CREDIT RISK SCORECARD ANALYSIS

NOVEMBER 29, 2021

KO TSZ NGA (VALERIE)
valerie.ktn@gmail.com

Abstract

Customer credit is a personal debt taken on to purchase goods and services. [1] It is a term that is more often used to describe unsecured debt. Credit risk, on the other hand, describes the risk involved when lending the customer goods and services. For instance, the customer may fail to repay the loan or fail to settle payment of unpaid invoices. To mitigate the risk, credit risk scorecard model is used to assess the credit worthiness of customer. Customer risk scorecard is a mathematical classification model to predict the probability of default of customer. In this project, we will focus on using the customer behavioural data gathered to build a predictive classification model. 5 different data mining algorithms were used, and their respective hyperparameters were tuned to compare their performance with relevant evaluation metrics against each other. Variable selection analysis was also carried out. Finally, we will conclude that random forest with hyperparameters tuning gives the best predicting performance for the dataset that we used.

Introduction

Credit risk scorecard is extensively used in banking industry. There are 2 types of scorecards, namely application scorecard and behavioural scorecard. Application scorecard, as the name implies, predicts the probability an applicant will default the payment based on data gathered in application. It helps the bank to make the decision for acceptance or rejection for approval. On the other hand, behavioural scorecard is based on the data gathered throughout or part of the lifetime of the loan or mortgage. The behavioural scorecard predicts the probability of default based on the customer paying habits or other behavioural changes. It aims to calculate the expected loss of the company.

This project focuses on behavioural credit risk scorecard analysis. The detail of the dataset will be described in *Data Source* after *Problem Statement*. Data cleaning will be done, followed by the in-depth *Exploratory Data Analysis*. Due to the time constraint and my limited scope of knowledge as a student, only 5 classification data mining models will be used for comparison in this project. They are 1) Logistics Regression, 2) Support Vector Machine (SVM), 3) Decision tree, 4) Random Forest and 5) XGBoost. The details of these algorithms will be explained in the *Proposed Methodology* section. The output of the analysis will be displayed in *Analysis and Results*. *Conclusion* will be drawn afterwards. *Lessons learnt* will be a subsection that summarises my learning in both the project and the course.

Problem statement

This project is aiming to answer the following questions:

1) What are the important variables for classifying potential customer default?
2) Which classification models can give us the best predictive performance?

Data source

The dataset is a record of customer default payment in Taiwan in October 2005. [2] It consists of 30,000 rows of data. There are 24 variables in total which correspond to 23 dependent variables and 1 independent variable. Below describes the details of the dataset:

Dependent variable (Y): **Default payment next month** (binary)

Independent variables (Xs):

- **LIMIT_BAL**: Amount of the given credit (NT dollar), it includes both the individual consumer credit and his/her family (supplementary) credit.
- **SEX** (1 = male; 2 = female)
- **EDUCATION** (1 = graduate school; 2 = university; 3 = high school; 4 = others).
- **MARRIAGE** (1 = married; 2 = single; 3 = others)
- **AGE** (year).
- **PAY_0, PAY_2 - PAY_6**: History of past payment (-1 = pay duly; 1 = payment delay for one month; 2 = payment delay for two months; . . .; 8 = payment delay for eight months; 9 = payment delay for nine months and above)
- **BILL_AMT1 - BILL_AMT6**: Amount of bill statement (NT dollar) in each month (from Sep, 2005 to Apr, 2005)
- **PAY_AMT1 - PAY_AMT6**: Amount of previous payment (NT dollar) in each month (from Sep, 2005 to Apr, 2005)

Exploratory Data Analysis

A preliminary data inspection was done to check if there is any missing data. No blank input was found so that suggests the absence of missing data. Then, for consistency purpose, the column name of 'PAY_0' was changed to 'PAY_1'. Before visualisation, the data type of each column is reinforced to ensure alignment. For instance, the data type of the 'LIMIT_BAL', the 6 columns of 'BILL_AMT' and the 6 columns of 'PAY_AMT' were changed to float.

To better understand the predicted variable, its distribution is visualised as shown in Figure 1. It is found that 77.9% clients are not expected to default payment whereas 22.1% clients are.

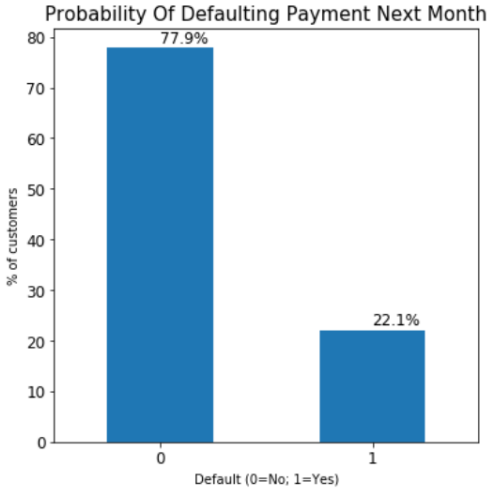


Figure 1: Distribution of response variable

The relationships between the response variable (ie. Default payment next month) and the other explanatory variables are visualised. Figure 2 in Appendix shows that the graph of 'SEX', 'EDUCATION' and 'MARRIAGE' do not show obvious differences in expected default among the respective groups. Therefore, these variables are not classified as important for predicting default preliminarily.

Figure 3 below however shows a more insightful finding. In terms of 'LIMIT_BAL', the number of expected default decreases when limit balance group increases. And for 'AGE', the number of expected default decreases when age group increases. Repayment history 'PAY_0' to 'PAY_1' is displayed and it is observed that the earlier the payment is made, the lesser are the chances of those clients defaulting the payment. 'BILL_AMT1'- 'BILL_AMT6', 'PAY_AMT1'- 'PAY_AMT6' are aggregated and displayed in terms of the total across 6 months respectively. As shown, a more significant amount of default is observed when bill total is in the (5000,50000) basket and pay amount total is in (0,25000) bucket. In short, based on the preliminary data analysis, variables 'LIMIT_BAL', 'AGE', 'PAY_1-PAY_6', 'BILL_AMT1- BILL_AMT6' and 'PAY_AMT1- PAY_AMT6' are potentially more important as their effects on the predicted variable are more profound.

