厦门大学经济学院

课程论文

科目：数据挖掘与机器学习

主题：**Comparative Analysis of Machine Learning Methods for Bankruptcy Prediction: A Case Study on Taiwanese Bankruptcy Prediction Dataset**

小组成员：李骁、邱亮、邱奕超

1 Introduction

In recent years, the global economic landscape has witnessed a surge in the application of machine learning (ML) methods to address complex financial challenges. Bankruptcy or business failure can have a negative impact both on the enterprise itself and the global economy. For example, the bankruptcy of Silicon Valley Bank has had a great impact on the financial system. Recognizing the critical need for accurate identification of bankruptcy, this project endeavors to compare multiple well known traditional machine learning methods and propose the most effective one with regard to this specific problem.

Firstly, this study examined the basic details of the data and found that it had high dimensions and contained some abnormal data points. After removing the abnormal data and performing dimensionality reduction, 5 different traditional machine learning models are trained on the cleaned data, and 5-fold cross-validation are used to measure model performance. The final results showed that XGBoost had the highest F2 score, demonstrating outstanding effectiveness in predicting bankruptcies in Taiwan.

2 Research Methodology

2.1 The Data Source

The dataset used in this project is the Taiwanese Bankruptcy Prediction dataset from UCI Machine Learning Repository. The dataset contains 96 financial ratios and 1 categorical variable, which is the bankruptcy status of the company. The dataset contains 6819 data points, of which 220 are bankrupt companies and 6599 are non-bankrupt companies (Figure 1). Moreover, it is highly imbalanced, with only 3.2% of the data points belonging to the bankrupt class.

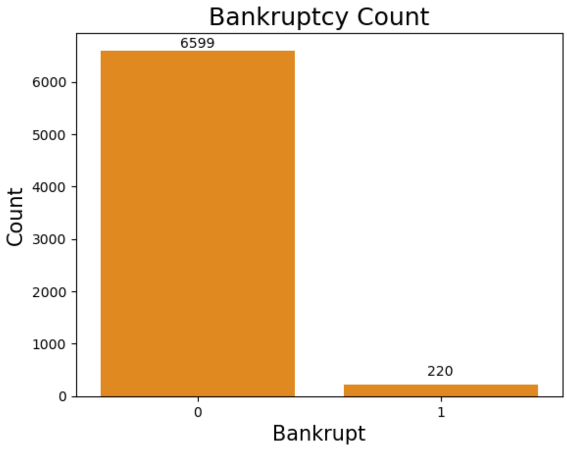


Figure 1 The distribution of the target

2.2 Anomaly Detection

The feature "Net Income Flag" in the dataset only has one single value of 1, indicating this feature does not provide any useful information for the classification problem. Therefore, it has been removed. In addition, 24 of the remaining 95 features have very extreme outliers. After analysis, the values of most features range from 0 to 1, but some of the values of these features are at least on the order of 10^8, indicating that these values are outliers. The boxplot (Figure 2) below illustrates the distribution of these 24 features.

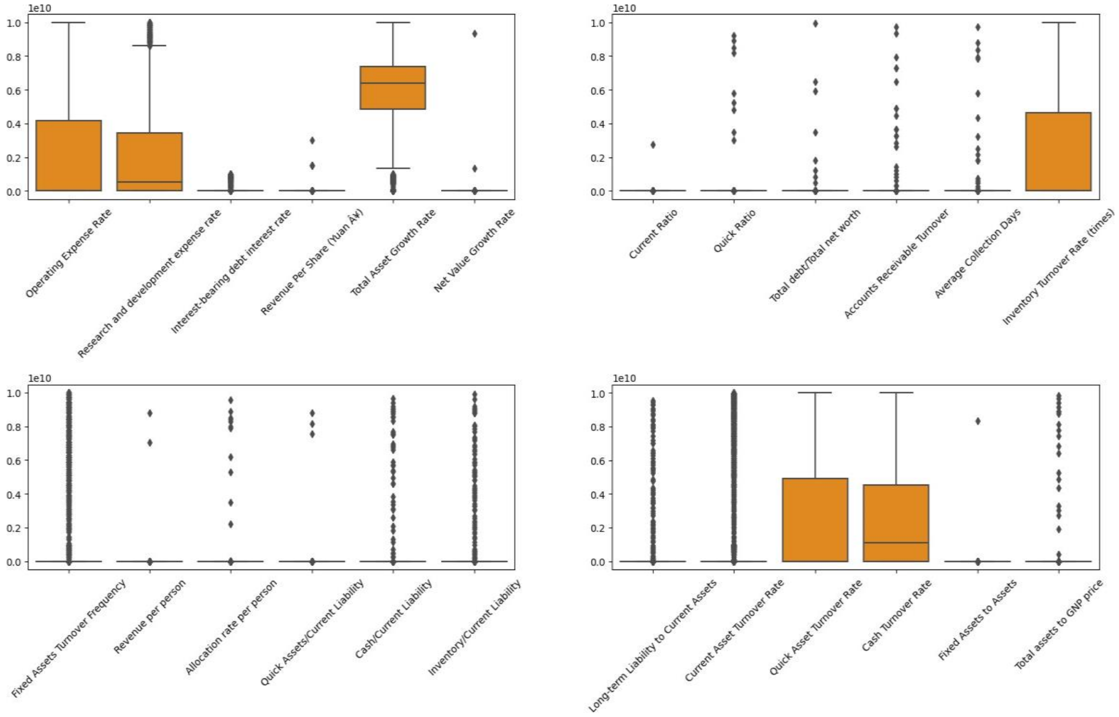


Figure 2 The boxplot of the 24 features with extreme outliers

According to the frequency table (Table 1), it can be found that among these 24 features, some features have fewer outliers, while others have more outliers.

In this study, Local Outlier Factor (LOF) was initially attempted to eliminate outliers, but it was found that some extreme outliers persisted. Furthermore, the application of LOF led to the removal of 47 bankruptcy samples, which was extremely unfavorable to the analysis for this study.

