MATH5836 - Assignment 3

Binary classification with imbalanced data
Vehicle Loan Default Prediction

Tram Dinh

Abstract

This study, using the vehicle loand default provided by L&T Financial Services (and available on Kaggle), aim to compare the performance of bagging and boosting ensemble classifier (random forest and XGBoost) at different setting of hyperparameters (number of trees and maximum tree depth) in the context of an imbalanced dataset. Different methods of mitigating class imbalance - resampling (SMOTE and SMOTEENN), cost-sensitive learning, and a combination of both - are also investigated and compared. The results suggest that both models seems to benefit from larger numbers of trees. Contrary to traditional view, however, random forest models' performance peaks at shallower depth, and deteriorates as maximum tree depths increases due to overfitting. Out of all combination of hyperparameters and balance methods, models using only increased positive class weight outperform ones which combine both increased positive class weight and resampling.

Date: October 2024

Contents

1 Introduction

Data imbalance is one of the most frequently encountered challenge in machine learning classification tasks for real-world application. A lot of examples can be seen in many industries such as fraud detection in finance and insurance, product defect detection in manufacturing, or cancer scanning in health care. When the data is imbalanced, classifiers are prone to over-predict the majority class, and under-predict the minority ones (H. He and Garcia 2009). Unfortunately, the implication of misclassification (false positive or false negative) can range from midly inconvenience to grave social, economic or health consequences.

This study's main aim is to evaluate and compare the performance of two types of ensemble classifier - bagging (Random Forest) and boosting (XGBoost) - in the context of highly imbalanced data, and the effect of mitigating strategies, including adjusted class weights, resampling, and a combination of both, using the vehicle loand default provided by L&T Financial Services for a competition (available on Kaggle). In addition, the study also investigate analyse the effect of two hyperparameters - number of trees and maximum depth - on model performance, and compares the results to conclusions established by several related studies.

2 Data description

2.1 Overview

The dataset used in this study consists of 233,154 rows, each corresponding to one customer identified by an unique ID. The output variable is the defaulted status of the customer, which is encoded as 1 (defaulted/positive) and 0 (not defaulted/negative).

There are 40 variables (not counting unique customer ID) providing various information about the buyers, including:

- personal information: date of birth (which is converted to age at the time of loan disbursement), employment type
- purchase information: value of purchased asset, location, branch ID, dealer ID, ID of employer who logged the disbursement, etc.
- loan information: loan-to-asset ration, amount of loan already dispersed, etc.
- variables indicating the financial situation of the customers at the time of disbursement. These variables provide information about past and present loans that are related to customers' accounts, including loans that the customers took out themselves, and recorded on their primary accounts, and loans for which the customers acted as co-applicants or guarantors (and thus recorded on their secondary account). Examples of information of this type are total number of loans, active loans, defaulted loans in the past, etc.
- variables indicating that customers have been flagged for suspicious behaviours: flagged passport, flagged voter ID, flagged driving license, flagged mobile phone number, etc.
- other information about customers credit, e.g. time since first loans,

For this study, three variables are not considered for the analysis:

- current pincode id - considered irrelevant to the analysis
- disbursal date - all data points have disbursed date within the year 2018
- perform cns score description : this is just a categorical mapping of another continuous variable (cns score)

The full list of variables used in the analysis, together with their descriptive statistics, is available in Appendix A.

2.2 Categorical variables

Table 1 provides information about the percentage of default and non-defaut group in the data. Overall, about 50,000 customers defaulted on their loans, accounting for roughly one-fifth of the total dataset.

Table 1: Proportion of loan default and not default

	default group	**non-default group**	**total**
Count	50,611	182,543	233,154
Percentage	21.7%	78.3%	100%

Surprisingly, the statistics for most categorical features - employment status and flagged behaviours - are very similar between default and non-default group, as shown in Table 2 and 3. For example, all customers in the dataset, regardless of whether they defaulted or not - have their mobile number flagged.
(For this reason, mobile number flag is also removed from the analysis)

Table 2: Percentage of employed customers (as opposed to self-employed individuals) (%)

	default group	**non-default group**	**overall**
employed percentage	39.3	42.7	42

Table 3: Percentage of customers flagged for suspicious documents or accounts (%)

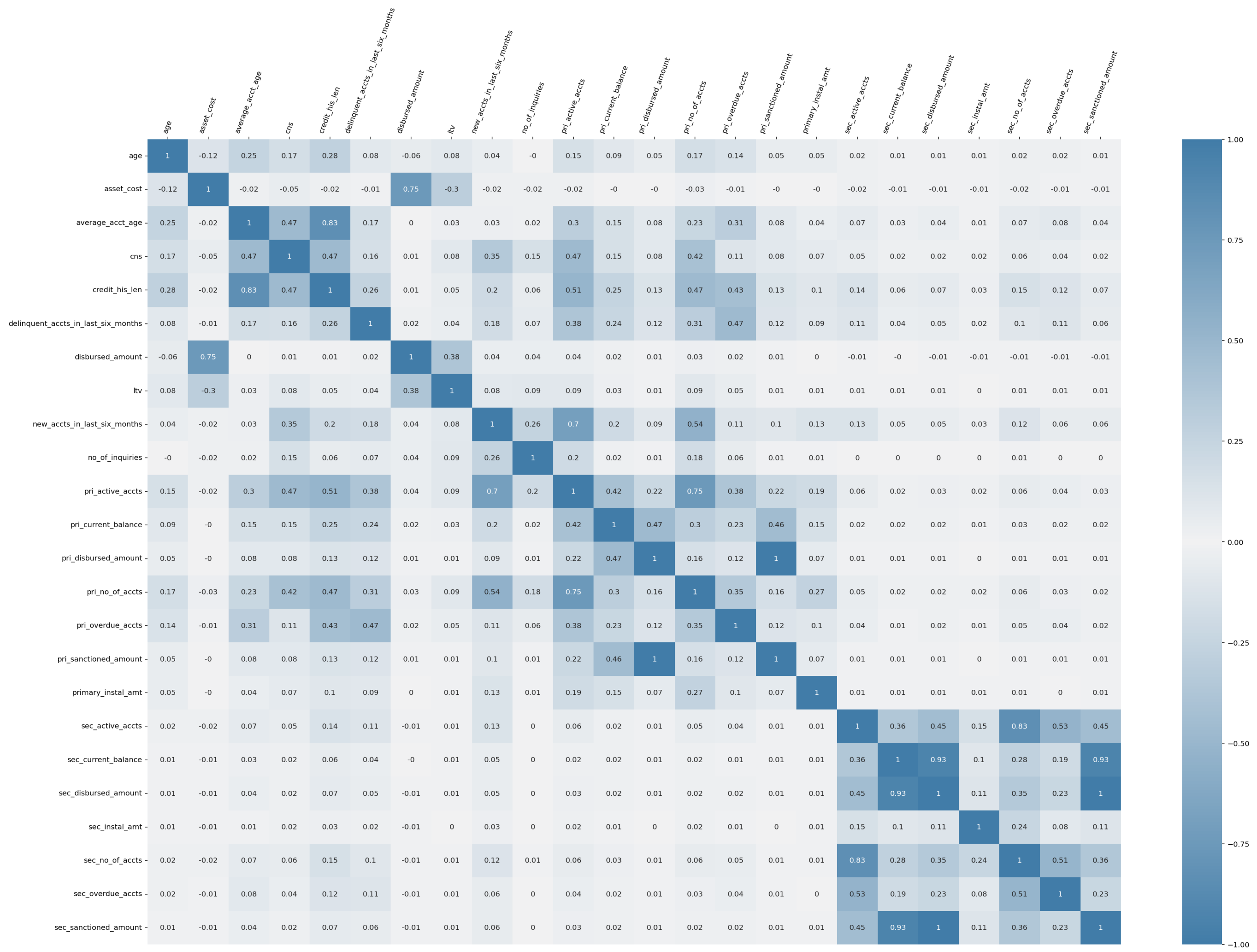
	default group	**non-default group**	**overall**
mobile number	100	100	100
aadhar	81.1	84.8	84
PAN	7.7	7.5	7.6
voter ID	17.4	13.7	14.5
driving license	2.2	2.4	2.3
passport	0.1	0.2	0.2

2.3 Continuous variables

The correlation matrix heatmap (Figure 1) shows that most features do not have correlate strongly with each other except for a group of variables related to customers' accounts and credit (Table 4). A notable example is the sanctioned amount and disbursed amount on both primary and secondary accounts, which have perfect correlation coefficient of 1. This is likely the result of the financial institution's official policy.
(Because of this, both primary sanctioned and secondary sanctioned amounts are remove from the analysis)

Table 4: Pairs of variables with highest correlation coefficient (absolute value above 0.5)