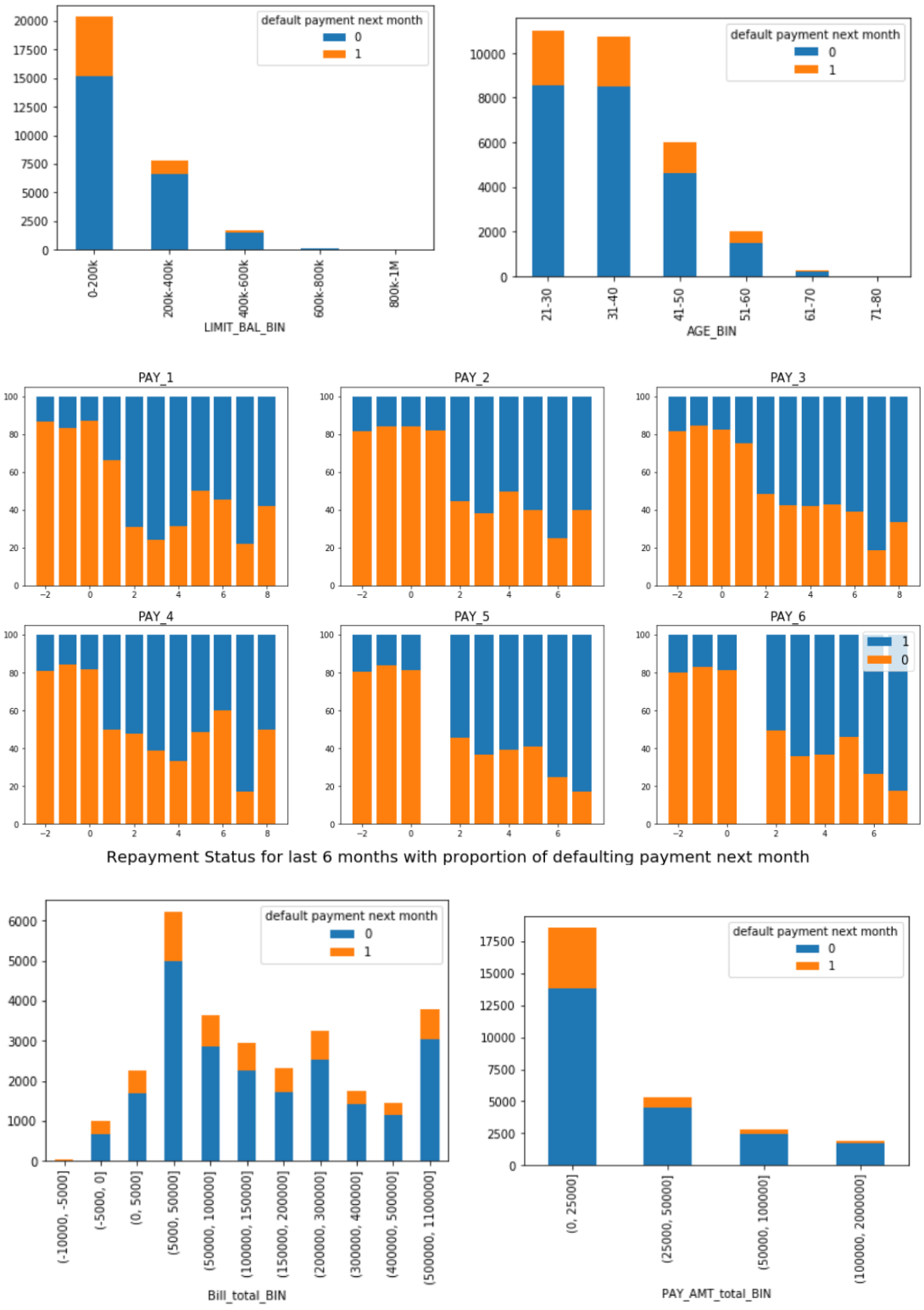


Figure 3: Relationship between response variable and explanatory variables (LIMIT_BAL, AGE, PAY_1-PAY_6, BILL_AMT1- BILL_AMT6, PAY_AMT1- PAY_AMT6)

As there are 24 variables in total, the correlation matrix of all variables would not be informative enough. Instead, the top 10 independent variables (that are mostly correlated to the predicting variable) are selected to be displayed in Figure 4 below.

It is shown that the repayment history (ie. 'PAY_1' to 'PAY_6') are highly correlated to the predicting variable. Yet, multicollinearity exists among them. Besides, contrary to the previous finding, 'EDUCATION' is listed to be one of the highly correlated explanatory variables with the response. Further analysis on that will be carried out in the variable selection section.

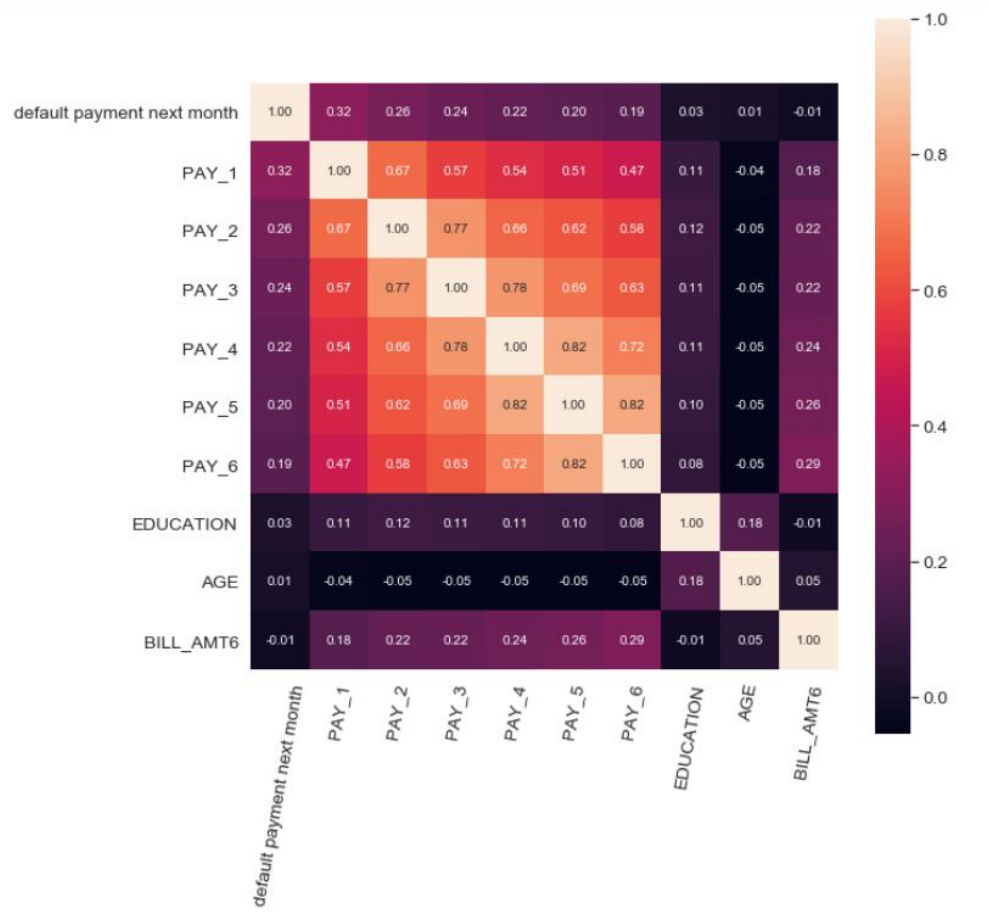


Figure 4: Correlation Matrix

Proposed Methodology

5 data mining classification models will be covered.

1. **Logistics regression**

It is an algorithm for binary classification problem. Unlike linear regression, the output range of logistics regression is bounded between 0 and 1. The conditional probability of the dependent variable is modelled as a logit-transformed multiple linear regression of the explanatory variables [3]. This method assumes error independence, linearity in the logit for continuous variables, absence of multicollinearity and lack of strongly influential outliers [4]. Logistics regression is used here as a baseline method due to its simplistic nature.

Data is scaled before training to speed up the optimisation process. 10-fold cross validation (model 1.2) will be used on the training dataset to compare with the predicting performance of the original model (model 1.1). Due to the lack of stepwise function available in Python Scikit learn library, the number of important variables is arbitrarily defined to be 10. Recursive feature elimination (RFE) function in sklearn.features package is used to select the top 10 most important variables as listed in Table 1 in Part 1 of *Analysis and Results*. A logistic regression model of the selected variables (model 1.3) is trained as well.

