STAT5003_14_Credit_Card_Analysis

STAT5003 group project 14

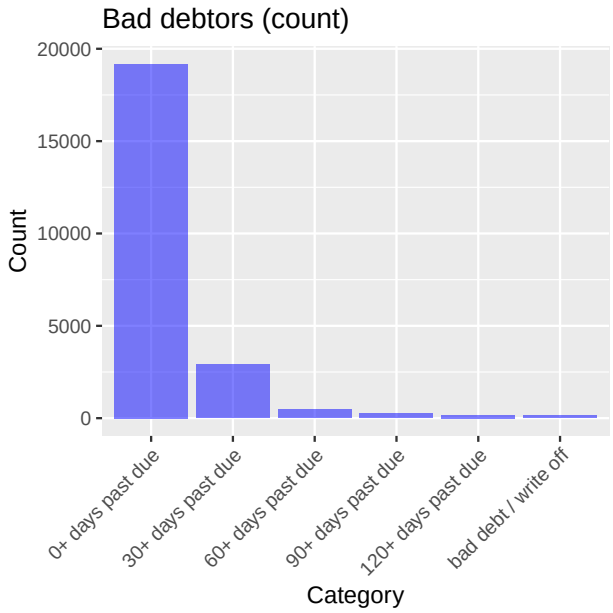
2023-10-23

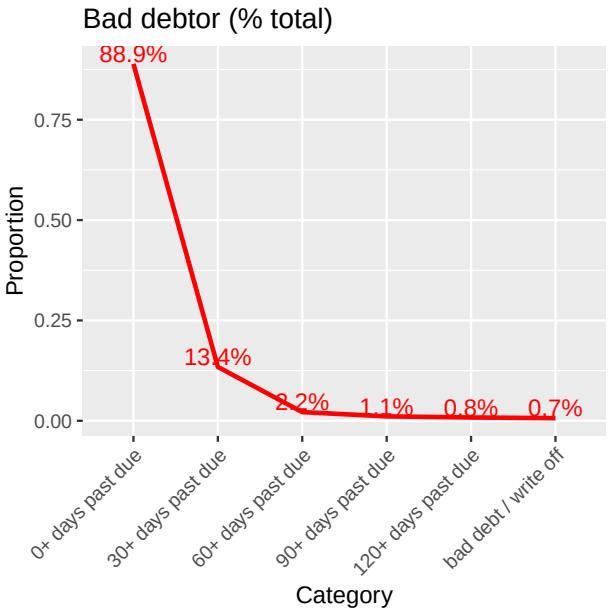
OVERVIEW

- **Problem statement:** Commercial banks need to continually observe the impact of customer's work and life stability on credit card defaults, which is costly to maintain (Li et. al., 2019). [Traditional] regression functions on demographic variables and credit card features yield limited explanatory power compared to other factors such as attitude variables and personality variables (Wang et. al, 2011). Previous attempts at applying machine learning to classify card applicants into different credit limits has achieved acceptable results (82% predictive accuracy of credit card defaulters with neural networkst) (Leong et. all., 2019).
- **Goal:** The goal of this exercise is to apply machine learning methods to predict potential credit defaulters based on a set of data available at point of application. The dataset can be accessed here.
- **The dataset is split into two files**: **credit_record** (3 features x 1+ million observations) which tracks customer credit performance overtime and **application_record** (18 features x 438+ thousand observations) which contains collected information at the initial credit card application such as ID, gender, occupation, number of children, marital status, etc. The total data points Both files are connected by the feature **ID**.