Correlation coeff	**Variable 1**	**Variable 2**
1.0	(primary) sanctioned amount	(primary) disbursed amount
1.0	(secondary) sanctioned amount	secondary disbursed amount
0.93	(secondary) disbursed amount	(secondary) current balance
0.93	(secondary) sanctioned amount	(secondary)current balance
0.83	credit history length	average loan tenure (average.acct.age)
0.83	(secondary) number of account	(secondary) number of active accounts
0.75	disbursed amount	asset cost
0.75	(primary) number of accounts	(primary) number of active accounts
0.7	(primary) number of active accounts	new accounts in last 6 months
0.54	(primary) number of accounts	new accounts in last 6 months
0.53	(secondary) overdue accounts	(secondary) active accounts
0.51	(primary) number of active accounts	credit history length
0.51	(secondary) overdue accounts	(secondary) number of accounts



	age	asset_cost	average_acct_age	cns	credit_his_len	delinquent_accts_in_last_six_months	disbursed_amount	ltv	new_accts_in_last_six_months	no_of_inquiries	pri_active_accts	pri_current_balance	pri_disbursed_amount	pri_no_of_accts	pri_overdue_accts	pri_sanctioned_amount	primary_instal_amt	sec_active_accts	sec_current_balance	sec_disbursed_amount	sec_instal_amt	sec_no_of_accts	sec_overdue_accts	sec_sanctioned_amount
age	1	-0.12	0.25	0.17	0.28	0.08	-0.06	0.08	0.04	-0	0.15	0.09	0.05	0.17	0.14	0.05	0.05	0.02	0.01	0.01	0.01	0.02	0.02	0.01
asset_cost	-0.12	1	-0.02	-0.05	-0.02	-0.01	0.75	-0.3	-0.02	-0.02	-0.02	-0	-0	-0.03	-0.01	-0	-0	-0.02	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01
average_acct_age	0.25	-0.02	1	0.47	0.83	0.17	0	0.03	0.03	0.02	0.3	0.15	0.08	0.23	0.31	0.08	0.04	0.07	0.03	0.04	0.01	0.07	0.08	0.04
cns	0.17	-0.05	0.47	1	0.47	0.16	0.01	0.08	0.35	0.15	0.47	0.15	0.08	0.42	0.11	0.08	0.07	0.05	0.02	0.02	0.02	0.06	0.04	0.02
credit_his_len	0.28	-0.02	0.83	0.47	1	0.26	0.01	0.05	0.2	0.06	0.51	0.25	0.13	0.47	0.43	0.13	0.1	0.14	0.06	0.07	0.03	0.15	0.12	0.07
delinquent_accts_in_last_six_months	0.08	-0.01	0.17	0.16	0.26	1	0.02	0.04	0.18	0.07	0.38	0.24	0.12	0.31	0.47	0.12	0.09	0.11	0.04	0.05	0.02	0.1	0.11	0.06
disbursed_amount	-0.06	0.75	0	0.01	0.01	0.02	1	0.38	0.04	0.04	0.04	0.02	0.01	0.03	0.02	0.01	0	-0.01	-0	-0.01	-0.01	-0.01	-0.01	-0.01
ltv	0.08	-0.3	0.03	0.08	0.05	0.04	0.38	1	0.08	0.09	0.09	0.03	0.01	0.09	0.05	0.01	0.01	0.01	0.01	0.01	0	0.01	0.01	0.01
new_accts_in_last_six_months	0.04	-0.02	0.03	0.35	0.2	0.18	0.04	0.08	1	0.26	0.7	0.2	0.09	0.54	0.11	0.1	0.13	0.13	0.05	0.05	0.03	0.12	0.06	0.06
no_of_inquiries	-0	-0.02	0.02	0.15	0.06	0.07	0.04	0.09	0.26	1	0.2	0.02	0.01	0.18	0.06	0.01	0.01	0	0	0	0	0.01	0	0
pri_active_accts	0.15	-0.02	0.3	0.47	0.51	0.38	0.04	0.09	0.7	0.2	1	0.42	0.22	0.75	0.38	0.22	0.19	0.06	0.02	0.03	0.02	0.06	0.04	0.03
pri_current_balance	0.09	-0	0.15	0.15	0.25	0.24	0.02	0.03	0.2	0.02	0.42	1	0.47	0.3	0.23	0.46	0.15	0.02	0.02	0.02	0.01	0.03	0.02	0.02
pri_disbursed_amount	0.05	-0	0.08	0.08	0.13	0.12	0.01	0.01	0.09	0.01	0.22	0.47	1	0.16	0.12	1	0.07	0.01	0.01	0.01	0	0.01	0.01	0.01
pri_no_of_accts	0.17	-0.03	0.23	0.42	0.47	0.31	0.03	0.09	0.54	0.18	0.75	0.3	0.16	1	0.35	0.16	0.27	0.05	0.02	0.02	0.02	0.06	0.03	0.02
pri_overdue_accts	0.14	-0.01	0.31	0.11	0.43	0.47	0.02	0.05	0.11	0.06	0.38	0.23	0.12	0.35	1	0.12	0.1	0.04	0.01	0.02	0.01	0.05	0.04	0.02
pri_sanctioned_amount	0.05	-0	0.08	0.08	0.13	0.12	0.01	0.01	0.1	0.01	0.22	0.46	1	0.16	0.12	1	0.07	0.01	0.01	0.01	0	0.01	0.01	0.01
primary_instal_amt	0.05	-0	0.04	0.07	0.1	0.09	0	0.01	0.13	0.01	0.19	0.15	0.07	0.27	0.1	0.07	1	0.01	0.01	0.01	0.01	0.01	0	0.01
sec_active_accts	0.02	-0.02	0.07	0.05	0.14	0.11	-0.01	0.01	0.13	0	0.06	0.02	0.01	0.05	0.04	0.01	0.01	1	0.36	0.45	0.15	0.83	0.53	0.45
sec_current_balance	0.01	-0.01	0.03	0.02	0.06	0.04	-0	0.01	0.05	0	0.02	0.02	0.01	0.02	0.01	0.01	0.01	0.36	1	0.93	0.1	0.28	0.19	0.93
sec_disbursed_amount	0.01	-0.01	0.04	0.02	0.07	0.05	-0.01	0.01	0.05	0	0.03	0.02	0.01	0.02	0.02	0.01	0.01	0.45	0.93	1	0.11	0.35	0.23	1
sec_instal_amt	0.01	-0.01	0.01	0.02	0.03	0.02	-0.01	0	0.03	0	0.02	0.01	0	0.02	0.01	0	0.01	0.15	0.1	0.11	1	0.24	0.08	0.11
sec_no_of_accts	0.02	-0.02	0.07	0.06	0.15	0.1	-0.01	0.01	0.12	0.01	0.06	0.03	0.01	0.06	0.05	0.01	0.01	0.83	0.28	0.35	0.24	1	0.51	0.36
sec_overdue_accts	0.02	-0.01	0.08	0.04	0.12	0.11	-0.01	0.01	0.06	0	0.04	0.02	0.01	0.03	0.04	0.01	0	0.53	0.19	0.23	0.08	0.51	1	0.23
sec_sanctioned_amount	0.01	-0.01	0.04	0.02	0.07	0.06	-0.01	0.01	0.06	0	0.03	0.02	0.01	0.02	0.02	0.01	0.01	0.45	0.93	1	0.11	0.36	0.23	1

Figure 1: Correlation matrix of all continuous variables

Due to the high number of variables the full descriptive statistics are provided in appendix . However, Table 5, 6, 7 and Figure 2 provide a quick look at the distribution of some continuous variables describing the financial situation of a customers. From this we can derive some worth-noting observations:

- The dataset is highly skewed with a lot of extreme outliers. An extreme example is the the primary account balance variable, which is demonstrated in Figure 3. Almost all customers have less than 20 in their current account but there are very few people with hundreds of times more money in both default and non-default group.
- For many variables, the distributions in default and not default groups are not very different apart from few outliers, as evidenced by similar mean median and standard deviations.