Table 1 Frequency table of the 24 features with extreme outliers

	non-bankrupt	bankrupt	sum
Operating Expense Rate	2215	67	2282
Research and development expense rate	3913	120	4033
Interest-bearing debt interest rate	219	2	221
Revenue Per Share (Yuan Â¥)	5	0	5
Total Asset Growth Rate	5805	212	6017
Net Value Growth Rate	1	1	2
Current Ratio	1	0	1
Quick Ratio	8	1	9
Total debt/Total net worth	7	1	8
Accounts Receivable Turnover	21	1	22
Average Collection Days	17	1	18
Inventory Turnover Rate (times)	2828	118	2946
Fixed Assets Turnover Frequency	1144	84	1228
Revenue per person	1	1	2
Allocation rate per person	11	1	12
Quick Assets/Current Liability	3	0	3
Cash/Current Liability	31	15	46
Inventory/Current Liability	96	3	99
Long-term Liability to Current Assets	107	2	109
Current Asset Turnover Rate	1181	53	1234
Quick Asset Turnover Rate	2286	97	2383
Cash Turnover Rate	4055	184	4239
Fixed Assets to Assets	0	1	1
Total assets to GNP price	17	3	20

Through the statistical table (Table 1) of the 24 features, it can be observed that only 8 features have more than 1000 outliers, and they also exhibit a considerable number of bankruptcy labels. In contrast, the remaining 16 features have only a few outliers, with the cumulative number of bankruptcy labels reaching a maximum of 33. Therefore, we attempted to directly remove features with more than 1000 outliers and then eliminate the remaining outlier samples. With this manual processing, only 29 bankruptcy samples were eventually removed. In summary, we choose the data processed manually as the basis for subsequent research.

2.3 Data Balancing

Before selecting a dimensionality reduction method, an examination of the feature correlations was conducted. As there are qualitative features (binary, not requiring explicit encoding), this study employed the Spearman correlation coefficient to assess the rank correlation between features. The heatmap (Figure 3) indicates that a majority of features exhibit significant correlations.

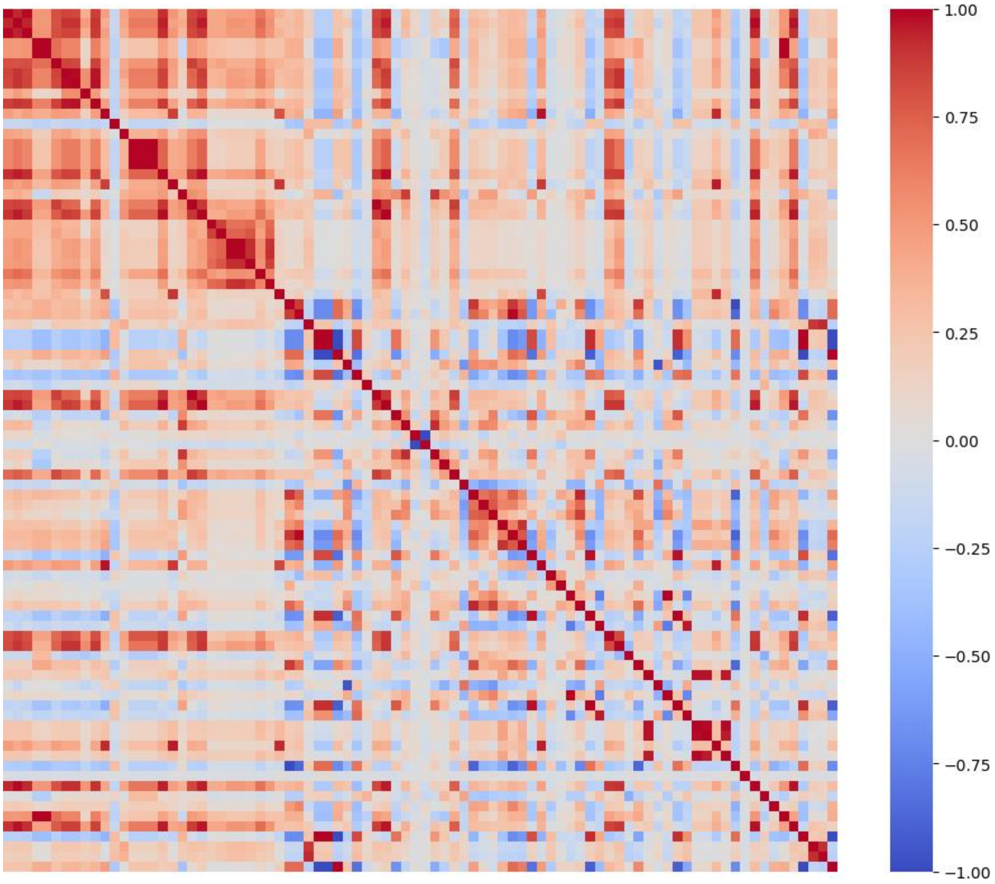
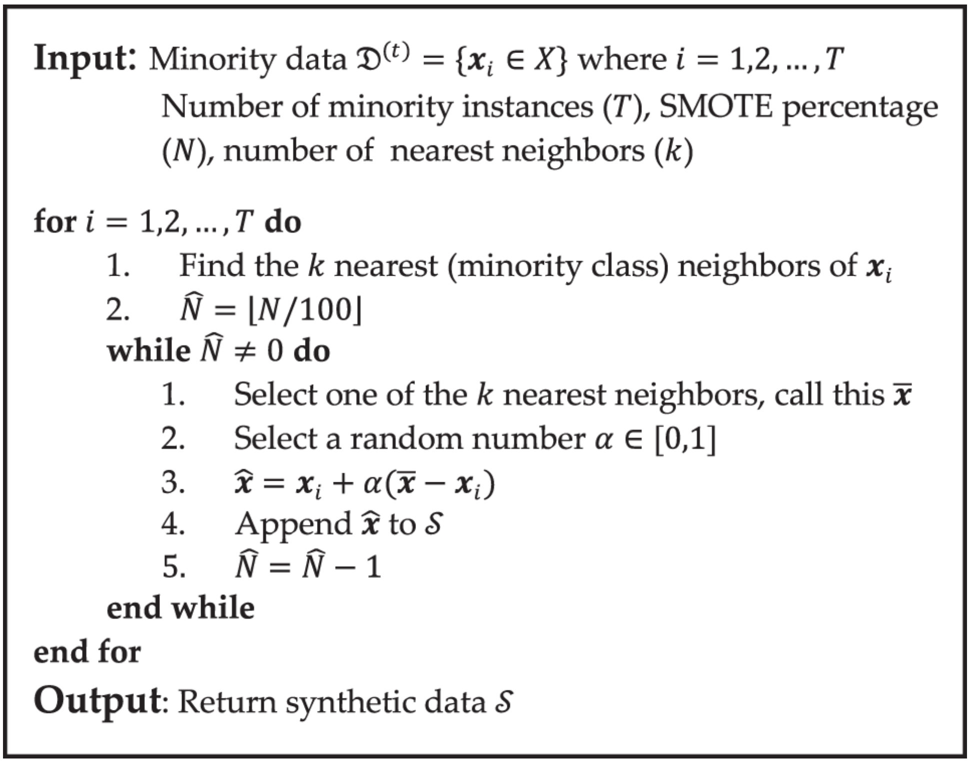