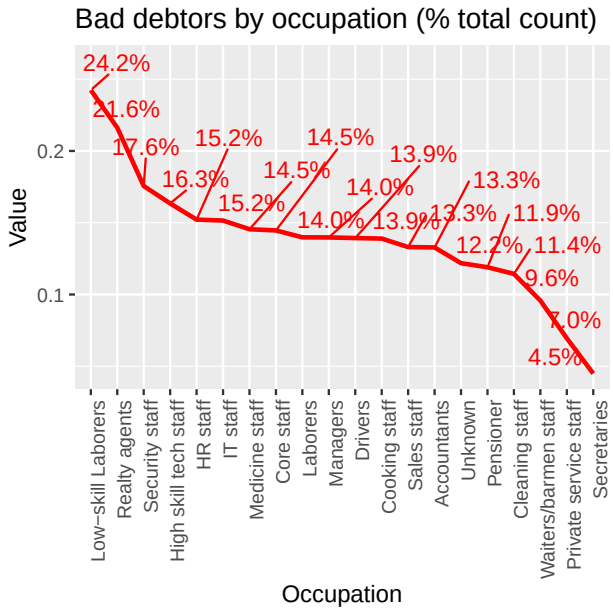
FEATURE ENGINEERING

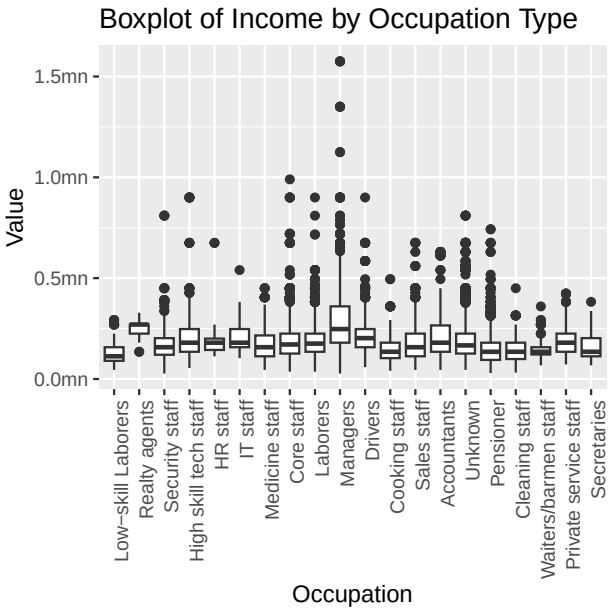
- **Data analysis and cleaning:** the dataset was screened for nulls, blanks, and duplicates and further trimmed to only include IDs with 20+ months banking history. As displayed below, 88.9% of debtors has more than 0 days of delinquency which could very well be an honest mistake and not representative of their ability to service their payments. Thus we defined **30+ days of delinquency as our target feature, denoted as class 1. The resulting dataset contained 21,575 customers of which 2,895 customers (c.13%) are in class 1**.



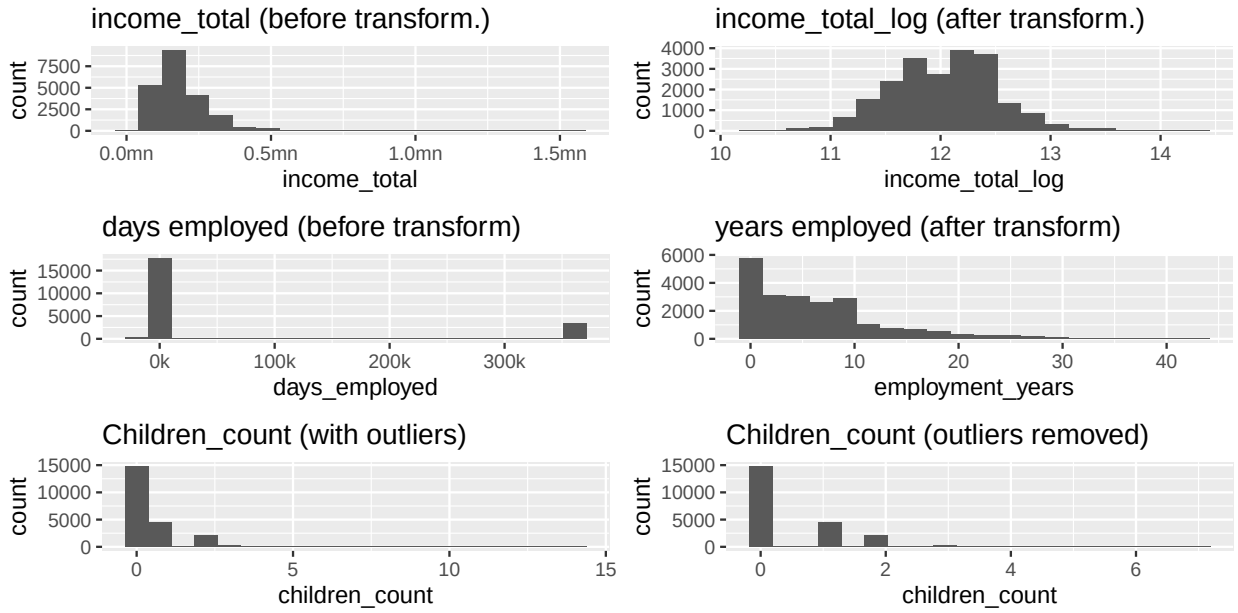


- Delinquency rate by occupation type shown below. We can see that delinquency rate does not necessarily increase with lower wages and some occupation has large variability in total income.