2. **Support Vector Machine (SVM)**

Unlike Logistics regression, SVM can be used in both regression and classification. It is another algorithm that is very popular due to its significant accuracy with less computation power [5]. This method aims to find a hyperplane in an N-dimensional space, in which N equals to the number of features, to distinctly classifies the data points. The plane that has the maximum margin distance between data points of both classes is chosen. Maximizing the margin distance helps to provide reinforcement so that future data points can be classified with more confidence [5]. No assumption on the dataset is required in SVM.

Data is again scaled before training to speed up the optimisation process. 10-fold cross validation (model 2.2) will be used on the training dataset to compare

with the predicting performance of the original model (model 2.1). As 'rbf' kernel is used, our SVM model is non-parametric. Variable selection is therefore inapplicable here. Instead, hyperparameters are adjusted as listed in Table 2 and the best parameters are selected to train a new SVM model (model 2.3).

Table 2: Hyperparameters selection list and the best parameters selected in SVM

Hyperparameters	Selection list	Best value selected
Kernel	rbf	**rbf**
Penalty parameter of the error term (C)	0.1, 1, 10	**1**

3. Decision tree

It is a method that can be used in both regression and classification. Classification tree is composed of branches that represent attributes, while the leaves represent decisions. The decision process starts at the trunk and follows the branches until a leaf is reached. A greedy algorithm is used for the selection of input variable and specific split or cut point to minimize a cost function. Tree construction ends using a predefined stopping criterion, such as a minimum count on the number of training instances assigned to each leaf node of the tree [6].

10-fold cross validation (model 3.2) will be used on the training dataset. It will then be used to compare the predicting performance of the original model (model 3.1). Important features are ranked and displayed in Figure 6 in Part 1 of *Analysis and Results*. The decision tree will be pruned by an optimal alpha selected (ie. 0.001) as depicted by Figure 5 below.

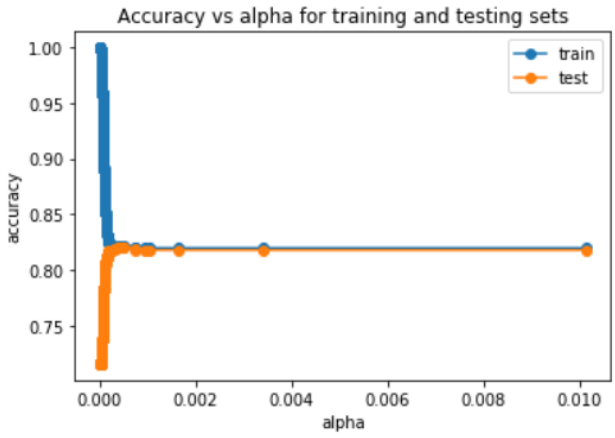


Figure 5: Accuracy plot with alpha

To further check if the pruned tree model (model 3.3) gives the best predicting performance, hyperparameters are selected from a list of value and the best combination is used to train a new decision tree model (model 3.4). Table 3 below shows the detail for the hyperparameters selection.

Table 3: Hyperparameters selection list and the best parameters selected in Decision Tree

Hyperparameters	Selection list	Best value selected
Maximum depth	2 to 20	**6**
Minimum samples leaf	1 to 20	**19**
Minimum sample split	2 to 40	**5**
Information gain	'Gini', 'entropy'	**'entropy'**

4. **Random forest**

Alike decision tree and SVM, it is a method that can be used in both regression and classification. It is classified as an ensemble method that consist of many individual decision trees. Random forest is an improved version of decision tree as it can reduce the overfitting of datasets and increases precision [7]. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. The correlation between the individual tree models is low so that guarantees the predictions and errors made are less likely to be correlated [8]. Bagging and feature randomness are used to achieve that.

A random forest model with default parameter values is built (model 4.1). Then, the best combination of hyperparameters is chosen with the 'RandomizedSearchCV' function in Scikit learn. The detail of the hyperparameter selection is displayed in Table 4. A new random forest model (model 4.2) is trained based on the selected hyperparameters. Important features are chosen with the tuned model and are displayed in Figure 7 in Part 1 of *Analysis and Results*.

Table 4: Hyperparameters selection list and the best parameters selected in Random Forest

Hyperparameters	Selection list	Best value selected
No. of estimator	50 to 200	**177**
Maximum features	1 to 24	**7**
Maximum depth	2 to 10	**9**
Minimum samples leaf	1 to 20	**9**
Minimum sample split	2 to 40	**2**
Information gain	'Gini', 'entropy'	**'entropy'**

5. **XGBoost**

It is a method that can be used in both regression and classification as well. XGBoost is another ensemble method that implements gradient boosted decision tree with improved speed and performance. Gradient boosting is an approach where new models are created to predict the residuals or errors of prior models and then added together to make the final prediction [9]. Gradient descent algorithm is used to minimize the loss when adding new models. XGBoost improves upon the gradient boosting framework through systems optimization and algorithmic enhancements [10]. In terms of system optimisation, it is done by parallelization, tree pruning and hardware optimisation. And for algorithmic enhancements, it is done by regularization, sparsity awareness, weighted quantile sketch and cross-validation.

A XGBoost model with default parameter values is built (model 5.1). Then, the best combination of hyperparameters is chosen with the 'RandomizedSearchCV' function in Scikit learn. The detail of the hyperparameter selection is displayed in Table 5. A new XGBoost model is built with the selected hyperparameters (model 5.2). Important variables are ranked based on the tuned model as displayed in Figure 8 in Part 1 of next section.

Table 5: Hyperparameters selection list and the best parameters selected in XGBoost

Hyperparameters	Selection list	Best value selected
Gamma	1 to 5	**5**
Subsample	0.8 to 1	**0.8**
Colsample_bytree	0.8 to 1	**0.8**
Minimum depth	5 to 7	**5**

No. of estimators	100 to 300	**100**

80% of data will be randomly selected for model training and the remaining 20% will be used for testing.

Analysis and Results

To evaluate model performance, I am going to use i) Testing accuracy, ii) Area under the roc curve (AUC) and iii) Sensitivity score as evaluation measures.

i) **Testing accuracy:** It is calculated by the 'accuracy_score' function in the Python Scikit learn metrics package. With the trained model, prediction is made with the testing set and the predicted y value is compared with the true y value in test set. A higher testing error indicates a better performance. The equation is displayed as follows:

$$Testing\ accuracy = \frac{No.\ of\ points\ classified\ correctly}{Total\ number\ of\ data\ points\ in\ test\ set}$$

ii) **Area under the roc curve (AUC):** The value measures the degree of separability. ROC is a curve that is plotted with True Positive Rate (TPR) against False Positive Rate (FPR). It represents how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting the classes correctly. The optimal value for AUC is 1.

iii) **Sensitivity score:** It is also called recall (REC) or true positive rate (TPR). It is the ratio of true positives to total (actual) positives in the data. Sensitivity is chosen instead of specificity, which indicates the ratio of true negatives to total (actual) negatives, is because the cost of false negatives outweighs that of false alarm in our case. The financial loss of having a default customer is hard to bear. Alike the other 2 mentioned metrics, sensitivity is again the higher the better. The equation is displayed as follows:

$$Sensitivity = \frac{No.\ of\ True\ Positive}{No.\ of\ True\ Positive + No.\ of\ False\ Negative}$$

Part 1: Variable selection

For **Logistic regression**, the top 10 most important variables are selected and listed in Table 1 below.