Figure 3 The heatmap of the correlation between features

Firstly, the process involves standardizing the data to eliminate differences in the range of values between features. Subsequently, the standardized data is used for SMOTE oversampling. The standardization step aims to address variations in the value ranges among features, ensuring a more reasonable outcome for SMOTE. [2] (It's worth noting that in this particular case, where the data ranges from 0 to 1, the impact of standardization might be minimal, but for the sake of procedural consistency, standardization is carried out.)

SMOTE (Synthetic Minority Oversampling Technique), which is based on a random oversampling algorithm is an improved scheme, due to the random oversampling to take a simple copy of the sample strategy to increase the minority class samples, which is prone to model overfitting, SMOTE algorithm basic idea is to analyze the minority class samples and manually synthesize new samples according to the minority class samples to add to the dataset, the algorithm flow is as follows.



```
Input: Minority data $\mathfrak{D}^{(t)} = \{\boldsymbol{x}_i \in X\}$ where $i = 1,2,\dots,T$
       Number of minority instances ($T$), SMOTE percentage
       ($N$), number of  nearest neighbors ($k$)

for $i = 1,2,\dots,T$ do
    1.  Find the $k$ nearest (minority class) neighbors of $\boldsymbol{x}_i$
    2.  $\widehat{N} = \lfloor N/100 \rfloor$
    while $\widehat{N} \neq 0$ do
        1.  Select one of the $k$ nearest neighbors, call this $\overline{\boldsymbol{x}}$
        2.  Select a random number $\alpha \in [0,1]$
        3.  $\widehat{\boldsymbol{x}} = \boldsymbol{x}_i + \alpha(\overline{\boldsymbol{x}} - \boldsymbol{x}_i)$
        4.  Append $\widehat{\boldsymbol{x}}$ to $\mathcal{S}$
        5.  $\widehat{N} = \widehat{N} - 1$
    end while
end for
Output: Return synthetic data $\mathcal{S}$
```

Figure 4 the algorithm flow of SMOTE

To streamline these four steps, a pipeline is constructed for ease of subsequent operations.[3] This study employs stratified 5-fold cross-validation to fit the aforementioned pipeline.

2.4 Dimensionality Reduction

For logistic regression, coefficient estimates in logistic regression become unstable due to multicollinearity. This instability arises from the matrix approaching singularity in the inverse process. In the coefficient estimation formula of logistic regression $(X^T X)^{-1} X^T Y$, when the matrix X is close to singularity, small data changes can lead to large fluctuations in the estimated coefficients, triggering instability.

Therefore, we use Elastic Net regression algorithm for model selection. The cost function w of Elastic Net regression algorithm

combines the regularization methods of Lasso regression and Ridge regression by controlling the size of the penalty term through two parameters, λ and ρ. The cost function w is as follows.

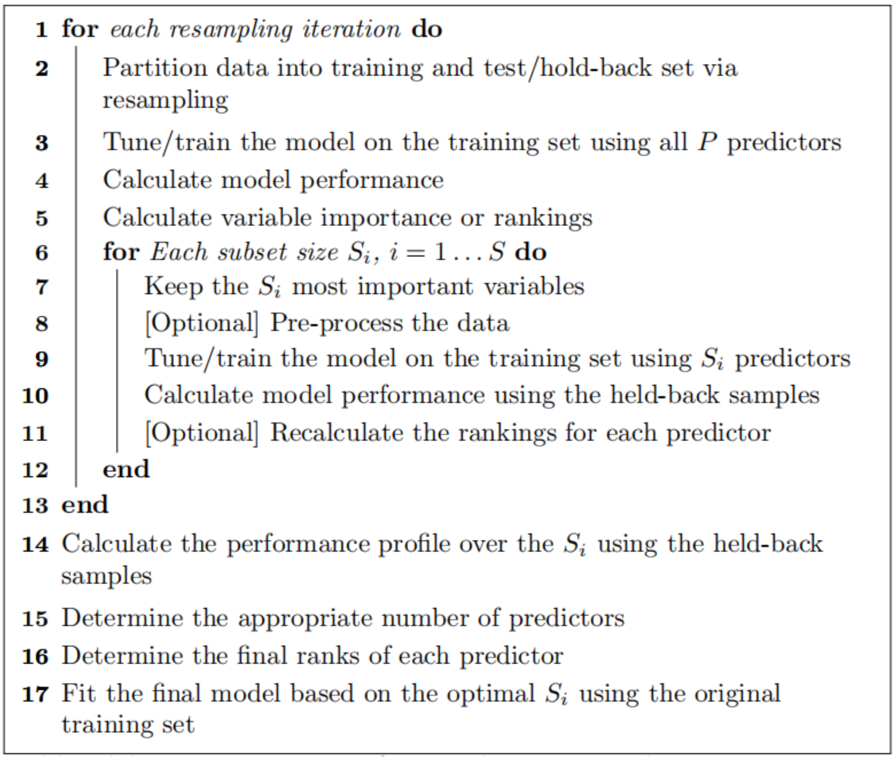
$$w = \underset{w}{\operatorname{argmin}}(\sum_{i=1}^{N}(y_i - w^T x_i)^2 + \lambda\rho\|w\|_1 + \frac{\lambda(1-\rho)}{2}\|w\|_2^2) \tag{1}$$