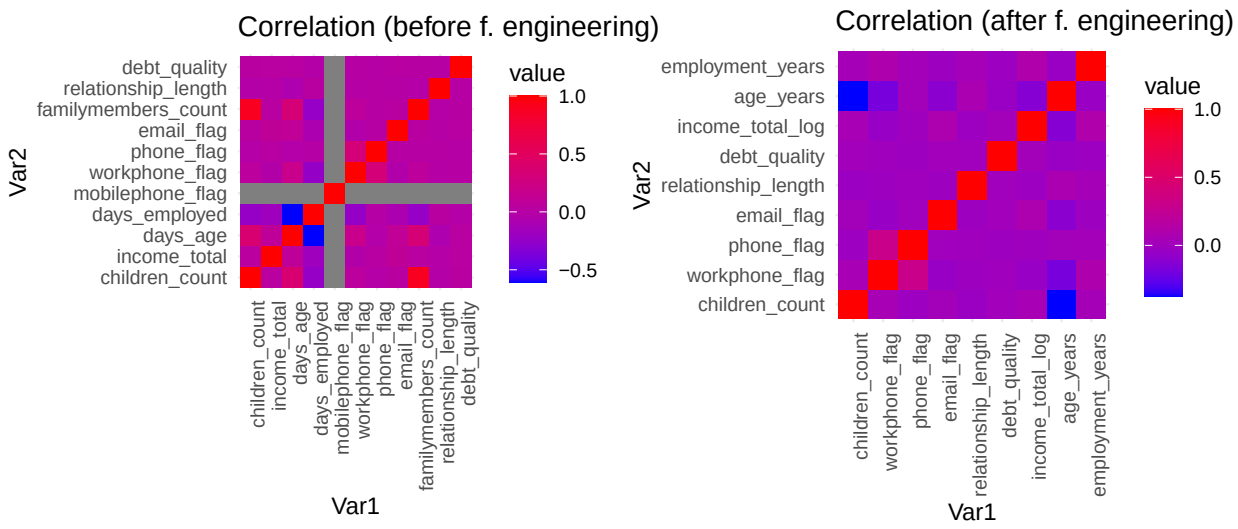




- Selected feature distributions shown below to show the impact of data cleaning/transformation.



- Correlation plots shown below to show the impact of data cleaning/transformation. Improvement in correlation between features can be seen by fewer red boxes on the right.



PRE-PROCESSING

Pre-processing used for this analysis are described below:

- **Min-max scaling:** applied to numerical data (excluding the target) to remove bias from large values.
- **One-hot-encode:** applied to categorical data (excluding the target) to create binary representation of each categorical value. This is done as some machine learning models require numerical input.

- **Synthetic Minority Over-sampling Technique (SMOTE):** applied to the training set to prevent bias towards majority class due to data imbalance. SMOTE works by creating synthetic data points between randomly-chosen k-nearest neighbor at random distances in the feature space. In practice, SMOTE should only be used on training data to prevent leakage.
- **Up-sampling:** applied to the training set to address significant imbalances between classes. upSample operates by increasing the number of instances of the minority class through duplication of random instances.
- **Principal Component Analysis (PCA)**: applied to the training set to reduce data dimensionality used in Support Vector Machine (SVM). As discussed below, SVM works by building a decision boundary/hyperplane that best separates features. This process can be overly complicated on overly complex dataset.

CLASSIFICATION METHODS

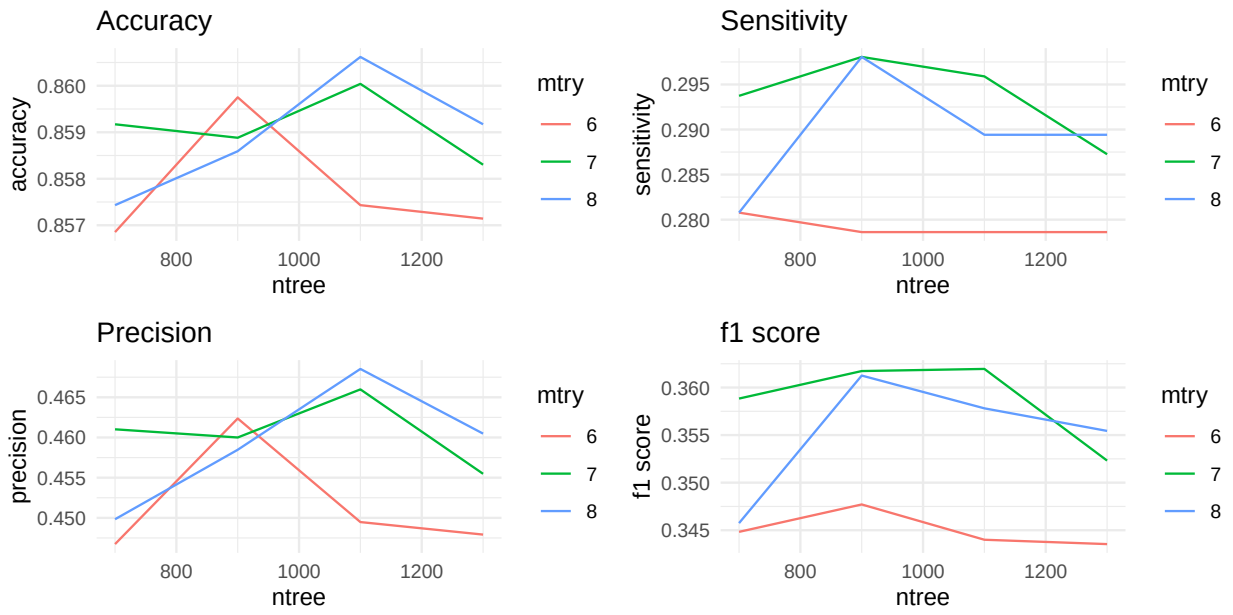
Five algorithms are used in this exercise and discussed below. Please see results on testing scores in the 'RESULTS' section.