Table 5: Descriptive statistics of some continuous variables (N = 233,154

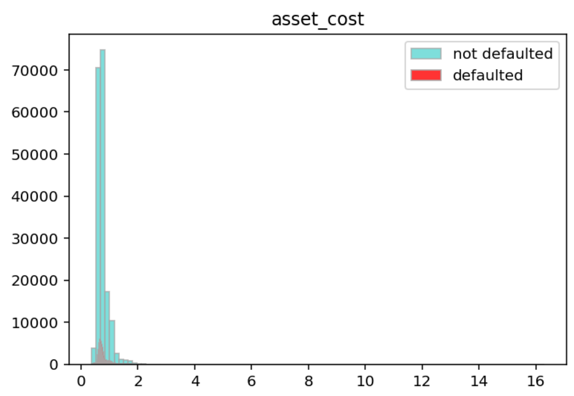
Variables	**mean**	**std**	**min**	**Q1**	**median**	**Q3**	**max**
loan to value	74.747	11.457	10.03	68.88	76.8	83.67	95
asset cost	0.759	0.189	0.37	0.657	0.709	0.792	16.29
current primary balance (rescaled)	1.659	9.423	-66.783	0	0	0.35	965.249
average loan tenure (months)	8.916	15.106	0	0	0	13	369
Bureau Score	289.463	338.375	0	0	0	678	890
Number of default accounts	0.157	0.549	0	0	0	0	25

Table 6: Descriptive statistics of some continuous variables - default group (N = 50,611

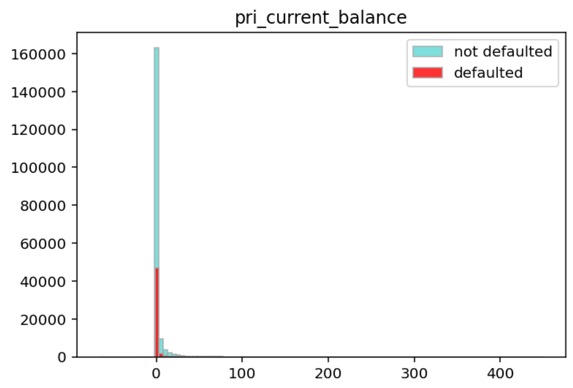
Variables	**mean**	**std**	**min**	**Q1**	**median**	**Q3**	**max**
loan to value	76.883	10.328	15.3	72.055	79.06	84.68	95.0
asset cost	0.764	0.187	0.37	0.659	0.713	0.801	2.812
current primary balance (rescaled)	1.169	7.205	-20.183	0	0	0.257	450.512
average loan tenure (months)	8.205	14.542	0	0	0	12	188
Bureau Score	252.236	318.826	0	0	0	610	879
Number of default accounts	0.199	0.603	0	0	0	0	18

Table 7: Descriptive statistics of some continuous variables - not-default group (N = 182,543

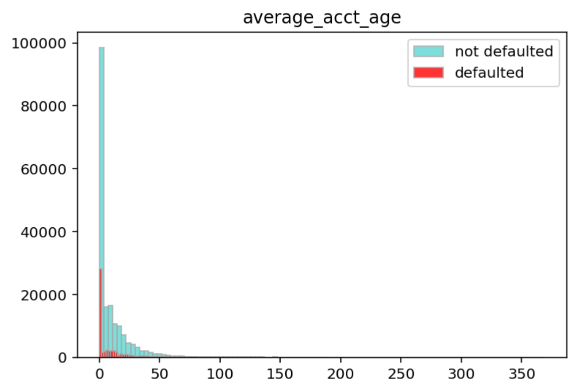
Variables	**mean**	**std**	**min**	**Q1**	**median**	**Q3**	**max**
loan to value	74.154	11.681	10.03	68.02	76	83.16	95
asset cost	0.757	0.19	0.37	0.657	0.708	0.79	16.29
current primary balance (rescaled)	1.795	9.946	-66.783	0	0	0.382	965.249
average loan tenure (months)	9.113	15.253	0	0	0	13	369
Bureau Score	299.784	342.884	0	0	15	690	890
Number of default accounts	0.145	0.532	0	0	0	0	25



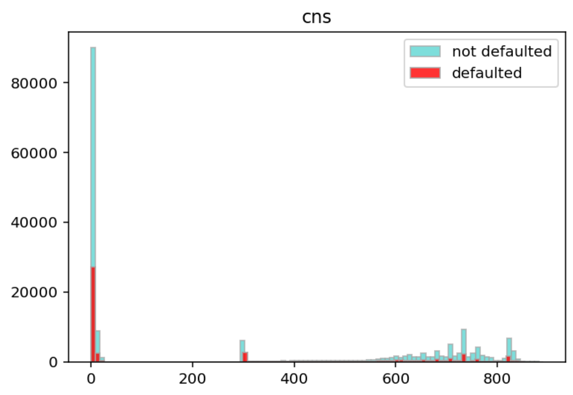
(a) asset cost



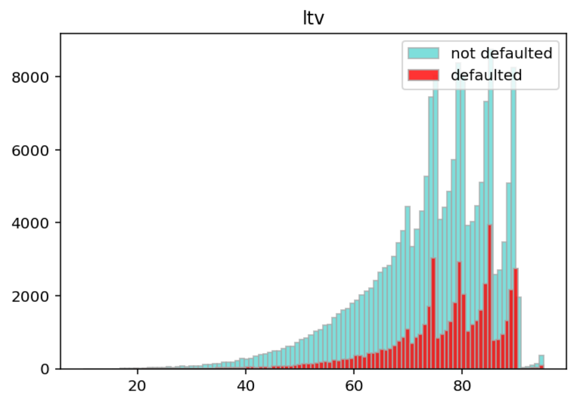
(b) primary account's current balance



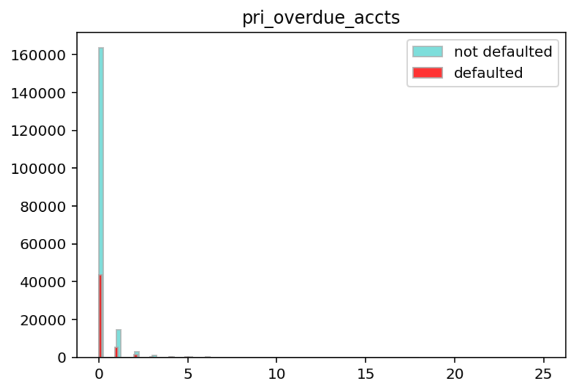
(c) average loan tenure



(d) Bureau Score

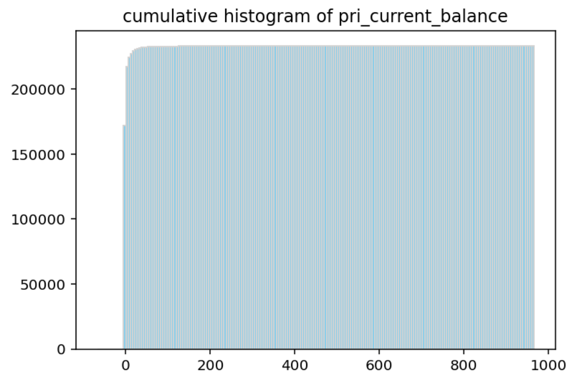


(e) loan-to-value (of asset)

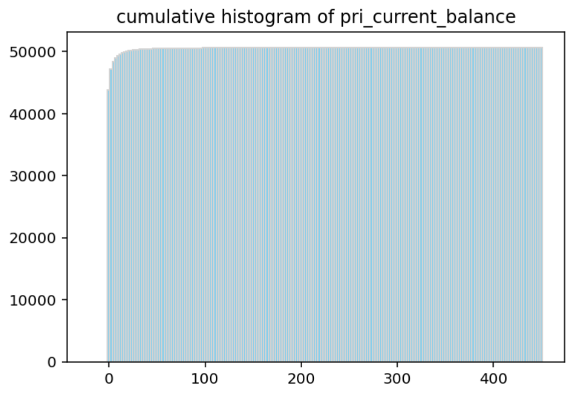


(f) number of default account at disbursement time

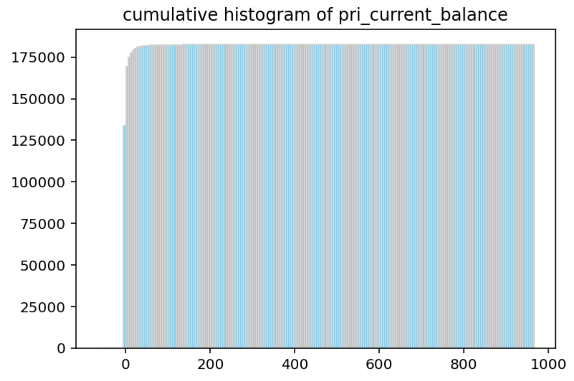
Figure 2: Histogram of some continuous variables



(a) all data



(b) default group



(c) not-default group

Figure 3: Cumulative histogram of primary account balance

3 Data preprocessing

To reduce the range of values and standard deviation (and mitigate the effect of extreme outliers), in this study continuous variables with positive values are log transformed. Then, all continuous variables are standardised using the standardised re-scaler from ski-learn library.

A few outliers belong to the top 0.0001% when comes to primary account balance are also removed manually. Since they all belong to the known default group, which is already over-presented, it should not have any impact on the classifier

4 Literature review and methodology

4.1 Consideration for model selection

4.1.1 Types of models and hyperparameters for tuning

As the data is imbalanced with many variables, single decision tree is not up to the challenge. In fact, even with tuning, resampling and adjusted class-weights, all attempts to apply these two models fails with zero or near-zero precision (all test data classified as majority class). Thus, this report will focus mainly on the application of ensemble models, which are more robust and less prone to over-fitting. Due to resource limitation, only one bagging model and one boosting model are tested.

For bagging model, within the scope of the course, random forest is tested. However, prior to random forest, a quick experiment on single decision tree model is carried out in order to estimate the maximum (possible) tree depth.

For boosting model, XGBoost model is selected. This decision is made after the review of several previous study, which points to high performance of XGBoost compared to other boost ensemble models (e.g. AdaBoost) on imbalanced data (Lai et al. 2021; S. He et al. 2021; Al-Essa and Appice 2021; Nasir et al. 2024).

