their naïve form and complexity can easily overfit to data. Therefore, more robust forms of these machine learning algorithms, such as ensemble learning, can be utilized to mitigate the overfitting problem of each naïve classifiers. Neural networks can also be considered due to its ability to uncover hidden nonlinear relationships, potentially leading to much better performances, but has the disadvantage of being significantly less explainable, which may result in questionable usage in actual banks and credit card companies.
- Run time is also a point of concern in our analysis process. Due to the enornimity of the dataset, the model training process often takes very long time, typically exceeding one hour for cross-validation or grid search

processes. Thus there exists a trade-off between run time and quality of our models. Given more time, more robust models can be obtained via cross-validation using more folds.

- If given more time, further manipulation can be done in data preprocessing and feature selections, e.g. using an ANOVA table to select significant features, and using the VIF to drop features with multicollinearity problem. This can further increase the explainability of the models as well as better analyze the information contained in each predictor. Applying Principle Component Analysis (PCA) to the dataset can also effectively eliminate multicollinearity in the data input into the model, hence leading to significantly improved performances in models that are by nature strongly impacted by multicollinearity problem.