When $\rho = 0$, its cost function is equivalent to that of the ridge regression, and when $\rho = 1$, its cost function is equivalent to that of the Lasso regression.

Decision trees, random forests and XGBoost do not require a specific variable selection method, as the tree model itself has a built-in variable selection feature mechanism that automatically selects the most important variables through the tree splitting process.

In the case of SVM, the presence of noises in high dimensional data may lead to overfitting of the model, poor generalization, and the inability to add L1 penalties for variable selection as in other models. Therefore, we use RFE for model selection.

RFE (Recursive Feature Elimination) is a backward selection method that starts with all features and then recursively removes the least important features based on model performance. The performance of the model is evaluated using cross-validation techniques. The RFE method provides feature ranking based on the importance of the features, and the top features can be selected to build the final model.



```
for each resampling iteration do
    Partition data into training and test/hold-back set via resampling
    Tune/train the model on the training set using all $P$ predictors
    Calculate model performance
    Calculate variable importance or rankings
    for Each subset size $S_i$, $i = 1 \ldots S$ do
        Keep the $S_i$ most important variables
        [Optional] Pre-process the data
        Tune/train the model on the training set using $S_i$ predictors
        Calculate model performance using the held-back samples
        [Optional] Recalculate the rankings for each predictor
    end
end
Calculate the performance profile over the $S_i$ using the held-back samples
Determine the appropriate number of predictors
Determine the final ranks of each predictor
Fit the final model based on the optimal $S_i$ using the original training set
```

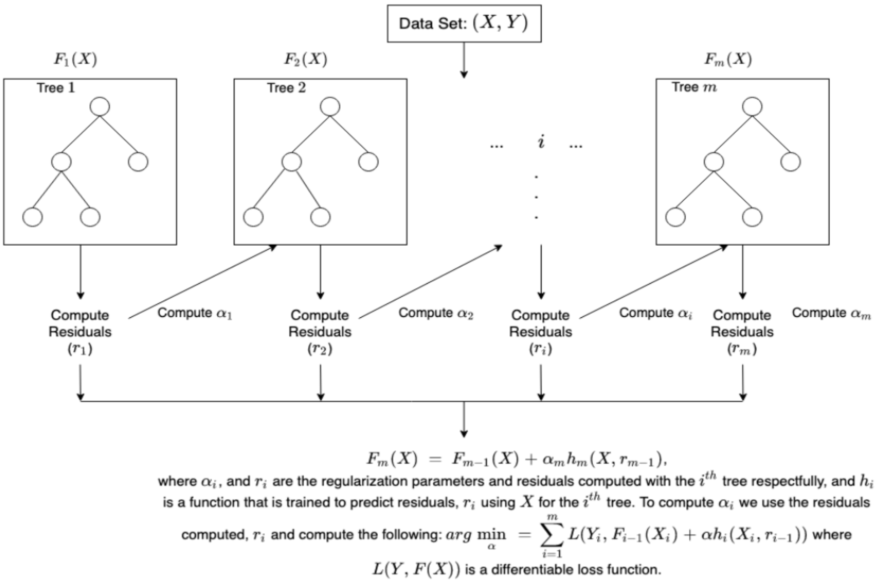
Figure 5 the algorithm flow of RFE

2.5 Method introduction

XGBoost is a popular and efficient open-source implementation of the gradient boosted trees algorithm. Gradient boosting is a supervised learning algorithm, which attempts to accurately predict a target variable by combining the estimates of a set of simpler, weaker models.

When using gradient boosting for regression, the weak learners are regression trees, and each regression tree maps an input data point to one of its leafs that contains a continuous score. XGBoost minimizes a regularized (L1 and L2) objective function that combines a convex loss function (based on the difference between the predicted and target outputs) and a penalty term for model complexity (in other words, the regression tree functions). The training proceeds iteratively, adding new trees that predict the residuals or errors of prior trees that are then combined with previous trees to make the final prediction. It's called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

Below is a brief illustration on how gradient tree boosting works.



$$F_m(X) = F_{m-1}(X) + \alpha_m h_m(X, r_{m-1}),$$

where α_i, and r_i are the regularization parameters and residuals computed with the i^{th} tree respectfully, and h_i is a function that is trained to predict residuals, r_i using X for the i^{th} tree. To compute α_i we use the residuals computed, r_i and compute the following: $arg \min_{\alpha} = \sum_{i=1}^{m} L(Y_i, F_{i-1}(X_i) + \alpha h_i(X_i, r_{i-1}))$ where $L(Y, F(X))$ is a differentiable loss function.

Figure 6 how gradient tree boosting works

2.6 Evaluation metrics

This article discusses the bankruptcy prediction problem of Taiwanese companies, which is a classification problem involving imbalanced samples. Therefore, traditional classification performance metrics such as Accuracy may be severely distorted. For instance, assuming there are 10 positive samples and 990 negative samples in the data, if a model predicts all samples as positive, the Accuracy of the model could be as high as 99%. However, this model would entirely fail to meet the requirements of bankruptcy prediction.