With each of the above-mentioned models, three hyperparameters are experimented with: (1) total number of trees, (2) the maximum tree depth, and (3) weight of positive/minority class (defaulted), with weight of negative class equal to one. For each combination of tree number, maximum tree depth and positive class weight, performance indicators are recorded and averaged.

4.1.2 Mitigating methods for class imbalance

Two most common types of methods to mitigate the effect of data imbalance are data resampling, and cost-sensitive learning (H. He and Garcia 2009). Cost-sensitive learning, on the other hand, approaches the problem by increasing the cost of minority class misclassification (false positive in this case) during training. One of the most popular methods (and probably the simplest to implement) is modifying class weights for loss function, which uniformly increase the weight of a minority class instance.

Data resampling methods, on the other hand, are carried out before the training to improve model prediction through the re-balancing of training data, either by increasing the presence of the minority class(oversampling), or reducing the presence of the majority class (undersampling). Traditionally, oversampling is done by replicating existing data from the minority class (random oversampling), which sometimes makes the model more prone to overfitting (H. He and Garcia 2009). Meanwhile, undersampling, which removes data from majority class from the training set, could cause loss of information. To address these issues, resample methods utilising information from the dataset have been developed. For oversampling, the most popular method is SMOTE oversampling algorithm in which synthetic data are generated by interpolation from KNN clusters (Chawla et al. 2002). For undersampling, KNN-based methods targeting overlapping regions between different classes to make the decision boundary clearer for classifiers are very popular with two notable examples being Edited Nearest Neighbours (ENN) and Tomek Links. The former removes observations from the majority class if its k closest neighbours are of a different class (Vuttipittayamongkol and Elyan 2020). The choice of k determines how aggressive the removal is. Meanwhile, Tomek Links based undersampling is more conservative as the majority class observation is removed only if it forms a Tomak link with an observation from a different class (Batista, Prati, and Monard 2004).

Many studies also combined both oversampling and undersampling to improve model prediction. Two ready-to-use implementations for this hybrid approach are provided by the the popular library imblearn: SMOTEENN - which combines SMOTE and Edited Nearest Neighbours, and SMOTETomek - which combones SMOTE and Tomek Links. For this study, as the data from default and non-default group are likely to have significant overlapping as shown in the data description section, a more aggressive SMOTEENN might be a better choice.

To sum up, in this study, the performance of ensemble models will be tested in the context of:

- no adjustment
- adjusting class weight with original data
- resampling training data using (1) SMOTE and (2) SMOTEENN
- combining both resampling and adjusting minority class weight

4.1.3 Choice of indicators

Data imbalance also necessitates a careful consideration of performance indicators. There are many indicators to evaluate prediction and fine-tuning classification model, each with its own advantages and disadvantages. As different indicators emphasise different aspect of prediction performance, the choice of indicators often depends on users' purposes, i.e. the underlying nature of the task.

In the context of financial default prediction, both types of error - false positive and false negative - could have significant economical implication. Too many false negative would translate into higher loan default rate, which would put the lenders in trouble. Too many false positive, on the other hands, means that the lender would miss out on profits from good customers who would be able to pay back the loan. Thus, the trade-off between precision and recall is a crucial consideration. In real-life application, there often is no one-size-fit-all threshold as the institution's risk tolerance/risk appetite is taken into account. However, in the absence of this context, the study will mostly rely on F1 score, which harmonises precision and recall, as the main indicator:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN}$$

However, other indicators will also be reported (except for accuracy score, which is often not suitable for highly imbalanced data).

4.2 Model tuning experiment designs

4.2.1 Single decision tree

As mentioned earlier, single decision tree is not the focus of this study, and this experiment's main purpose is to provide an estimate of maximum (possible) depth in random forest model. Thus, a simple design is used in which tree depth is varied from 5 to 60, and the AUC, accuracy score and F1 score are recorded for prediction on both training and testing sets.

4.2.2 XGBoost

For the positive/minority class weight, the experimented range is established around the recommendation of scikit-learn library's "balanced" option for class weights[1], which is estimated by

$$\begin{aligned} class\ weight &= \frac{number\ of\ samples}{number\ of\ classes \times number\ of\ occurance} \\ &= \frac{233,154}{2 \times 50,611} \\ &= 2.3 \end{aligned} \tag{1}$$

As XGBoost often involves many trees with shallow depth (weak classifiers), the experimented ranges for XGBoost model are:

- number of trees: 150, 200, 250, 300, 350, 400, 450, 500
- maximum tree depth: 4, 5, 7, 10, 15, 20
- weight of positive/minority class (with weight of negative class = 1): **1 (i.e. no adjusment)**, 3, 4, 5, 10, 20, 25, 30

4.2.3 Random forest

Similar experiments are also conducted for random forest model, albeit with much less extensive ranges. This is partly because the traditional approach for random forest model favours high depth for each tree, which means an extensive experiment on random forest depth would take too much time considering the available resources within the scope of this study. Thus, a simplified depth range was established based on the result from decision tree experiment, which provides an estimate of the maximum depth for a single tree in the forest at which no further classification is needed, which in this case is 35.

While it is said by many that the best approach to random forest is to let each single decision tree grow to the maximum depth, results from some investigations on high-dimensional data such as Nadi and Moradi 2019 suggest that increasing the number of trees while reducing maximum tree depth might deliver similar performance but with much lower running time. Inspired by this, the study also experiments with a shallow depth of 5 and a medium depth of 20 with different numbers of trees. Eventually, the experiment ranges for random forest model are established as following:

- number of trees: 150, 200,
- maximum tree depth: 5, 20, 35
- weight of positive/minority class: 1 (default - i.e. no balancing), 3, 4, 5, 10, 20

4.2.4 Resampling

For both XGBoost and Random Forest model, due to the time constraints imposed by the SMOTE and SMOTEENN algorithm (which is based on the computationally expensive KNN), the experiment with SMOTE and SMOTEENN, only models with best performance from Random Forest and XGB experiments are selected to conduct experiment on resampling. Thus, the specific range is provided in the result section. Overall, each model will be tested with resampling plus no re-balancing weight (i.e. positive weight = negative weight = 1), and resampling in combination with increased positive class weight.

[1] `https://scikit-learn.org/dev/modules/generated/sklearn.utils.class_weight.compute_class_weight.html`

4.3 Software

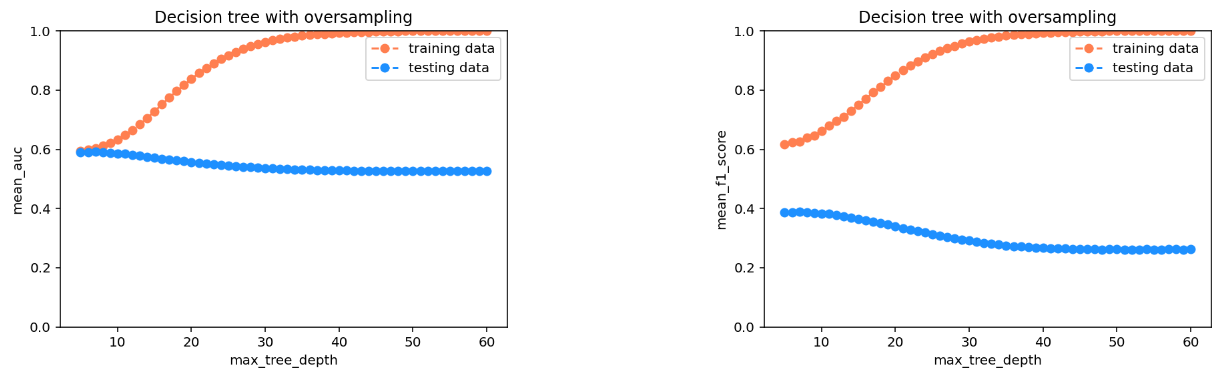
The experiments was conducted in Python3, and the following packages were used:

- Pandas, Numpy: data clearning, data processing and calculation.
- Scikir-learn: data splitting and machine learning models
- xgboost for XGBoost model
- imblearn: resampling (SMOTE and SMOTEENN)
- Matplotlib, Seaborn: data visualisation

5 Results

5.1 Decision tree with oversampling

A very simple experiment with decision tree and random oversampling shows that overfitting start to happen at around maximum depth of 10, and the performance on the train set reach the maximum (i.e.classify every customer correctly) around maximum depth of 35 (Figure 4). As mentioned in the methodology section, this was used as a benchmark to develop testing range for the Random forest model.