Table 1: Top 10 most important variables selected by RFE of Logistic Regression model

LIMIT_BAL	AGE	PAY_1	PAY_2	PAY_3
BILL_AMT1	BILL_AMT2	BILL_AMT3	PAY_AMT1	PAY_AMT2

Variable selection is not performed for **SVM**. As mentioned in *Proposed Methodology*, we used a non-linear kernel 'rbf' for training SVM, therefore, variable selection is not applicable here due to its non-parametric nature.

For **Decision tree**, the feature importance is ranked and displayed by Figure 6 below. 'PAY_1' is shown to outperform the other variables.

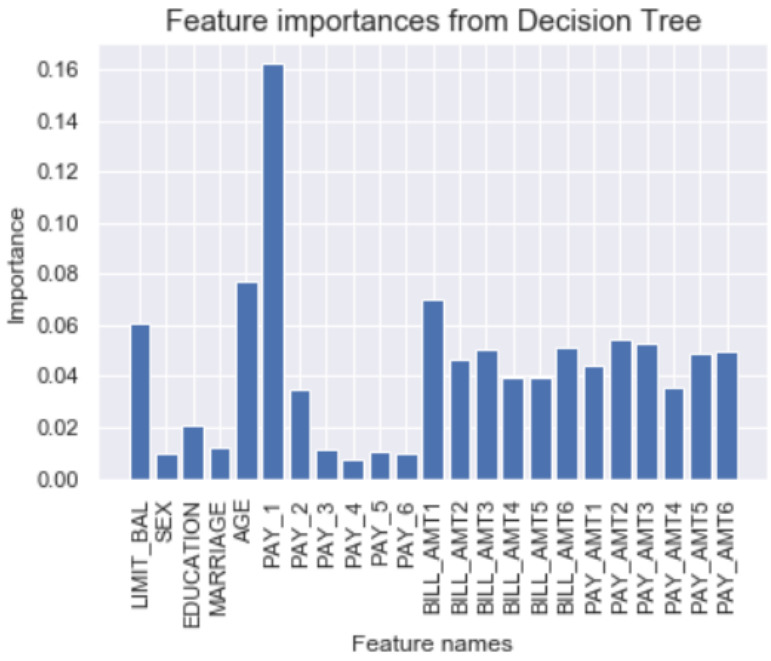


Figure 6: Feature Importance with Decision Tree model

For **random forest**, the feature importance is chosen by the tuned model and is ranked as shown in Figure 7 below. Again, 'PAY_1' outperforms the other variables.

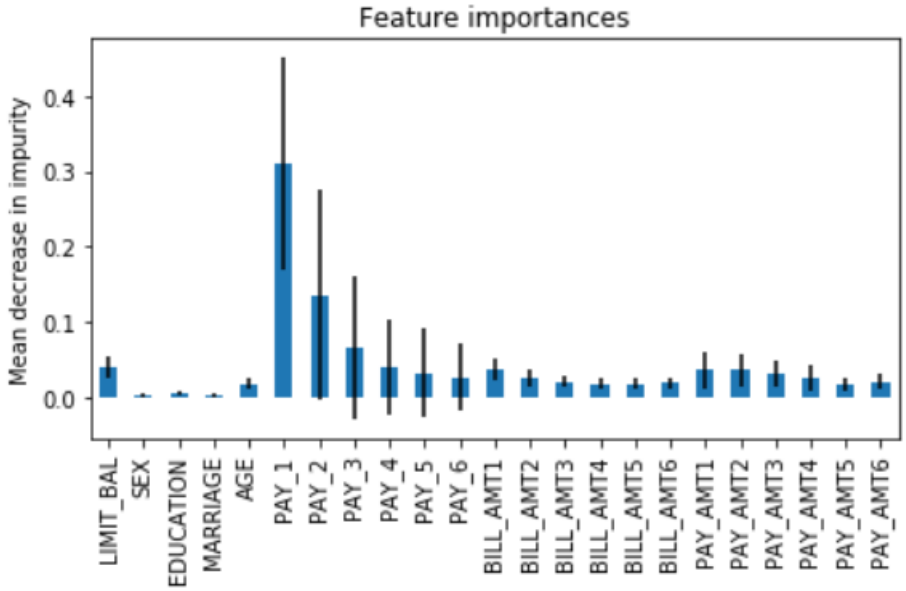


Figure 7: Feature Importance with tuned Random Forest model

Last but not least, feature importance result for **XGBoost** is visualised in Figure 8 below. This time, 'LIMIT_BAL' is found to be the most important.

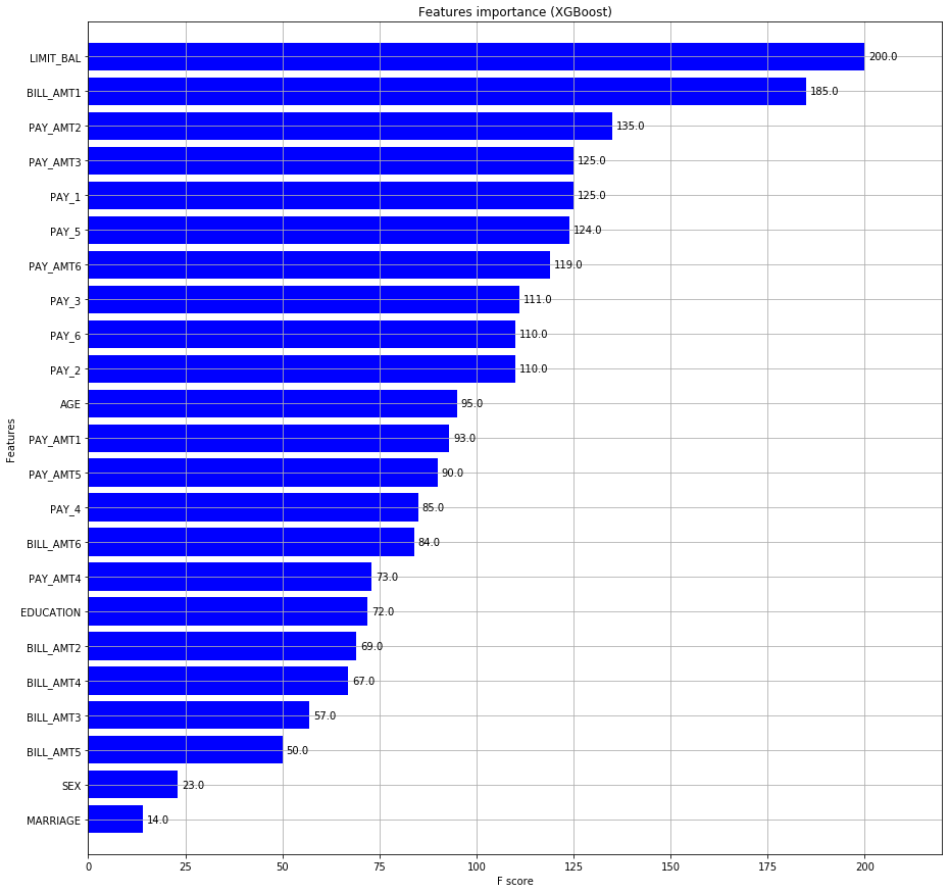
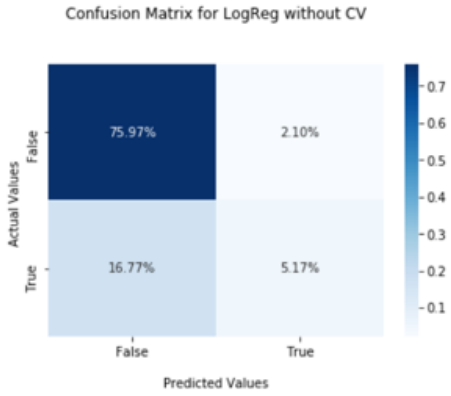