Hence, this study introduces metrics like Recall, Precision, F-score, and PR Curve to evaluate the model performance. These metrics provide a more rational and insightful assessment of the model's ability to predict bankruptcies, a data imbalance classification problem.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$F_\beta = \frac{(\beta^2 + 1) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{4}$$

Specifically, in the bankruptcy prediction case, "Positive" is defined as bankruptcy samples, while "Negative" refers to non-bankruptcy samples. Recall measures the probability of correctly predicting positive samples, with a higher value indicating a stronger ability of the model to correctly identify positive samples. Precision represents the ratio of true positive samples among those predicted as positive, and a higher precision implies a lower likelihood of misclassifying negative samples.

The F_β score can be viewed as a weighted average of Recall and Precision, with values ranging from 0 to 1. A higher F_β score indicates better overall predictive performance of the model. In the F_β score, the β value determines the weight of Recall in this score. In bankruptcy prediction problems, prioritizing Recall over Precision is often appropriate because misclassifying non-bankruptcy (negative) as bankruptcy (positive) is more acceptable than failing to identify companies that may be labeled as bankruptcy (positive).[4]

In this paper, F2 score will be the primary metric to assess the performance of the company bankruptcy prediction model, as it places more emphasis on Recall over Precision.

Additionally, this paper calculates the Confusion Matrix, Precision, Recall, F1 Score, Type I/II Error, Average Precision, and ROC AUC for the 5-fold cross-validation. Furthermore, Precision-Recall Curve (PR Curve) and Receiver Operating Characteristic Curve (ROC Curve) are plotted with cross- validation to provide a visual representation of the model's performance across different thresholds.[5] [6]

3 Experiments

3.1 Model Training

In the model training phase, this study utilizes 5-fold cross-validation and a pipeline to train five different models: logistic regression, decision tree, random forest, support vector machine, and XGBoost. The pipeline follows the sequence: Standardization -> Data Balancing -> Dimensionality Reduction -> Model Training.[7]

For the first four models (logistic regression, decision tree, random forest, and support vector machine), GridSearchCV [8] is employed for an exhaustive search over hyperparameters to optimize F2 score, while XGBoost uses Bayesian optimization over hyperparameters, which is more efficient.[9]

3.2 Result Analysis

3.2.1 Simulation Analysis

The process for computing the confusion matrix is as follows: (1) Predict the probability of labels for each test set in each fold; (2) Stack the predicted probability values of 5 folds together; (3) Use a probability threshold of 0.5 to predict binary labels; (4) Calculate the confusion matrix based on the predicted labels.

The rationale behind this calculation is: (1) Each sample is predicted only once, avoiding duplicates; (2) The confusion matrix calculated in this way is equivalent to the accumulation of all confusion matrices for 5 folds; (3) It provides an overview of the overall misclassification patterns of the model.

The results are shown below:

Table 2 The confusion matrix of Logistic regression

	Predicted Negative	Predicted Positive
Actual Negative	5169	921
Actual Positive	25	166

Table 3 The confusion matrix of Decision Tree

	Predicted Negative	Predicted Positive
Actual Negative	5304	786
Actual Positive	56	135

Table 4 The confusion matrix of Random Forest

	Predicted Negative	Predicted Positive
Actual Negative	5562	528
Actual Positive	59	132

Table 5 The confusion matrix of SVM

	Predicted Negative	Predicted Positive
Actual Negative	5234	856
Actual Positive	43	148

Table 6 The confusion matrix of XGBoost

	Predicted Negative	Predicted Positive
Actual Negative	5537	553
Actual Positive	50	141

It appears that Logistic regression achieved the highest accuracy in predicting positive samples, followed by SVM and XGBoost. However, when considering misclassifications of negative samples, XGBoost made the fewest errors among the three models, followed by XGBoost. This suggests that SVM may have a better overall performance in terms of balancing both positive and negative predictions.

The steps for drawing the PR Curve (ROC Curve) with cross-validation [10] are as follows: (1) Calculate Precision and Recall (TPR and FPR) for each fold; (2) Draw the PR Curve (ROC Curve) for each fold; (3) Perform linear interpolation on Precision (TPR) for each fold; (4) Calculate the mean and standard deviation of Precision (TPR) lists across all folds; (5) Draw the Mean PR Curve (ROC Curve).

Note: The Average Precision (AUC) values obtained using this method may differ from those calculated using *cross_validate* or *cross_val_score* in *sklearn*. This discrepancy arises because *cross_validate* or *cross_val_score* directly averages the metrics across k-folds, whereas this method does not. For most models, this approach is reasonable, except for a single classification tree. This is because it outputs a limited number of countable probabilities for all samples, leading to only a few points that can be plotted on the figure. Consequently, the PR curve and ROC curve may not be accurate for a single classification tree.