(a) mean AUC

(b) mean F1 score

Figure 4: Decision tree prediction performance

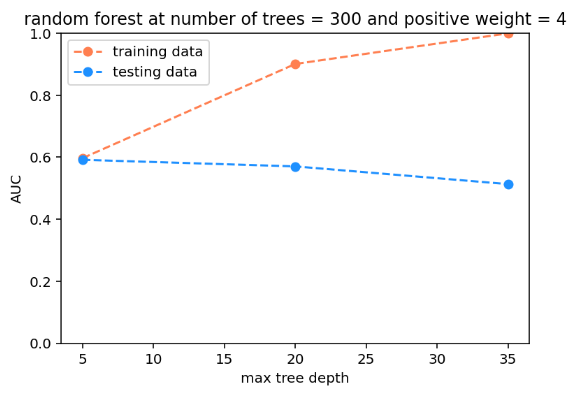
5.2 Random Forest

The best performed models are summarised in Table 8. Contrary to the traditional view that random forest classifiers should have deep trees, the results show that at positive class weight of 4, models with 300 trees and a shallow maximum depth of 5 perform best, giving average F1 score of 0.3958. Figure 5 further shows that when the number of trees and positive class weight are fixed, increasing tree depth just leads to overfitting as evidenced by the difference in training and testing set prediction. The model also becomes to be more biased toward the majority group (non-default) as evidenced by an increase in precision (Figure 5c accompanied by a decrease in recall (Figure 5d). Consequently, performance deteriorates. At a shallow depth of 5, the model performs very similarly between train and test set regardless of number of trees (Figure 6.

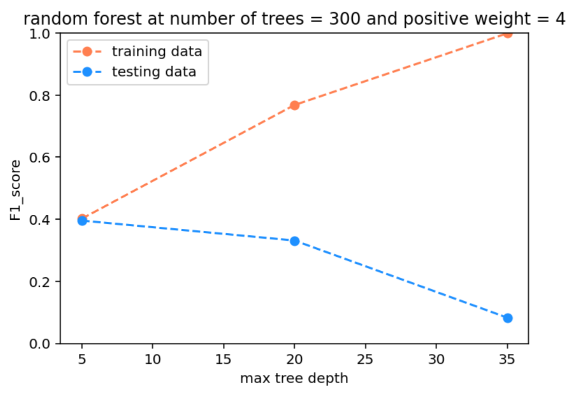
While the best performance indeed came from the largest number of trees in the experiment, Figure 6 shows that when the maximum tree depth and positive weight are fixed, increasing the number of trees from 150 to 300 only leads to a very small improvement in performance.

Table 8: Random Forest - best models based on different indicators

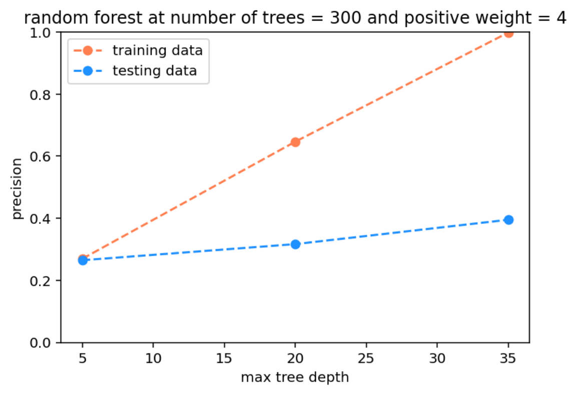
Max indicator	N trees	Max dept	Positive weight	AUC	F1 score	Precision	Recall
AUC	300	5	4.0	**0.5918**	0.3958	0.2649	0.7829
F1 score	300	5	4.0	0.5918	**0.3958**	0.2649	0.7829
precision	300	20	1.0	0.5042	0.021	**0.562**	0.0107
recall	150	5	20.0	0.5	0.3555	0.2162	**1.0**



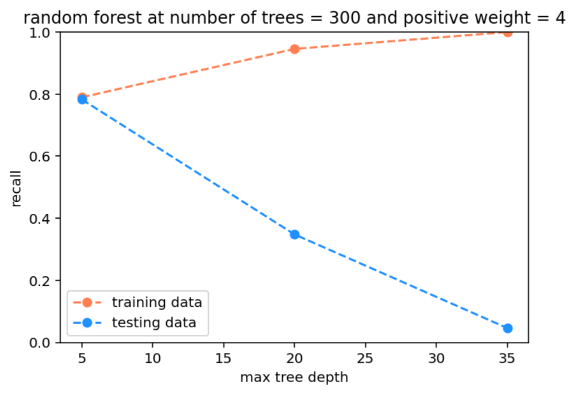
(a) AUC



(b) F1 score

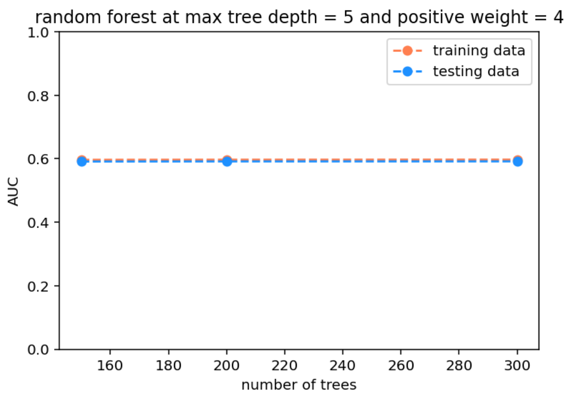


(c) precision

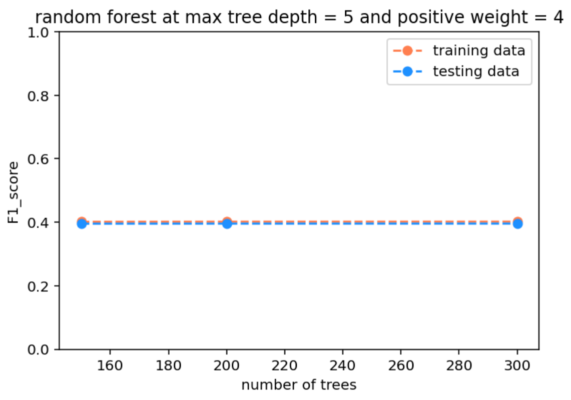


(d) recall

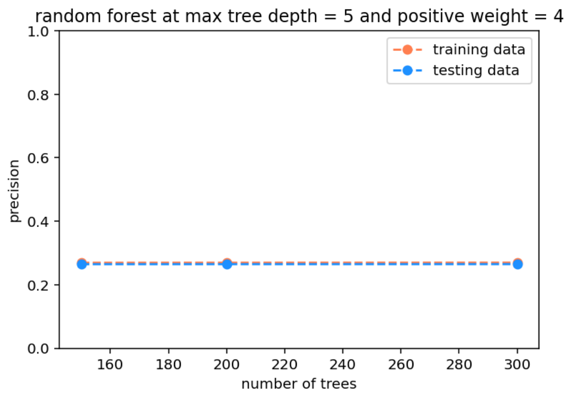
Figure 5: Random forest performance at 300 trees and positive weight = 4



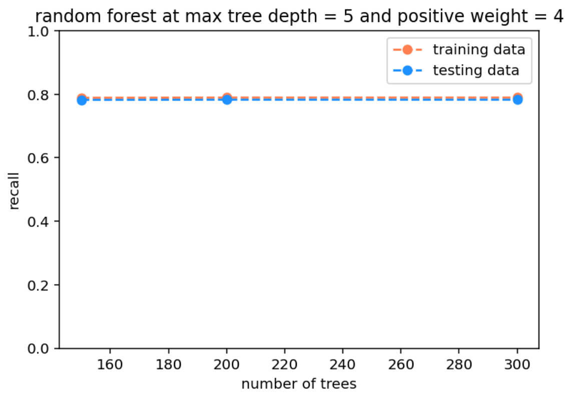
(a) AUC



(b) F1 score



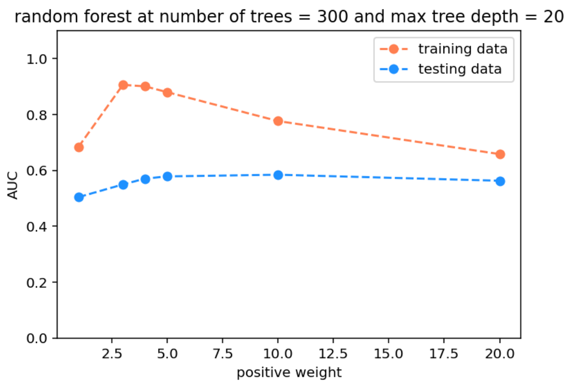
(c) precision



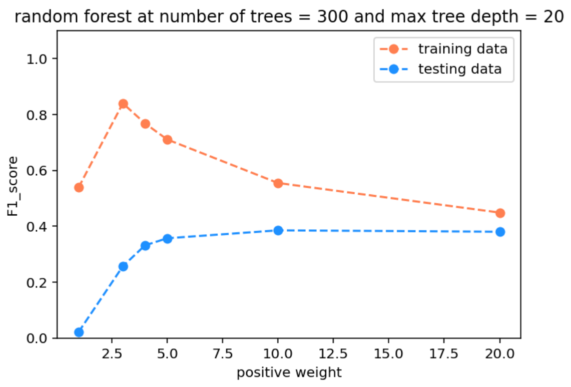
(d) recall

Figure 6: Random forest performance at max depth = 5 and positive class weight = 4

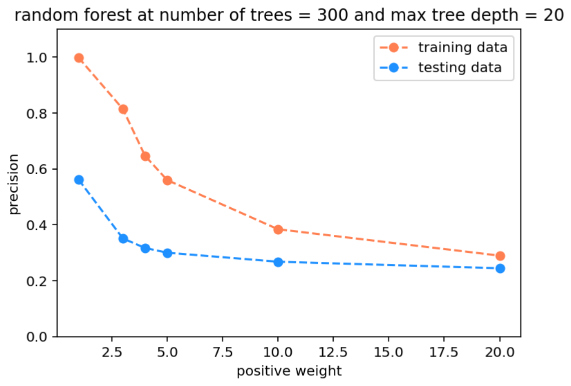
Figure 7 look at the effect of adjusting minority/positive class-weight on model performance when other hyper-parameters are fixed. At positive class weight of 1 - i.e. no balancing - there is considerable overfitting, and the model identifies all customers as defaulted, hence recall and F1 score of 0. As positive class weight increases, model performance starts to improve, and then plateaus.