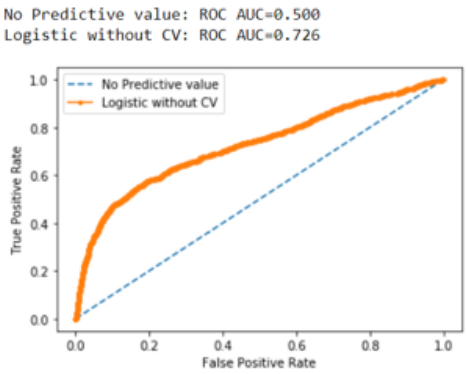
Figure 8: Feature Importance with tuned XGBoost model

Part 2: Models prediction performance

1. Logistics regression

3 models are built in total. Model 1.1 is built without cross validation; model 1.2 is built with 10-fold cross validation; model 1.3 is built with the top 10 important variables. Their result is displayed in term of confusion matrix and ROC curve in Figure 9. Testing accuracy of model 1.1 is the same with model 1.2. That suggests the original model is robust in terms of prediction. However, the testing accuracy, AUC and sensitivity of the model 1.3 is the worst compared with the other two. That suggest the model only based on top 10 most important variables has a poor predicting power.

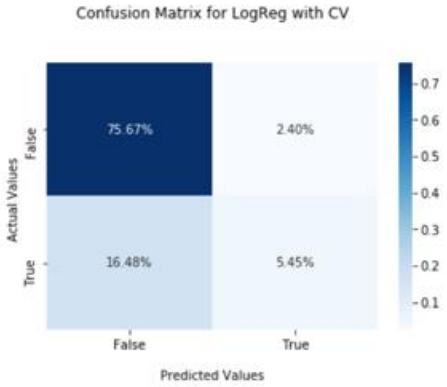


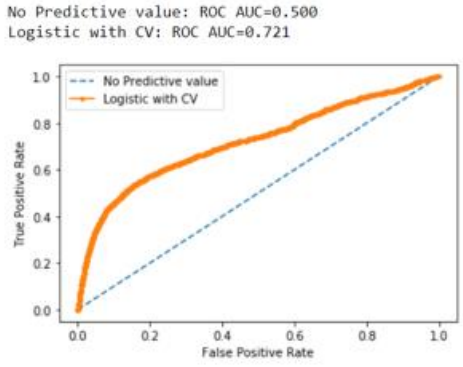


Sensitivity: 0.236

Testing accuracy: 0.811

Model 1.1

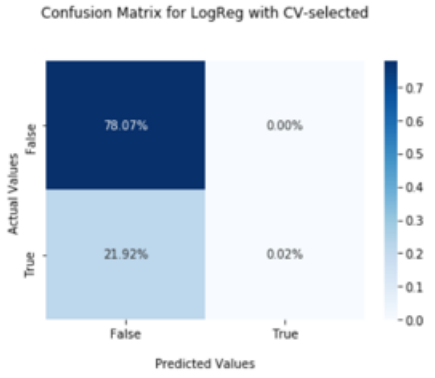


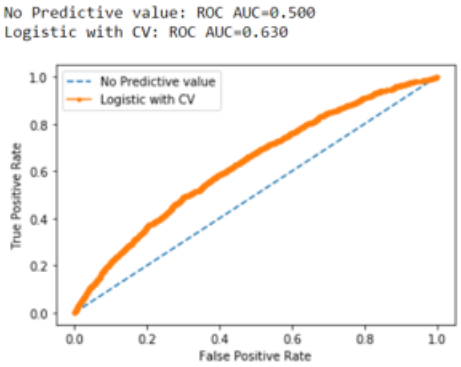


Sensitivity: 0.249

Testing accuracy: 0.811

Model 1.2





Sensitivity: 0.001

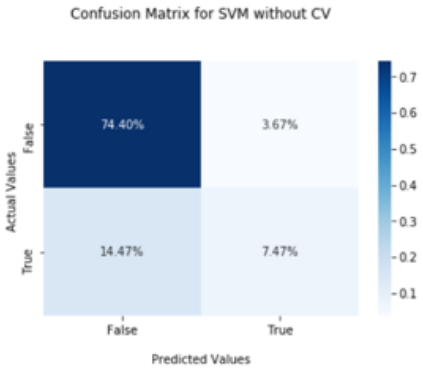
Testing accuracy: 0.781

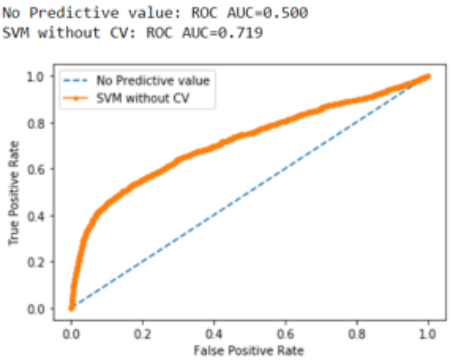
Model 1.3

Figure 9: Summary statistics for Logistic Regression

2. Support Vector Machine (SVM)

Model 2.1 is built without cross validation; model 2.2 is built with 10-fold cross validation; model 2.3 is built with hyperparameters tuned with the best selected value. Their result is displayed in term of confusion matrix and ROC curve in Figure 10. The summary statistics of model 2.1 and model 2.3 are the same. Both gives the best result among the three models. That is because the best hyperparameters selected are found to be the same as the original model with the default setting.