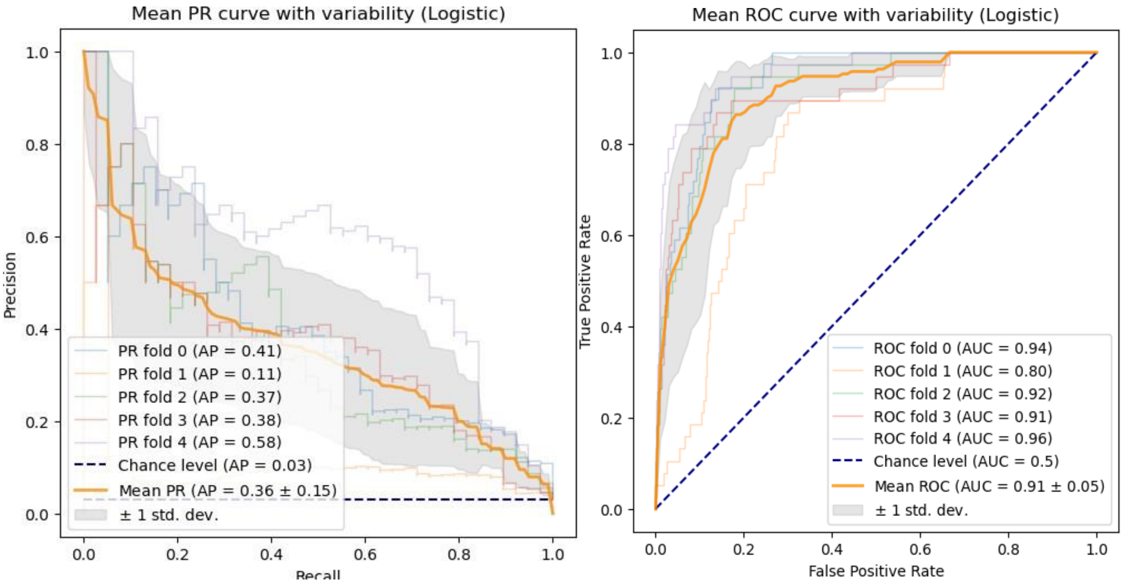


Figure 7 The PR curve and ROC curve of Logistic regression

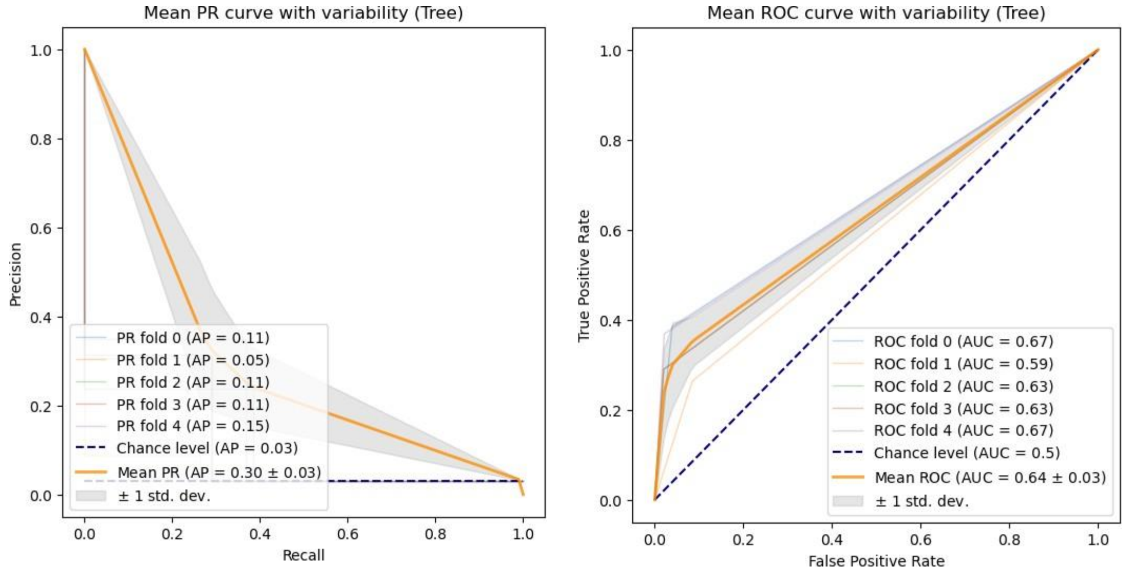


Figure 8 The PR curve and ROC curve of Decision Tree

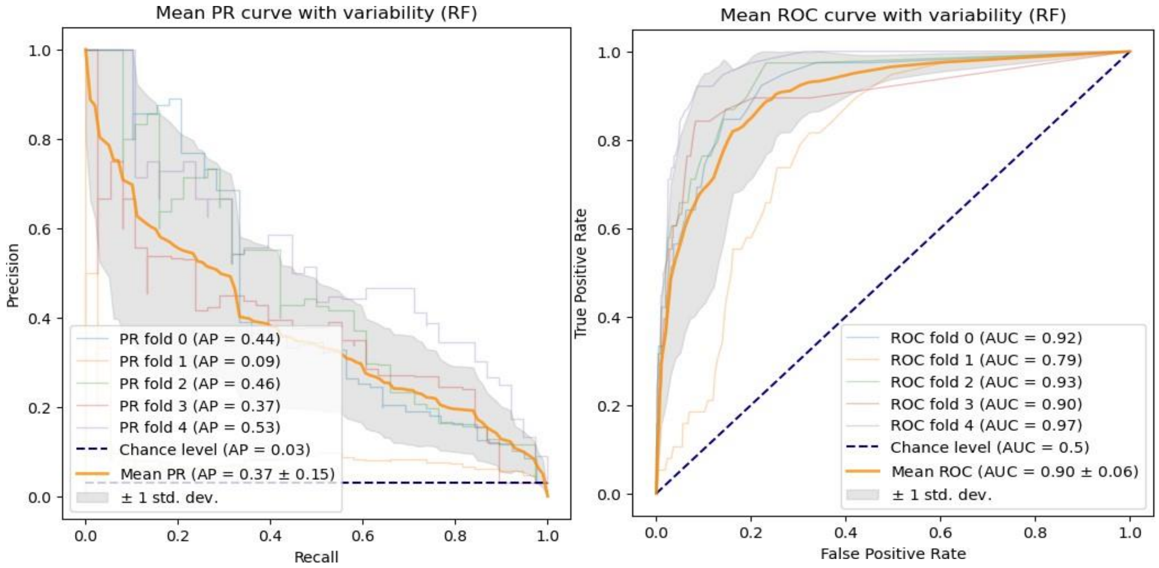


Figure 9 The PR curve and ROC curve of Random Forest

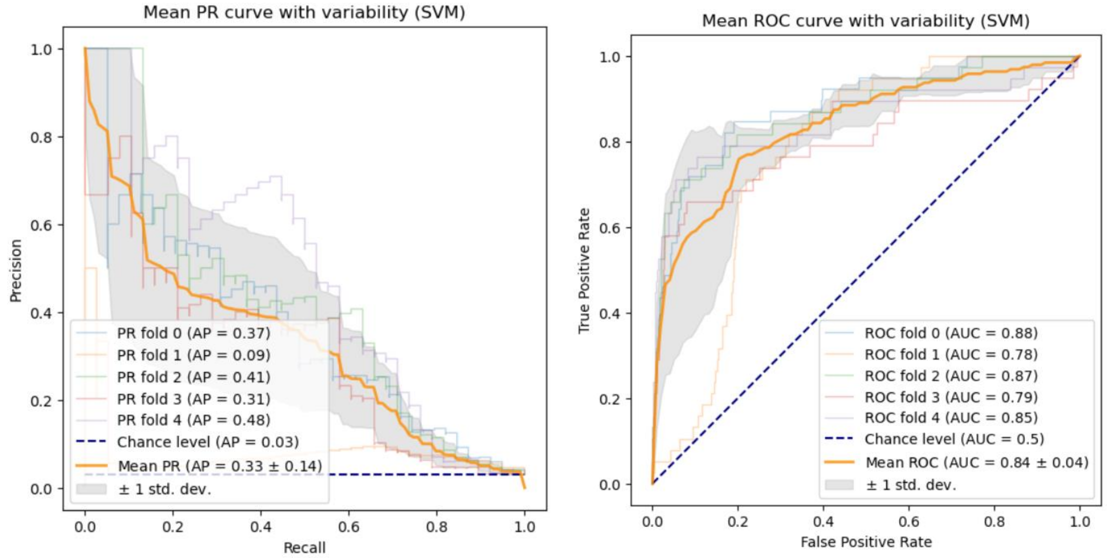


Figure 10 The PR curve and ROC curve of SVM

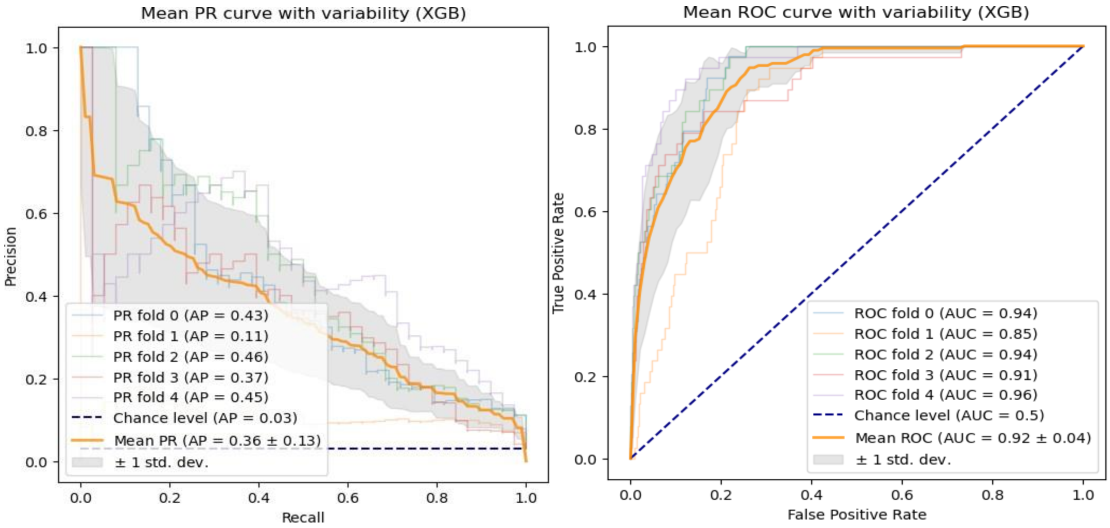


Figure 11 The PR curve and ROC curve of XGBoost

Table 7 The performance metrics of the five models

	Logistic	Tree	RF	SVM	XGB
Precision	0.1694	0.1631	0.2242	0.1715	0.2337
Recall	0.8687	0.7059	0.6904	0.8687	0.7378
F1	0.2800	0.2599	0.3315	0.2815	0.3463
F2	0.4647	0.4097	0.4734	0.4639	0.4974
AP	0.3808	0.1993	0.3456	0.3983	0.3506
AUC	0.9210	0.8098	0.9260	0.9198	0.9291
Type1_error	0.1512	0.1291	0.0867	0.1406	0.0908
Type2_error	0.1309	0.2932	0.3089	0.2251	0.2681

Preliminarily, it can be seen from the F2 score that XGBoost has the best model effect, while a single classification tree has the worst effect. From the perspective of AUC, except for the single classification tree model, the AUC of other models are relatively good, but their AP values are quite low, indicating that in the problem of imbalanced data sets, AUC will have a misleading effect, so AP is more convincing metric. (Note: Except for the decision tree model, the AP and AUC values for other models are based on the values shown in the figure.)

Among the five models, the Mean PR Curve of Random Forest has the highest AP value. XGBoost's Mean PR Curve has the second highest AP value with a smaller standard deviation, which indicates that XGBoost performs well in predicting this imbalance data. When we take a closer look at the Recall and Type I/II error of the models, it is obvious that XGBoost's Recall and Type II error performance are better than Random Forest, so in this case XGBoost is better than Random Forest.

However, XGBoost has a lower Recall and relatively higher Type II error compared to logistic regression, indicating that in terms of the efficiency of predicting True Positive, XGBoost may not be as effective as the traditional logistic regression.

3.2.2 Interpretive analysis

To further explore which variables are more important in the context of predicting whether a company is likely to go bankrupt, we use the metric "importance" to characterize the extent to which each variable contributes to the final projected results. The figures of feature importance of different models are as follows.