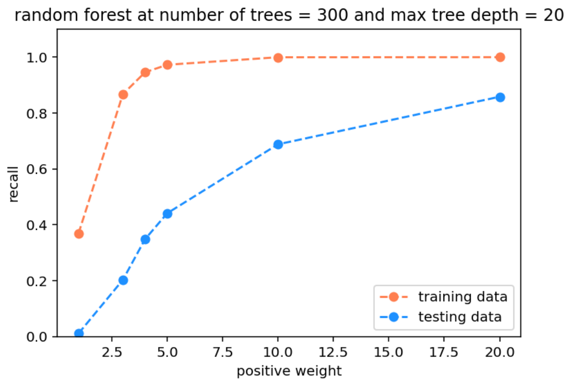
(a) AUC



(b) F1 score



(c) precision



(d) recall

Figure 7: Random forest performance at 300 trees, and max depth of 20

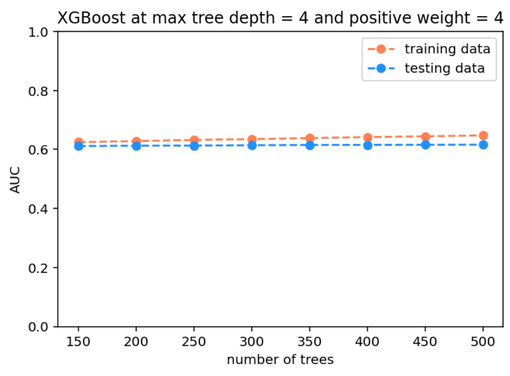
5.3 XGBoost

Table 9 summarises the best average results from the XGBoost experiments based on all performance indicators. Overall, models with 500 trees, maximum tree depth of 4, and positive class weight of 4 perform best when it comes to the main indicator of interest -i.e. F1 score (0.4126 on average). Models with 200 trees at higher depth of 7 have very similar results with slightly lower F1 score (0.4126) and slightly higher AUC (0.6169). The differences between the performance of these two settings are essentially negligible.

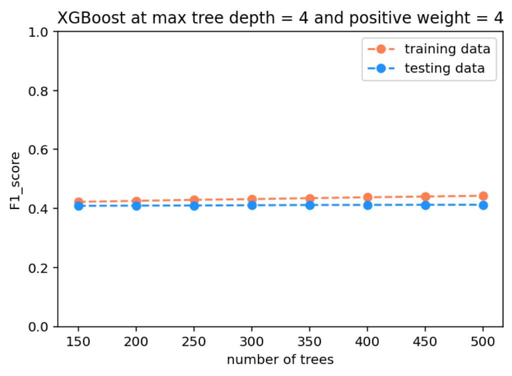
The model with the least number of tree (150) and shallowest trees has highest precision of 0.5846 at positive weight of 1 (i.e. default weight with no re-balancing), and highest recall - 0.9987 - at positive weight of 30 (i.e. the minority class's weight is excessively scaled up). This is part of the trade-off between precision and recall in the context of imbalanced data, which is further illustrated in Figure 9. As the positive class weight increases, the cost of false negative becomes higher, so the model starts to classify more cases as positive/default, which increases recall at the expense of precision. At one point, the model plateaus.

Table 9: XGBoost performance by different indicators (average)

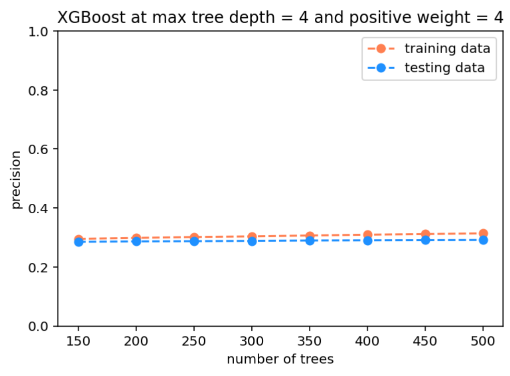
Max indicator	N trees	Max dept	Positive weight	AUC	F1 score	Precision	Recall
AUC	200	7	4.0	**0.6169**	0.4122	0.2969	0.6742
F1 score	500	4	4.0	0.6164	**0.4126**	0.2917	0.7046
precision	150	4	1.0	0.5037	0.0181	**0.5846**	0.0092
recall	150	4	30.0	0.5033	0.3569	0.2173	**0.9987**



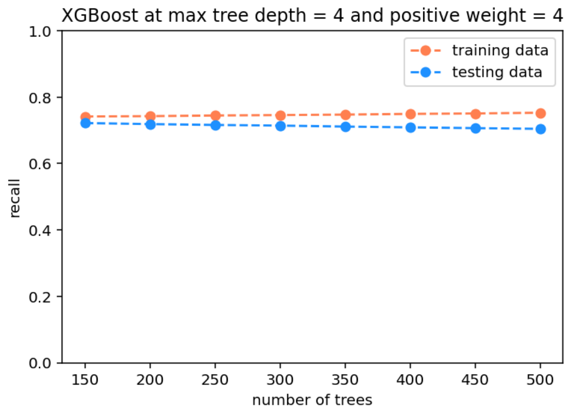
(a) AUC



(b) F1 score

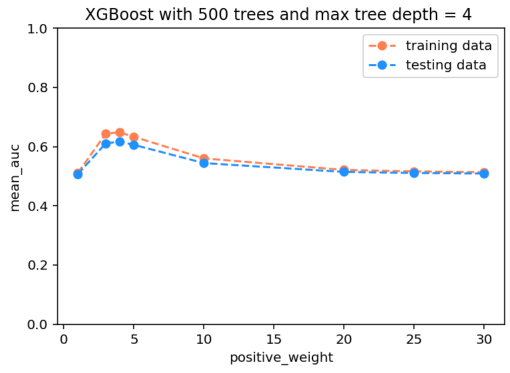


(c) precision

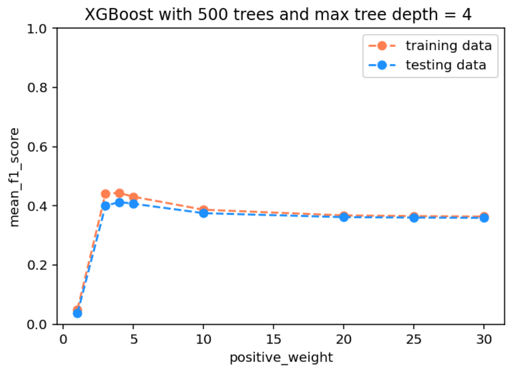


(d) recall

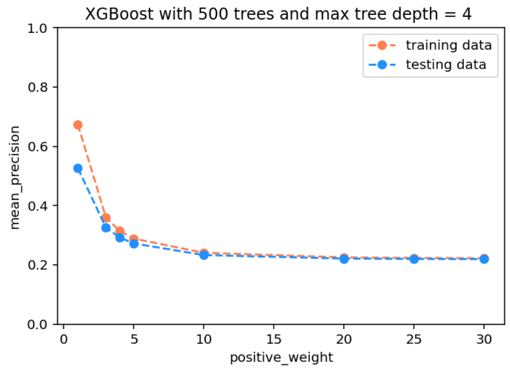
Figure 8: XGBoost at max depth = 4 and positive class weight = 4