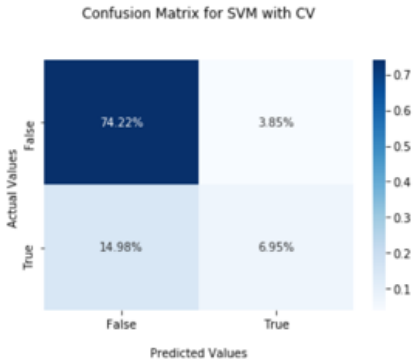


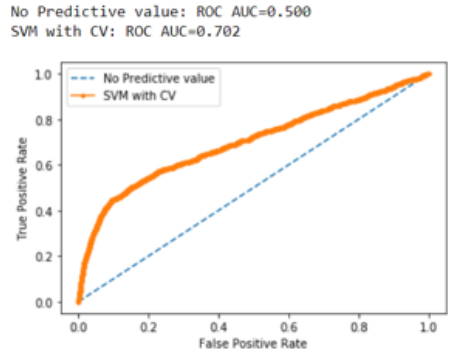


Sensitivity: 0.340

Testing accuracy: 0.819

Model 2.1

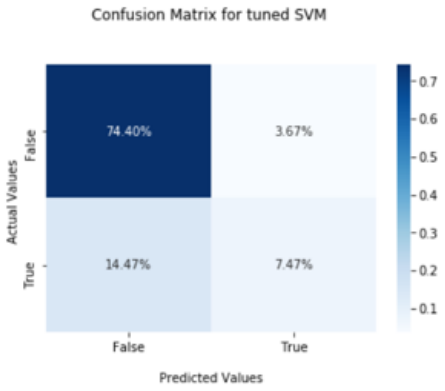


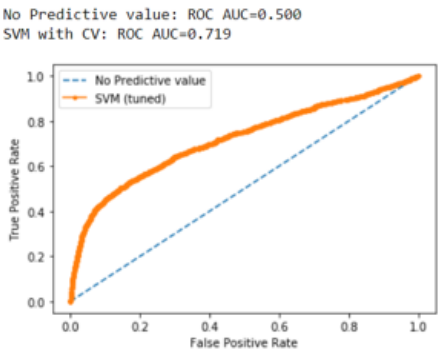


Sensitivity: 0.317

Testing accuracy: 0.812

Model 2.2





Sensitivity: 0.34

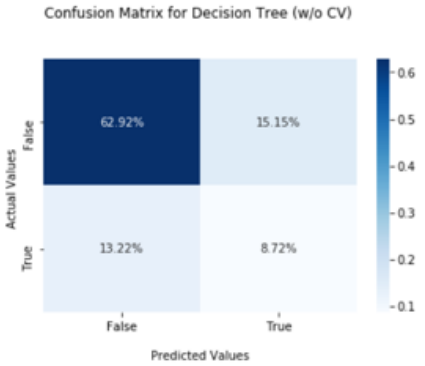
Testing accuracy: 0.819

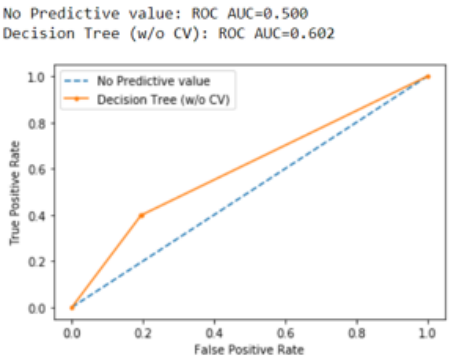
Model 2.3

Figure 10: Summary statistics for Support Vector Machine (SVM)

3. Decision Tree

4 models are built this time. Model 3.1 is built without cross validation; model 3.2 is built with 10-fold cross validation; tree pruning is performed to build model 3.3; hyperparameters are adjusted to build model 3.4. Their result is displayed in term of confusion matrix and ROC curve in Figure 11. In terms of sensitivity, model 3.2 performs the best. One thing to note here is that the testing accuracy of model 3.3 and model 3.4 are the same, both reaches 0.818, the best score among the 4 decision tree models. Yet, the AUC of model 3.4 is slightly higher, suggesting it performs better in general.

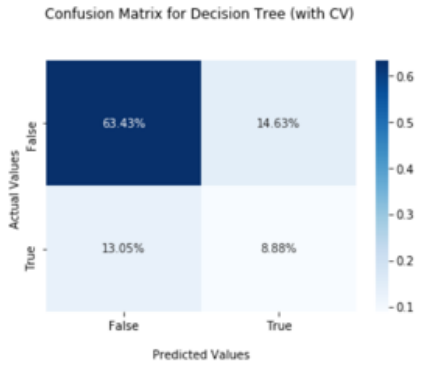


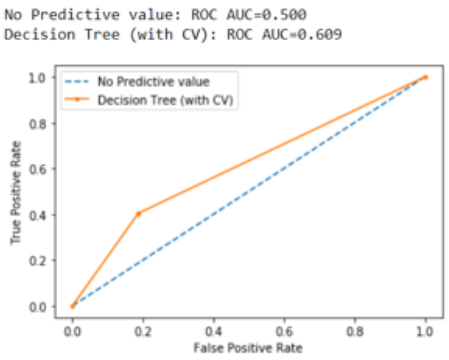


Sensitivity: 0.397

Testing accuracy: 0.716

Model 3.1

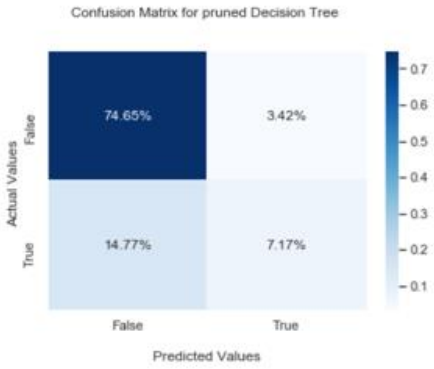


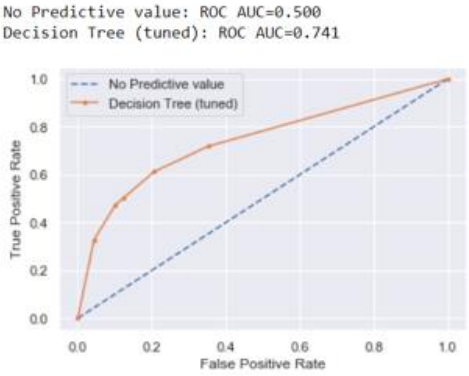


Sensitivity: 0.405

Testing accuracy: 0.723

Model 3.2

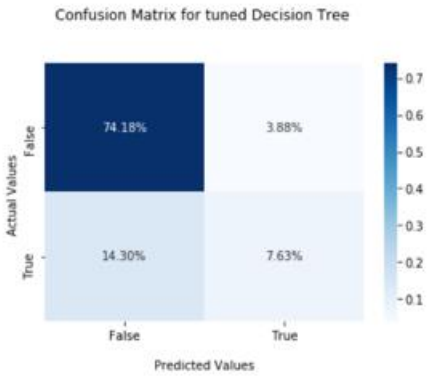


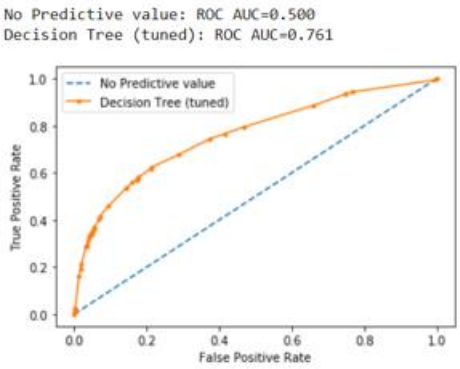


Sensitivity: 0.327

Testing accuracy: 0.818

Model 3.3





Sensitivity: 0.348

Testing accuracy: 0.818

Model 3.4

Figure 11: Summary statistics for Decision Tree

4. Random Forest

Model 4.1 is built with default setting; model 4.2 is built with hyperparameters adjusted based on the best value found as listed in Table 4 in *Proposed Methodology*. Their result is displayed in term of confusion matrix and ROC curve in Figure 12. Despite a slight drop in sensitivity, the tuned model performs better in term of AUC and testing accuracy. Therefore, in short, improvement is made with hyperparameter tuning.