1. Random Forest

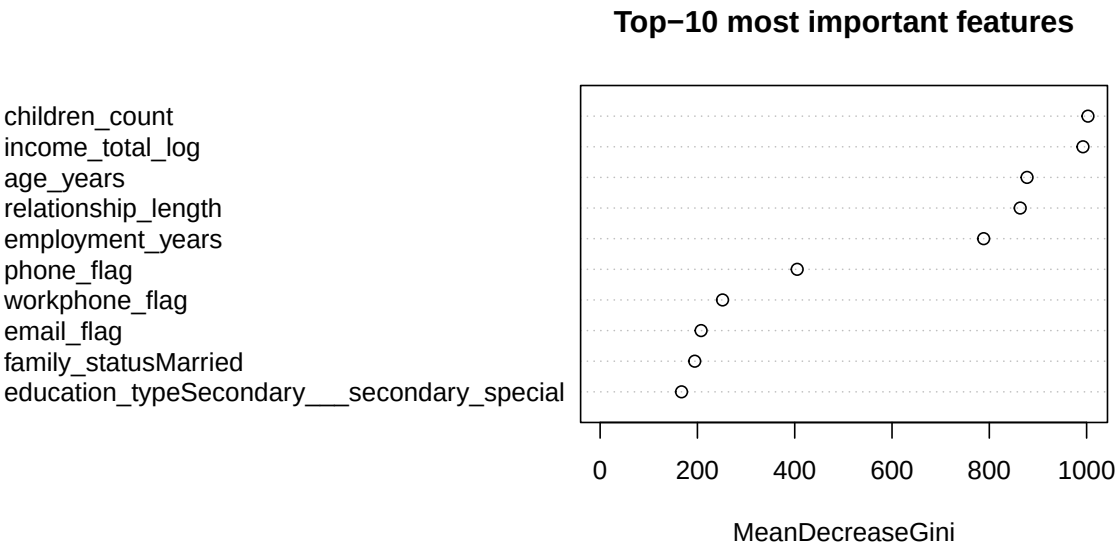
- **Description:** Random forest is an ensemble method which combines multiple decision trees to create predictions. Random forest is known for its robustness against overfitting as it takes random features in each of the individual decision trees and takes an average/vote of these trees for a combined result.
- **Design:** (i) Data pre-processing: min-max scaling, one-hot-encoding and SMOTE were applied to the dataset. (ii) Hyperparameter tuning: grid search using train and validation sets were applied on **'ntree'** (number of trees) and **'mtry'** (number of features considered for splitting) to identify the model with the best **f1-score**. (iii) Rationale for metric selection: f1-score gives a balance between precision (minimizing false positives) and recall (minimizing false negatives). (iv) Key feature identification: the top-10 most important features were identified to understand the drivers of bad debt.
- **The best model parameters by f1-score shown below:**

```
## Best model parameters (maximizing f1 score):
## - ntree:  1100
## - mtry:  7
##
## Training scores for the best model:
## - accuracy:  0.8600406
## - precision:  0.4659864
## - sensitivity:  0.2958963
## - specificity:  0.9474565
## - balanced accuracy:  0.6216764
## - f1 score:  0.3619551
```

- Low sensitivity, precision, and f1-score implies the model has high proportion of false negatives and false positives and is unreliable. Overall grid search results shown below.



- Key features: the top-5 features, which are children_count, income_total_log, age_years, relationship_length, and employment_years, show significant gini decrease values.