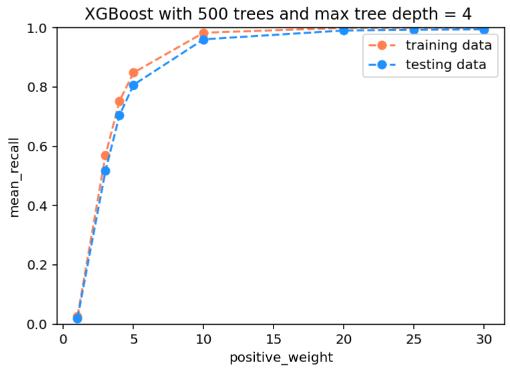
(a) AUC



(b) F1 score



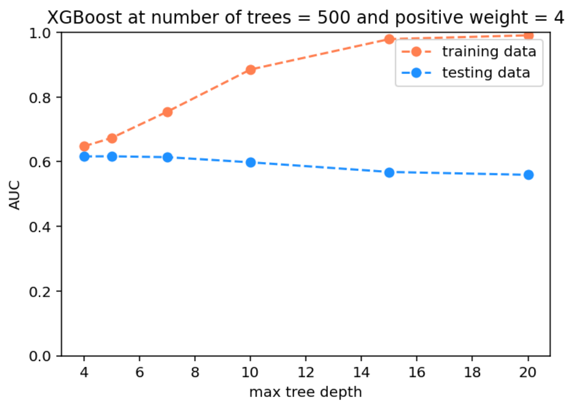
(c) precision



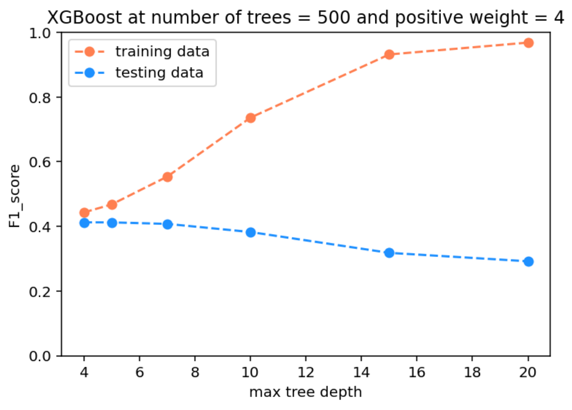
(d) recall

Figure 9: XGBoost model performance over different positive class weights with fixed number of trees (500) and maximum depth (4)

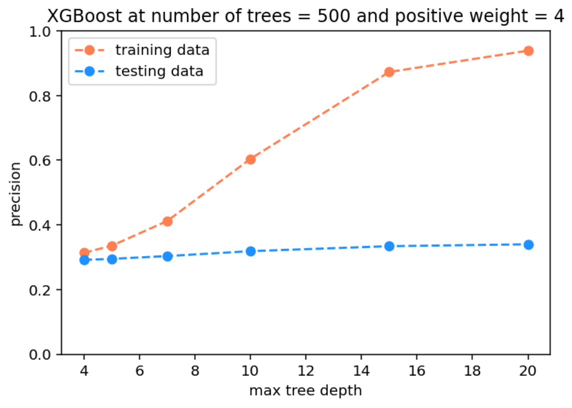
Similar to the case of random forest model, increasing the maximum depth leads to performance drop due to overfitting as the model performs better on train set but worse on test set (Figure 10).



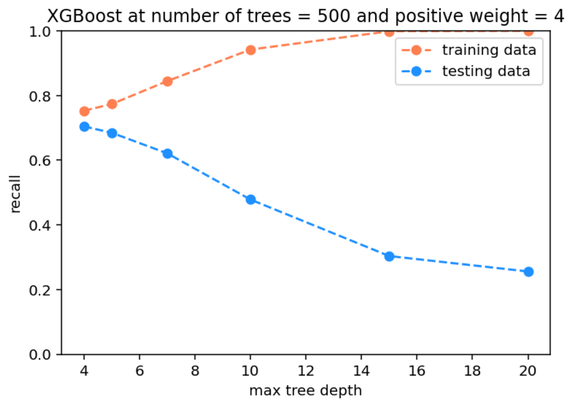
(a) AUC



(b) F1 score



(c) precision



(d) recall

Figure 10: XGBoost at number of trees = 500 and positive weight = 4

5.4 Resampling

For both random forest, the experimented values for positive class weight is 1 (default, no balancing), 2, 3 and 4 (the best option from the previous section). For random forest, the ranges for other parameters are:

- number of trees: 200, 300
- maximum depth: 5, 20, 35

For XGBoost, the ranges are:

- number of trees: 200, 500
- maximum depth: 4, 7

5.4.1 Random forest

The best performed models with SMOTE and SMOTEENN are summarise in Table 10 and 11. This times, models with 200 trees and maximum depths of 5 perform best for SMOTE (average F1 of 0.3754), models with 200 trees and max depth of 20 perform best for SMOTEENN (average F1 of 0.3895). However, none of them outperform the result of random forest or XGBoost with only increased minority class weight from the previous section.

Table 10: Random forest performance with SMOTE by different indicators

Max indicator	N trees	Max dept	Positive weight	AUC	F1 score	Precision	Recall
AUC	300	20	4.0	**0.5787**	0.3753	0.2708	0.6109
F1 score	200	5	2.0	0.5491	**0.3754**	0.236	0.917
precision	300	35	1.0	0.5343	0.21	**0.3332**	0.1533
recall	200	5	4.0	0.5046	0.3575	0.2177	**0.9977**

Table 11: Random forest - best performed models with SMOTEENN by different indicators

Max indicator	N trees	Max dept	Positive weight	AUC	F1 score	Precision	Recall
AUC	300	20	3.0	**0.5923**	0.3888	0.2809	0.6311
F1 score	200	20	4.0	0.5898	**0.3895**	0.2731	0.6792
precision	300	35	1.0	0.5755	0.338	**0.3273**	0.3494
recall	300	5	4.0	0.5044	0.3578	0.218	**0.9976**

5.4.2 XGBoost

Table 12 and 13 summarises the best results when SMOTE and SMOTEENN are applied to the model. In both cases, the best results in terms of F1 score and AUC come from a combination of resampling and increasing positive class weight, and highest number of trees (500). The highest average F1 score with SMOTE is 0.401 from models with maximum depth of 7 and positive class weight of 4. SMOTENN models have slightly better average at 4.035, achieved by models with maximum depth of 4 and positive class weight of 3. Neither of them, however, outperforms the best models with only adjusted positive class weight models (F1 average of 0.4126 from XGBoost with 500 trees and maximum depth of 4 at positive weight of 4 - Table 9).

Table 12: XGBoost performance with SMOTE by different indicators

Max indicator	N trees	Max dept	Positive weight	AUC	F1 score	Precision	Recall
AUC	500	7	3.0	**0.6038**	0.3961	0.2993	0.5857
F1 score	500	7	4.0	0.6027	**0.401**	0.28	0.7061
precision	500	7	1.0	0.5264	0.1434	**0.4137**	0.0868
recall	200	4	4.0	0.5795	0.3901	0.2536	**0.8455**

Table 13: XGBoost performance with SMOTEENN by different indicators

Max indicator	N trees	Max dept	Positive weight	AUC	F1 score	Precision	Recall
AUC	500	7	3.0	**0.6066**	0.4008	0.2954	0.6228
F1 score	500	4	3.0	0.6051	**0.4035**	0.2803	0.7202
precision	500	7	1.0	0.5753	0.3266	**0.3542**	0.303
recall	200	4	4.0	0.5803	0.3914	0.2531	**0.8626**

6 Discussion

Overall, the results from tested models are not very high. This could be due to the nature of the dataset as the defaut group and non-default group shares very similar distribution for many variables. As the data is originally for a competition organised by L&T Financial Services as a part of their recruitment, there is a possibility that the data already contain a lot of difficult borderline cases.

While the traditional approach to random forest is to let the tree grow to maximum possible depth, the results from this study show that deeper trees could lead to a deterioration in prediction performance due to overfitting. On the other hand, increasing the number of trees improves performance in general. This seems to be consistent with the conclusion of Nadi and Moradi 2019, which suggests that random forest might miss some useful subset of features and sample with an inadequate number of trees. However the improvement recorded in this study is too small for a definite conclusion.

Generally, XGBoost models has better performance than Random Forest on this dataset, with the best performance achieved by models with 500 trees and maximum depth of 4. This is at minority class weight of 4, which is almost twice as high as the "balanced" class weight option provided by ski-learn library of 2.3 (as calculated in the methodology section).

7 Conclusion

Overall, the results show that both tested ensemble methods perform well with large number of trees at shallow depth. As the maximum depth increases, even random forest model's performance drops as opposed to common perception. Both resampling and adjusted class weight improves model performance in the context of class imbalance when implemented separately. For this study, combining both of them also delivers decent performance, but not better than just using adjusted class weight.

References

[1] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. "A study of the behavior of several methods for balancing machine learning training data". In: *ACM SIGKDD explorations newsletter* 6.1 (2004), pp. 20–29.

[2] Nitesh V Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[3] Malik Al-Essa and Annalisa Appice. "Dealing with imbalanced data in multi-class network intrusion detection systems using xgboost". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2021, pp. 5–21.

[4] Haibo He and Edwardo A. Garcia. "Learning from Imbalanced Data". In: *IEEE Transactions on Knowledge and Data Engineering* 21.9 (2009), pp. 1263–1284. DOI: 10.1109/TKDE.2008.239.

[5] Shen He et al. "An Effective Cost-Sensitive XGBoost Method for Malicious URLs Detection in Imbalanced Dataset". In: *IEEE Access* 9 (2021), pp. 93089–93096. DOI: 10.1109/ACCESS.2021.3093094.