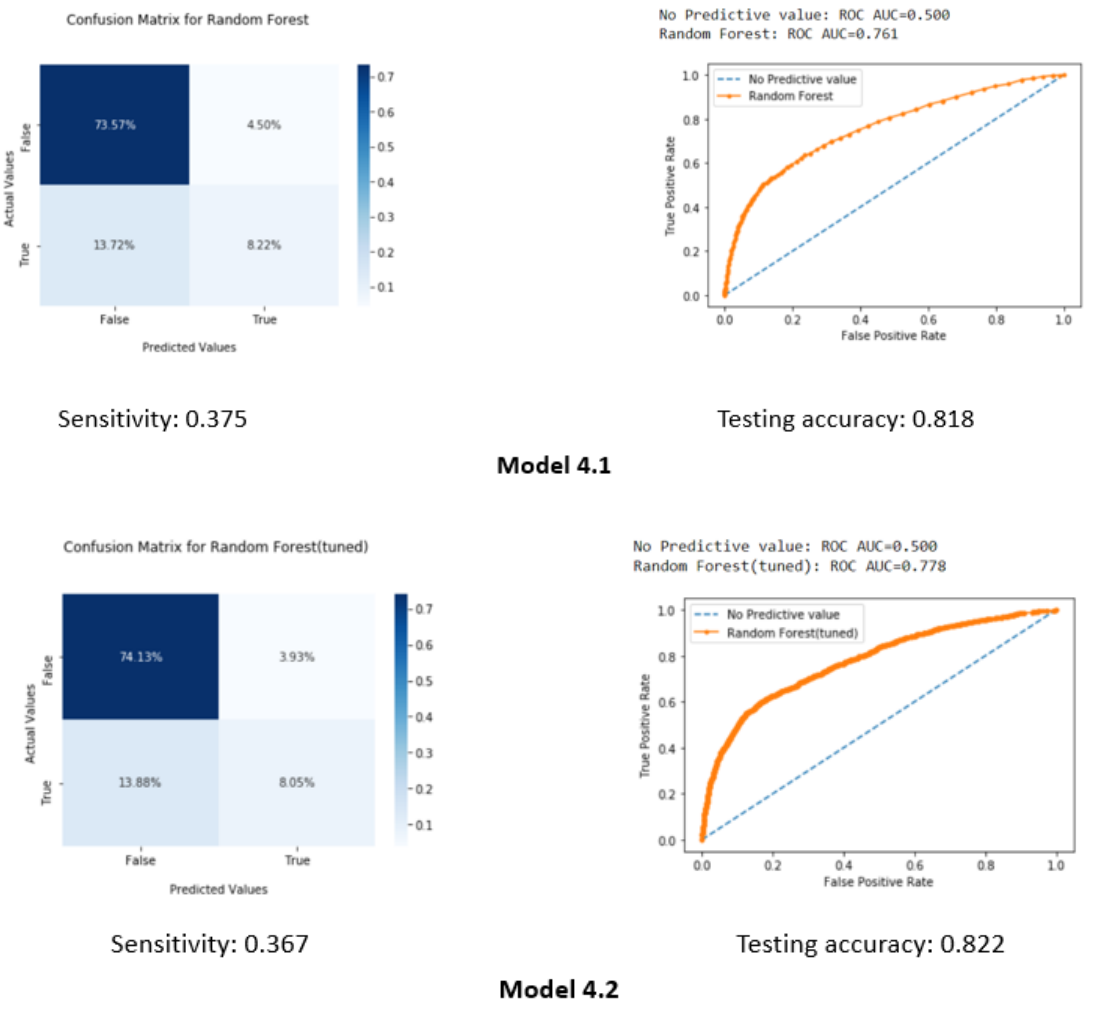
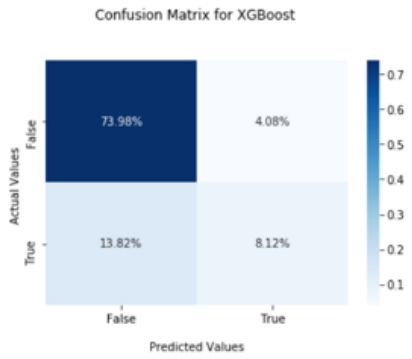
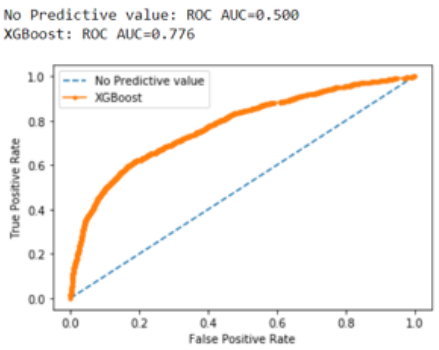


Figure 12: Summary statistics for Random Forest

5. XGBoost

Model 5.1 is built with Scikit learn xgboost default setting; model 5.2 is built with hyperparameters tuning as of Table 5 in *Proposed Methodology*. The testing accuracy stays the same while the sensitivity and AUC drop. That suggests hyperparameter tuning does not help to improve model performance.

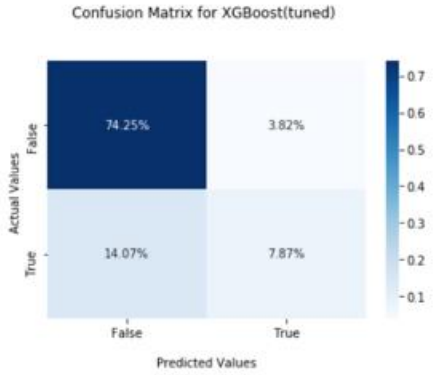


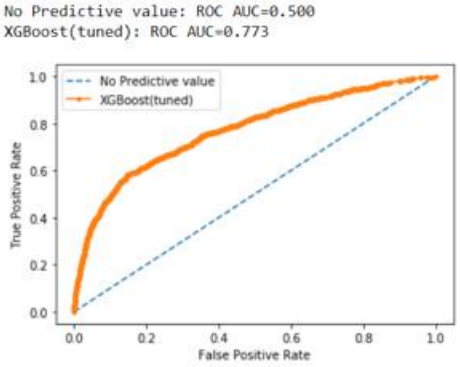


Sensitivity: 0.37

Testing accuracy: 0.821

Model 5.1





Sensitivity: 0.359

Testing accuracy: 0.821

Model 5.2

Figure 13: Summary statistics for XGBoost

Conclusions

The variable selection suggested by different models is listed in Table 6 below. Only the top 3 variables are displayed here due to space constraint. 'PAY_1' and 'BILL_AMT1' appear 3 times respectively. That suggests they are potential highly related to predict default of customer. This result agrees with our *Exploratory Data Analysis*. **'PAY_1'** and **'BILL_AMT1'** suggest that the most recent month's payment has the biggest impact on predicting customer default next month.

Table 6: Summary of variable selection

The model comparison is displayed in Table 7 below. Model 3.2, which is the Decision tree with 10-fold cross validation achieves the highest sensitivity. This means the model has the lowest chance of having false negative which is crucial to the business. However, it gives a relatively poor performance in terms of testing accuracy and AUC.