2. Logistic Regression

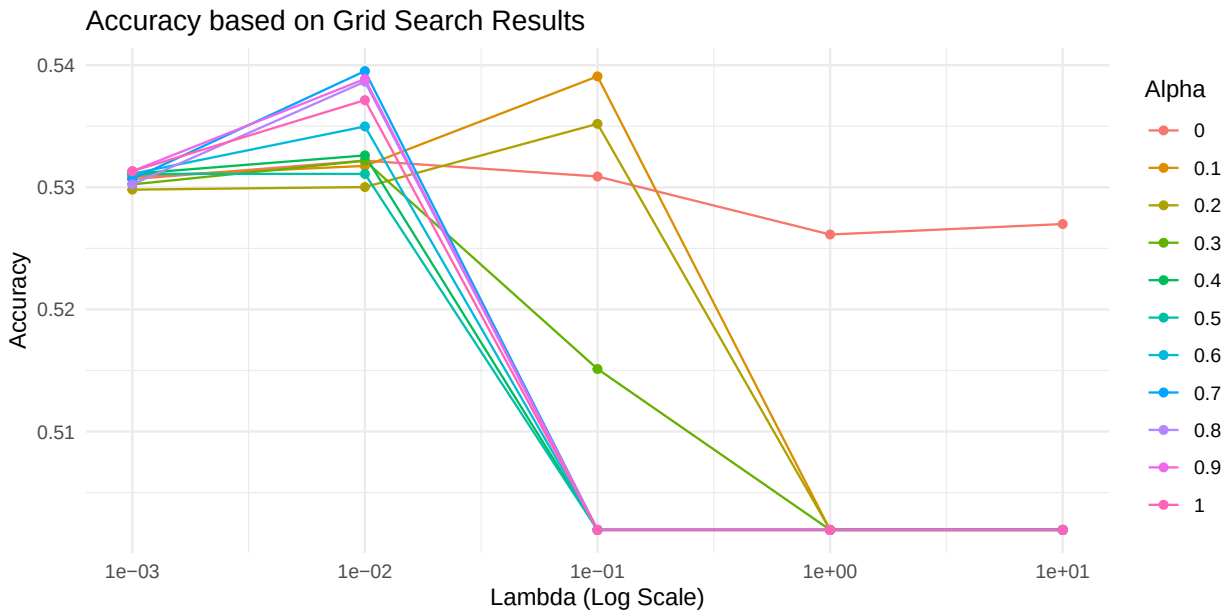
- **Description:** Logistic Regression is a statistical model used for predicting the probability of occurrence of an event by fitting data to a logistic curve. It is particularly adept for binary classification problems. One of the inherent strengths of logistic regression is its ability to estimate the probability of an event, allowing for a nuanced decision-making process. When combined with regularization techniques such as L1 (Lasso) and L2 (Ridge), the model can mitigate overfitting, especially in scenarios with a large feature set.

- **Design:** (i) Hyperparameter Tuning with Cross-Validation: A comprehensive grid search, coupled with 5-fold cross-validation, was conducted over hyperparameters alpha (indicating the type of regularization) and lambda (denoting the regularization strength). Cross-validation ensures a robust estimation of model performance across different subsets of the training data, preventing overfitting and providing a more generalized model.(ii) Model Training and Evaluation: The logistic regression model was trained using the best hyperparameters derived from the grid search. Performance metrics, including accuracy, precision, sensitivity, specificity, and F1 score, were evaluated on both the training and test datasets to understand the model's predictive power. (iii) Rationale for Metric Selection: Accuracy was chosen as the primary metric for hyperparameter tuning and model evaluation due to several factors: the pre-processing steps ensured a balanced representation of both classes, making accuracy a relevant and reliable performance measure. Logistic regression's output of probabilities necessitates a threshold for classification. With a balanced dataset, accuracy provides a clear and direct measure of the model's correctness in predictions. Among various metrics, accuracy is intuitive, straightforward to interpret, and computationally efficient, making it a practical choice for evaluating model performance.
- **The best model parameters by f1 score shown below:**

```
## Best model parameters(maximizing accuracy score):
## - alpha: 0.7
## - lambda: 0.01
## Training scores for the best model:
## - accuracy:  0.5514039
## - precision:  0.5445352
## - sensitivity:  0.6071119
## - specificity:  0.4961274
## - balanced accuracy:  0.5516196
## - f1 score:  0.5741234
```

```
##
## Call:  glmnet(x = as.matrix(train_data[, -ncol(train_data)]), y = train_data$debt_quality,      famil
##
##   Df %Dev Lambda
## 1 25 1.18   0.01
```

- Accuracy scores for different alpha and lambda values shown below:

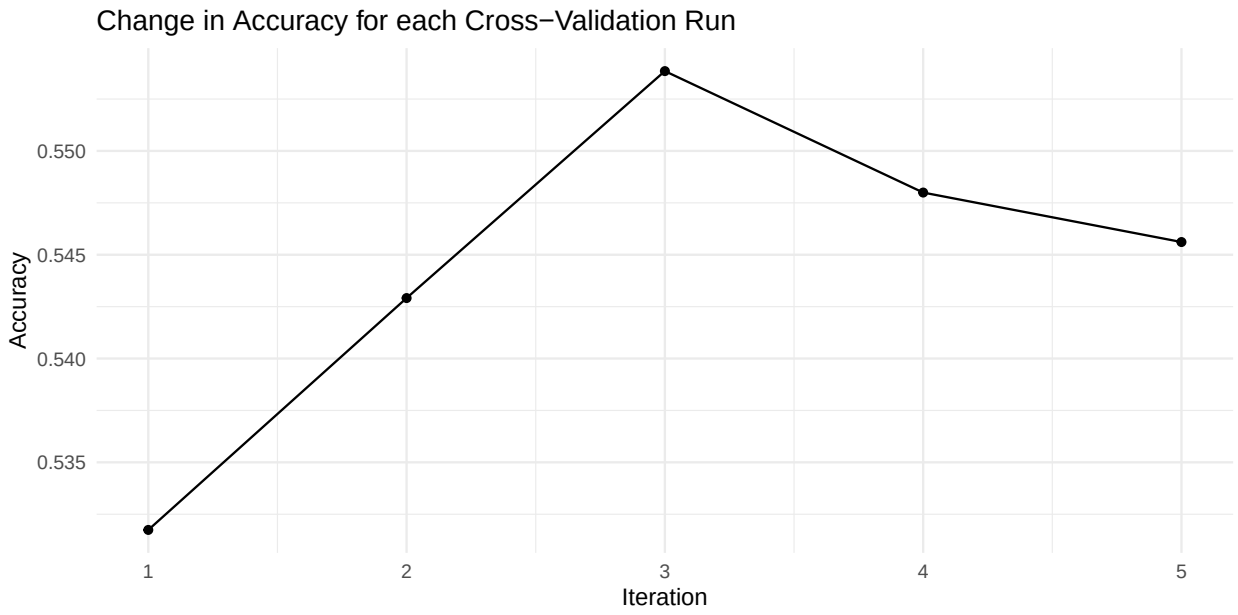


3. Linear discriminant analysis (LDA)

- **Description:** LDA is a dimensionality reduction technique that works by finding the linear combination of features that best separates multiple classes in data. It is very interpretable, and very robust against overfitting since it best estimates the discriminant boundaries using covariance information. LDA does not require any hyperparameter tuning, since the lda() function in the MASS package tunes parameters itself, using the covariance matrices of the input data. This means that for this technique, only cross validation has been done to ensure robustness.
- **Design:** (i) Data pre-processing: min-max scaling, one-hot-encoding and SMOTE were applied to the dataset, (ii) hyperparameter tuning: lda() was run with 5 fold cross-validation, (iii) Rationale for metric selection: best accuracy score is chosen since : 1) The SMOTE pre processing done means that the classes are not too imbalanced, hence accuracy becomes a relevant metric , 2) LDA works in such a way that it uses correlation matrices to find the best results, hence accuracy is generally the more useful metric than sensitivity or precision since this mitigates a lot of the imbalance and 3) accuracy is the easiest metric to interpret and it is the most computationally efficient to calculate.
- **The best model parameters by accuracy score shown below:**

```
## Accuracy on run 1 : 0.531744312026002
## Accuracy on run 2 : 0.542912873862159
## Accuracy on run 3 : 0.553846153846154
## Accuracy on run 4 : 0.547995666305525
## Accuracy on run 5 : 0.545612134344529
```

- Accuracy scores shown below:

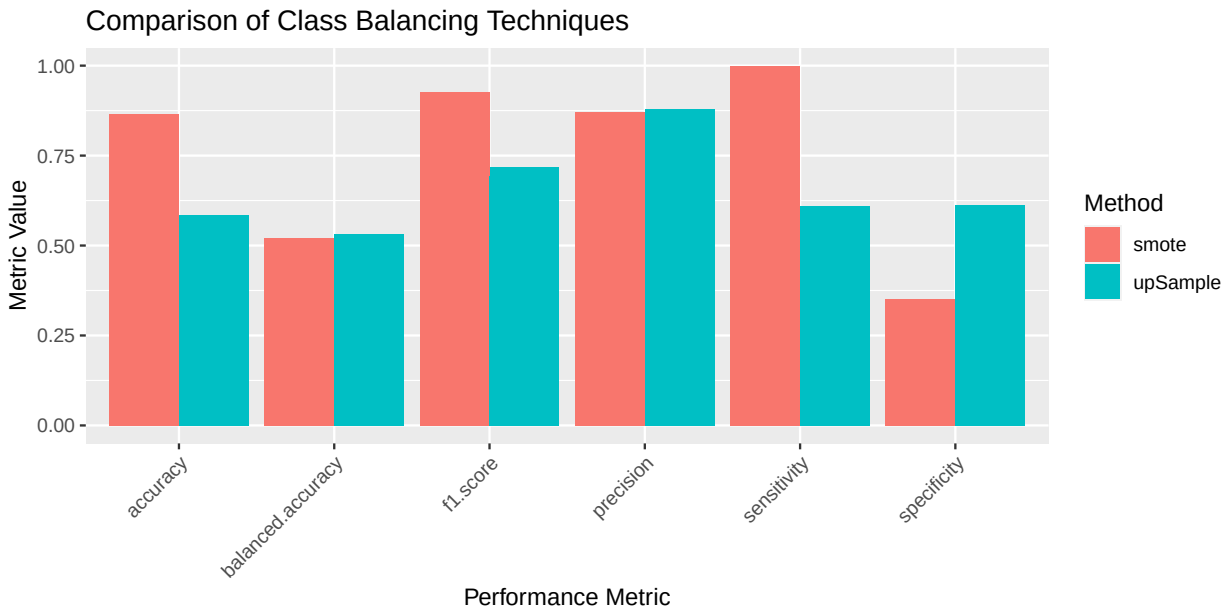


4. AdaBoost

- **Description:** Ada is an ensemble learning algorithm that builds a strong classifier by combining multiple weak learners, in this case decision trees. Ada works by focusing on instances of the dataset that are most difficult to classify by adaptively adjusting weights during successive training rounds.
- **Design:** (i) Data pre-processing: mix-max scaling, one-hot encoding were applied. Data was then oversampled using upSample and SMOTE. (ii) Hyperparameter tuning: grid search was run on iterator, loss functions, maxdepth values as well as the two oversampling techniques. (iii) Rationale for metric selection: balanced accuracy was chosen as it takes into account both the true positive rate (sensitivity) and true negative rate, while mitigating the impact of class imbalance.
- **The best model parameters by accuracy score shown below:**

```
## Best model parameters (maximizing balanced accuracy):
## - loss function:  huber
## - max depth:  5
## - iterations:  60
## - balancing technique:  upSample
##
## Training scores for the best model:
## - accuracy:  0.5575196
## - precision:  0.8790867
## - sensitivity:  0.5669344
## - balanced accuracy:  0.5318473
## - f1 score:  0.6893184
```

- Scores for SMOTE and up-sample shown below:

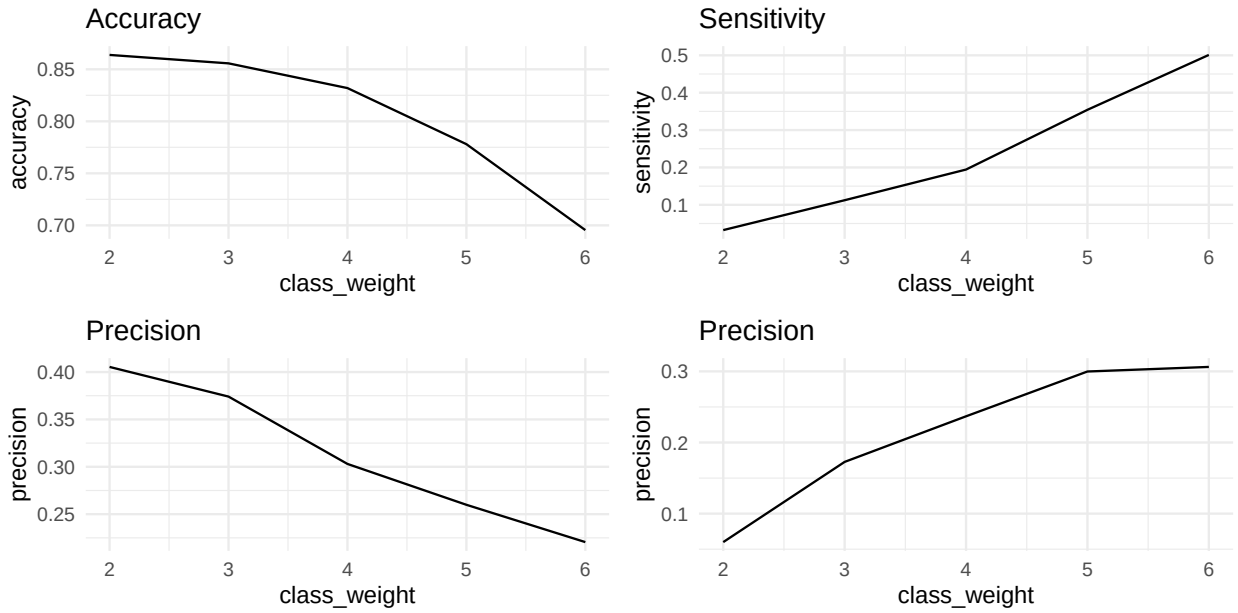


5. Support vector machine

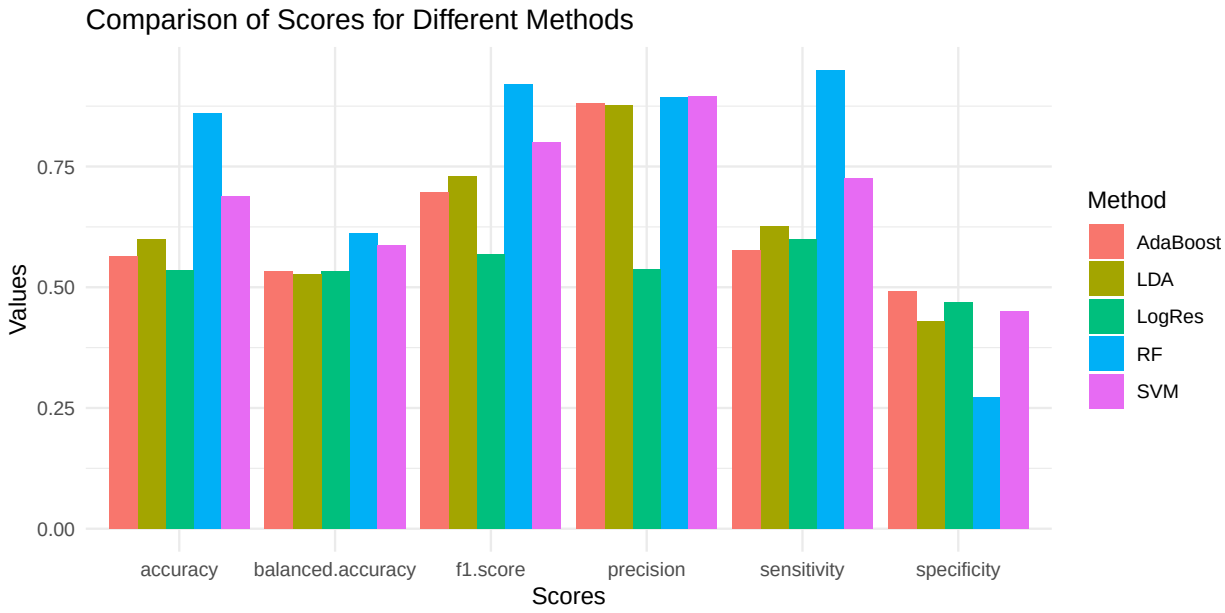
- **Description:** Support Vector Machines (SVM) are a set of supervised learning methods used for classification, regression, and outliers detection. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other. It is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.
- **Design:** (i) Data pre-processing: min-max scaling and one-hot-encoding were applied to the dataset. Principal Component Analysis (PCA) was used on train and test sets for feature selection. (ii) Hyper-parameter tuning: the grid search technique was applied on **'class.weights'** to find the **best f1 score**. (iv) Rationale for metric selection: f1-score was chosen to achieve a good balance between precision (minimizing false positives) and recall (minimizing false negatives). Accuracy was not maximised as the dataset is imbalanced.
- **The best model parameters by f1 score shown below:**

```
## Best model parameters (maximizing f1 score):
## - class weights:  1 6
## Training scores for the best model:
## - accuracy:  0.6954506
## - precision:  0.2205323
## - sensitivity:  0.5010799
## - specificity:  0.7255689
## - balanced accuracy:  0.6133244
## - f1 score:  0.3062706
```

- Plots of results shown below:



RESULTS



Due to data imbalance, performance measures which accounts for true values while minimizing false positives and false negatives were analyzed:

- **Precision = (True Positives / (True Positives + False Positives))**: precision measures the accuracy of a model when it predicts a positive class. All models aside from logistic regression performed relatively well (87%+) which means they were able to correctly identify actual bad debtors and not misclassifying too many good debtors as bad debtors.
- **Sensitivity = (True Positives / (True Positives + False Negatives))**: sensitivity measures the model's ability to correctly identify positive instances from the total number of actual positive

instances in the dataset. Random forest scored highest which means the model was able to correctly identify almost all bad debtors in the test set.

Final model recommendation: We recommend Random Forest (RF) for this analysis, though it still has its limitations, as shown by its low specificity score. This implies that the RF model is 'too strict', where some good debtors are also flagged as bad debtors. This limitation may be caused by data imbalance which has been minimized by SMOTE but still presents some bias. We are comfortable with this flaw as having false positives carry lesser economic impact than having false negatives (e.g., lost revenue instead of bad debt on the banking book).

DISCUSSION

Potential shortcomings and future work: (i) Data imbalance: additional data points on bad debt would address the bias experienced in this dataset. (ii) Data features: additional features could be collected which may have higher correlation with the target feature. For example: customer credit score, criminal records, past history of indebtedness, identifier on any existing factors that would signal lower delinquency risk such as letter of guarantee, employer reference, etc. (iii) Hyperparameter tuning: a wider tuning grid could be applied to ensure that each model achieves its global optimum solution.

Conclusion: Machine learning can be utilized as a layer of first screening for potential bad debtors if it is provided with a rich dataset. Our largest obstacle was data imbalance which can be solved with a larger number (and wider spread) of customer track record and information at point of application. This exercise also makes clear why the use of credit score is prevalent in the credit card industry, as it accumulates credit behaviour data from sources not directly available otherwise at point of application.

REFERENCES

Li, Y., Li, Y., & Li, Y. (2019). What factors are influencing credit card customer's default behavior in China? A study based on survival analysis. Physica A: Statistical Mechanics and Its Applications, 526, 120861. https://doi.org/10.1016/j.physa.2019.04.097

Leong, O. J., & Jayabalan, M. (2019). A Comparative Study on Credit Card Default Risk Predictive Model. Journal of Computational and Theoretical Nanoscience, 16(8), 3591–3595. https://doi.org/10.1166/jctn.2019.8330

Wang, L., Lu, W., & Malhotra, N. K. (2011). Demographics, attitude, personality and credit card features correlate with credit card debt: A view from China. Journal of Economic Psychology, 32(1), 179–193. https://doi.org/10.1016/j.joep.2010.11.006

CONTRIBUTION STATEMENT

All members contribute equally on all aspect of this project.