[6] Sharmeen Binti Syazwan Lai et al. "Comparing the performance of AdaBoost, XGBoost, and logistic regression for imbalanced data". In: *Mathematics and Statistics* 9.3 (2021), pp. 379–385.

[7] Abolfazl Nadi and Hadi Moradi. "Increasing the views and reducing the depth in random forest". In: *Expert Systems with Applications* 138 (2019), p. 112801.

[8] Fahim Nasir et al. "Data-Driven Decision-Making for Bank Target Marketing Using Supervised Learning Classifiers on Imbalanced Big Data." In: *Computers, Materials & Continua* 81.1 (2024).

[9] Pattaramon Vuttipittayamongkol and Eyad Elyan. "Neighbourhood-based undersampling approach for handling imbalanced and overlapped data". In: *Information Sciences* 509 (2020), pp. 47–70. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2019.08.062. URL: https://www.sciencedirect.com/science/article/pii/S0020025519308114.

A Descriptive statistics of all variables

Table 14: Descriptive statistics - all data

Variables	**mean**	**std**	**min**	**Q1**	**median**	**Q3**	**max**
disbursed amount	0.544	0.13	0.133	0.471	0.538	0.604	9.906
asset cost	0.759	0.189	0.37	0.657	0.709	0.792	16.29
loan-to-value	74.747	11.457	10.03	68.88	76.8	83.67	95
branch ID	72.936	69.835	1	14	61	130	261
supplier ID	19638.635	3491.95	10524	16535	20333	23000	24803
manufacturer ID	69.028	22.141	45	48	86	86	156
state ID	7.262	4.482	1	4	6	10	22
employee ID	1549.477	975.261	1	713	1451	2362	3795
mobile number flag	1	0	1	1	1	1	1
aadhar flag	0.84	0.366	0	1	1	1	1
PAN flag	0.076	0.264	0	0	0	0	1
voter ID flag	0.145	0.352	0	0	0	0	1
driving license flag	0.023	0.151	0	0	0	0	1
passport flag	0.002	0.046	0	0	0	0	1
Bureau score	289.463	338.375	0	0	0	678	890
(primary) number of accounts	2.441	5.217	0	0	0	3	453
(primary) active accounts	1.04	1.941	0	0	0	1	144
(primary) overdue accounts	0.157	0.549	0	0	0	0	25
(primary) current balance	1.659	9.423	-66.783	0	0	0.35	965.249
(primary) sanctioned amount	2.185	23.748	0	0	0	0.625	10000
(primary) disbursed amount	2.181	23.777	0	0	0	0.608	10000
(secondary) number of accounts	0.059	0.627	0	0	0	0	52
(secondary) active accounts	0.028	0.316	0	0	0	0	36
(secondary) overdue accounts	0.007	0.111	0	0	0	0	8
(secondary) current balance	0.054	1.702	-5.746	0	0	0	360.329
(secondary) sanctioned amount	0.073	1.832	0	0	0	0	300
(secondary) disbursed amont	0.072	1.826	0	0	0	0	300
(primary) EMI amount	0.131	1.514	0	0	0	0.02	256.428
(secondary) EMI amount	323.268	15553.691	0	0	0	0	4170901
new accounts (last 6 months)	0.382	0.955	0	0	0	0	35
delinquent accts (last 6 months)	0.097	0.384	0	0	0	0	20
average loan tenure	8.916	15.106	0	0	0	13	369
credit history length	16.252	28.581	0	0	0	24	468
Number of inquiries	0.207	0.706	0	0	0	0	36
defaulted	0.217	0.412	0	0	0	0	1
age	34.101	9.806	18	26	32	41	69
employed	0.42	0.494	0	0	0	1	1

Table 15: Descriptive statistics - defaulted group

Variables	**mean**	**std**	**min**	**Q1**	**median**	**Q3**	**max**
disbursed amount	0.563	0.122	0.134	0.493	0.555	0.619	1.914
asset cost	0.764	0.187	0.37	0.659	0.713	0.801	2.812
loan-to-value	76.883	10.328	15.3	72.055	79.06	84.68	95
branch ID	76.941	71.959	1	16	64	135	261
supplier ID	19820.059	3452.023	10524	16680	20597	23108	24803
manufacturer ID	67.975	22.265	45	45	51	86	153
state ID	7.671	4.558	1	4	6	12	22
employee ID	1587.737	979.694	1	738	1512	2414	3795
mobile number flag	1	0	1	1	1	1	1
aadhar flag	0.811	0.391	0	1	1	1	1
PAN flag	0.077	0.266	0	0	0	0	1
voter ID flag	0.174	0.379	0	0	0	0	1
driving license flag	0.022	0.145	0	0	0	0	1
passport flag	0.001	0.038	0	0	0	0	1
Bureau score	252.236	318.826	0	0	0	610	879
(primary) number of accounts	2.089	5.04	0	0	0	2	453
(primary) active accounts	0.887	1.67	0	0	0	1	35
(primary) overdue accounts	0.199	0.603	0	0	0	0	18
(primary) current balance	1.169	7.205	-20.183	0	0	0.257	450.512
(primary) sanctioned amount	1.675	45.257	0	0	0	0.5	10,000
(primary) disbursed amount	1.677	45.271	0	0	0	0.5	10,000
(secondary) number of accounts	0.049	0.527	0	0	0	0	38
(secondary) active accounts	0.024	0.288	0	0	0	0	22
(secondary) overdue accounts	0.007	0.11	0	0	0	0	6
(secondary) current balance	0.036	1.06	-0.096	0	0	0	107.16
(secondary) sanctioned amount	0.051	1.307	0	0	0	0	119
(secondary) disbursed amont	0.05	1.304	0	0	0	0	119
(primary) EMI amount	0.101	1.247	0	0	0	0.019	154.204
(secondary) EMI amount	277.528	11045.628	0	0	0	0	1,447,600
new accounts (last 6 months)	0.329	0.886	0	0	0	0	20
delinquent accts (last 6 months)	0.123	0.431	0	0	0	0	12
average loan tenure	8.205	14.542	0	0	0	12	188
credit history length	13.966	25.519	0	0	0	21	468
Number of inquiries	0.265	0.835	0	0	0	0	19
defaulted	1	0	1	1	1	1	1
age	33.407	9.659	18	25	31	40	64
employed	0.393	0.489	0	0	0	1	1

Table 16: Descriptive statistics - non-defaulted group

Variables	mean	std	min	Q1	median	Q3	max
disbursed amount	0.538	0.131	0.133	0.464	0.533	0.599	9.906
asset cost	0.757	0.19	0.37	0.657	0.708	0.79	16.29
loan-to-value	74.154	11.681	10.03	68.02	76	83.16	95
branch ID	71.826	69.194	1	13	61	121	261
supplier ID	19588.334	3501.284	10524	16445	20289	22995	24803
manufacturer ID	69.32	22.098	45	48	86	86	156
state ID	7.149	4.454	1	4	6	9	22
employee ID	1538.869	973.765	1	704	1436	2345	3794
mobile number flag	1	0	1	1	1	1	1
aadhar flag	0.848	0.359	0	1	1	1	1
PAN flag	0.075	0.264	0	0	0	0	1
voter ID flag	0.137	0.344	0	0	0	0	1
driving license flag	0.024	0.152	0	0	0	0	1
passport flag	0.002	0.048	0	0	0	0	1
Bureau score	299.784	342.884	0	0	15	690	890
(primary) number of accounts	2.538	5.261	0	0	1	3	354
(primary) active accounts	1.082	2.008	0	0	0	1	144
(primary) overdue accounts	0.145	0.532	0	0	0	0	25
(primary) current balance	1.795	9.946	-66.783	0	0	0.382	965.249
(primary) sanctioned amount	2.326	12.343	0	0	0	0.691	1058.657
(primary) disbursed amount	2.32	12.402	0	0	0	0.667	1057.557
(secondary) number of accounts	0.062	0.652	0	0	0	0	52
(secondary) active accounts	0.029	0.323	0	0	0	0	36
(secondary) overdue accounts	0.007	0.111	0	0	0	0	8
(secondary) current balance	0.059	1.841	-5.746	0	0	0	360.329
(secondary) sanctioned amount	0.079	1.952	0	0	0	0	300
(secondary) disbursed amont	0.078	1.946	0	0	0	0	300
(primary) EMI amount	0.14	1.58	0	0	0	0.02	256.428
(secondary) EMI amount	335.95	16588.053	0	0	0	0	4,170,901
new accounts (last 6 months)	0.397	0.973	0	0	0	0	35
delinquent accts (last 6 months)	0.091	0.37	0	0	0	0	20
average loan tenure	9.113	15.253	0	0	0	13	369
credit history length	16.886	29.342	0	0	0	24	449
Number of inquiries	0.19	0.666	0	0	0	0	36
defaulted	0	0	0	0	0	0	0
age	34.293	9.838	18	26	33	41	69
employed	0.427	0.495	0	0	0	1	1