Instead, the **random forest model after hyperparameter tuning (ie. model 4.2)** gives the best result in these 2 metrics. High testing accuracy and high AUC imply the model has the highest predictability and degree of separability respectively. This result varies with my hypothesis as initially I thought XGBoost would outperform the other models due to its renowned robustness and performance. That said, the predictability of tuned XGBoost (ie. model 5.2) is just slightly poorer than the model 4.2, suggesting it is a well-performed model as well.

In short, model 3.2 has the best sensitivity but in general, model 4.2 performs the best. Yet, I believe the performance of the optimal model can be further improved by a more extensive hyperparameter search. Outlier analysis is also suggested to be done to ensure the quality of data which can help to build a more robust predictive model.

	Most important	2nd important	3rd important
1) Logistics regression	PAY_1	BILL_AMT1	PAY_AMT2
2) SVM		*NA*	
3) Decision Tree	AGE	PAY_1	BILL_AMT1
4) Random Forest	PAY_1	PAY_2	PAY_3
5) XGBoost	LIMIT_BAL	BILL_AMT1	PAY_AMT2

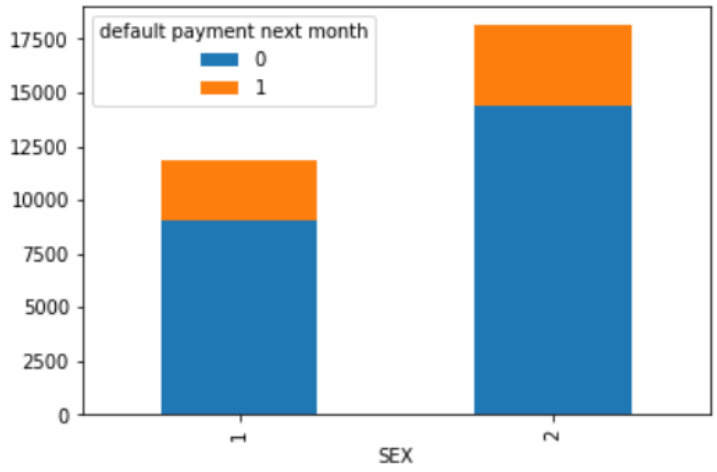
Model		**Testing accuracy**	**Area under the curve (AUC)**	**Sensitivity**
1) Logistics regression	**1.1.** Without CV	0.811	0.726	0.236
	1.2. With 10-fold CV	0.811	0.721	0.249
	1.3. With Top 10 variables	0.781	0.63	0.001
2) SVM	**2.1.** Without CV	0.819	0.719	0.340
	2.2. With 10-fold CV	0.812	0.702	0.317

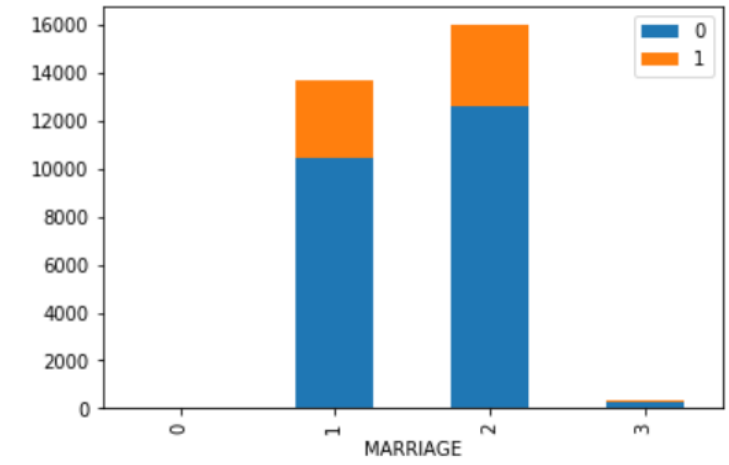
	2.3. After hyperparameter tuning	0.819	0.719	0.340
3) Decision Tree	**3.1.** Without CV	0.716	0.602	0.397
	3.2. With 10-fold CV	0.723	0.609	**0.405**
	3.3. Tree pruning	0.818	0.741	0.327
	3.4. After hyperparameter tuning	0.818	0.761	0.348
4) Random Forest	**4.1.** Original	0.818	0.761	0.375
	4.2. After hyperparameter tuning	**0.822**	**0.778**	0.367
5) XGBoost	**5.1.** Original	0.821	0.776	0.370
	5.2. After hyperparameter tuning	0.821	0.773	0.359

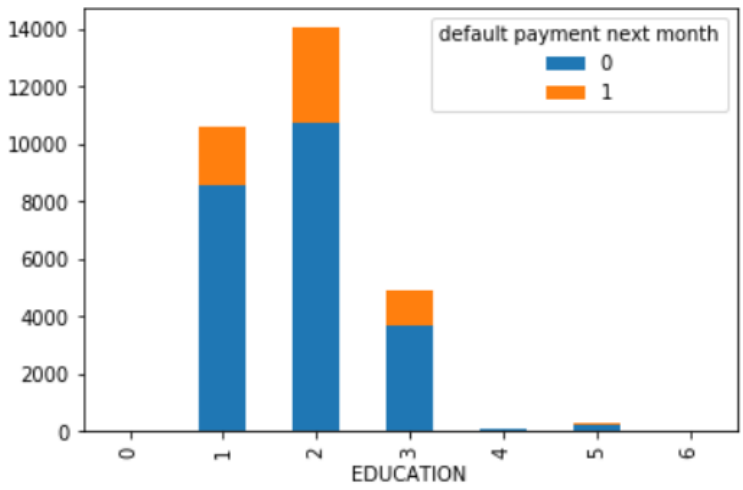
Table 7: Summary statistics for model performance

Appendix

Figure 2: Relationship between response variable and explanatory variables (SEX, EDUCATION, MARRIAGE)







Reference

[1] J.Kagan (2021, May). "Credit Card Debt". *Investopia* [online]

Available: https://www.investopedia.com/terms/c/consumercredit.asp

[2] I.C Yeh (2016, Jan). "default of credit card clients Data Set". *UCI Machine Learning Repository* [online].

Available: http://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients#

[3] P.Caie, N.Dimitriou and O.Arandjelovic (2020, Jun). "Chapter 8 – Precision medicine in digital pathology via image analysis and machine learning ". *Artificial Intelligence and Deep Learning in Pathology. pp. 149-173*

[4] J.C. Stoltzfus (2011, Oct). "Logistics regression: a brief primer". PMID: 21996075 DOI: 10.1111/j.1553-2712.2011.01185.x

[5] R.Gandhi (2018, Jun). "Support Vector Machine – Introduction to Machine Learning Algorithms". *towards data science* [online].

Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[6] J.Brownlee (2016, Apr). “Classification and Regression Tress for Machine Learning” *Machine Learning Mastery* [online]

Available: https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/

[7] O.Mbaabu (2020, Dec). “Introduction to Random Forest in Machine Learning” *Section.io* [online] Available: https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/

[8] T.Yiu (2019, Jun). “Understanding Random Forest” *towards data science* [online]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[9] J.Brownlee (2016, Aug). “A Gentle Introduction to XGBoost for Applied Machine Learning” *Machine Learning Mastery* [online]

Available: https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

[10] V.Morde (2019, Apr). “XGBoost Algorithm: Long May She Reign!” *towards data science* [online]. Available:https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d