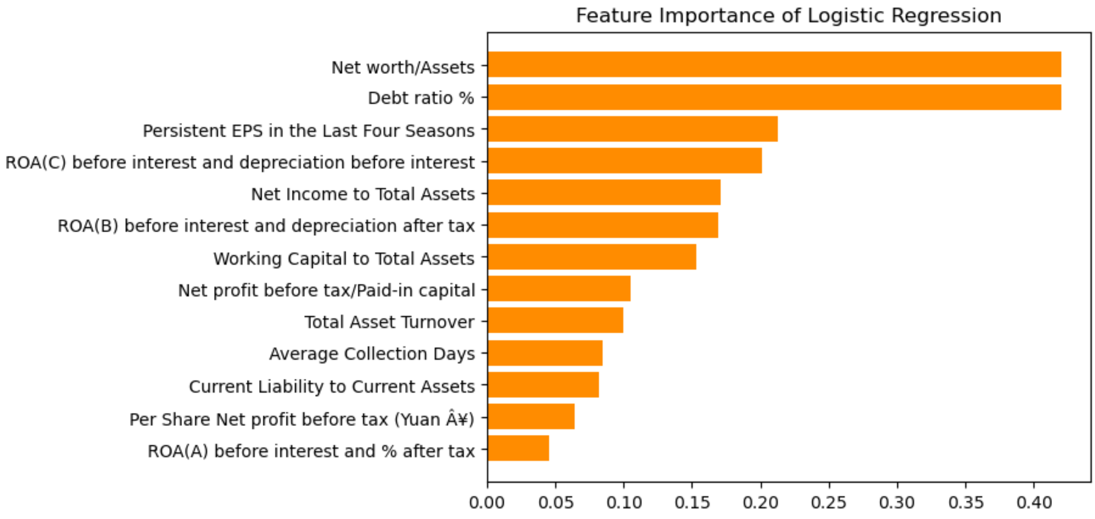


Figure 12 Feature Importance of Logistic Regression

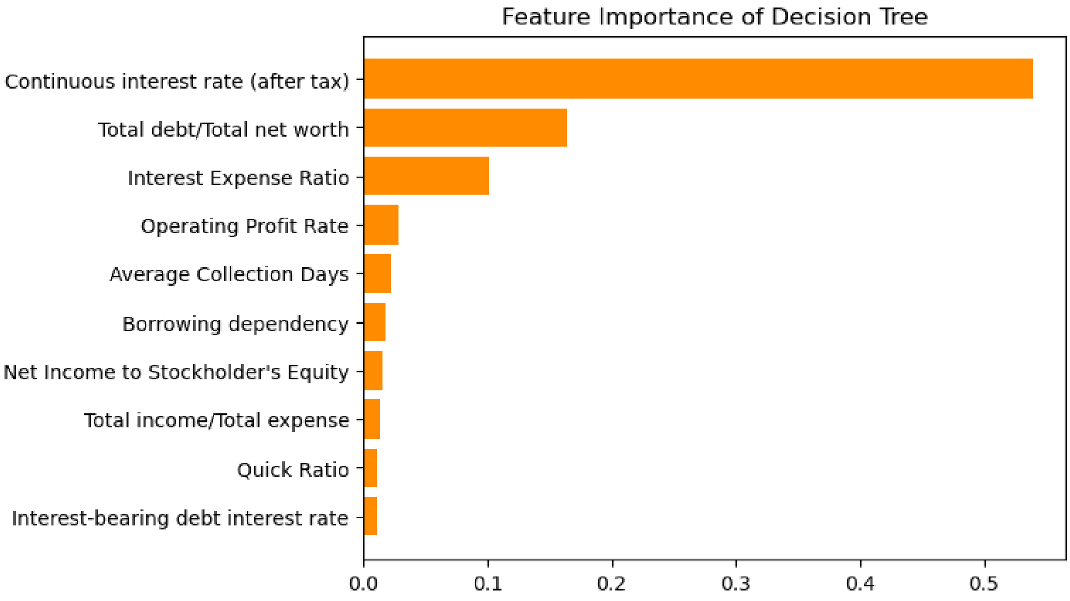


Figure 13 Feature Importance of Decision Tree

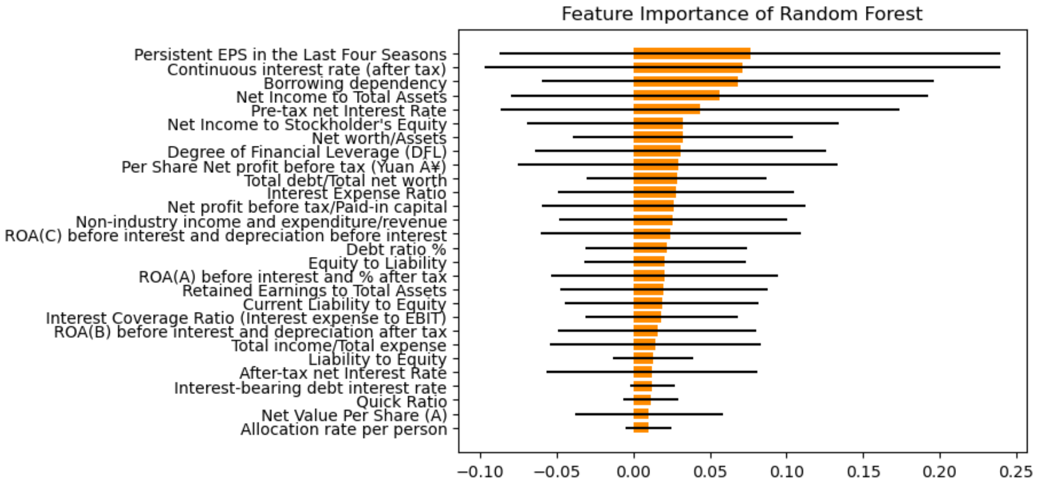


Figure 14 Feature Importance of Random Forest

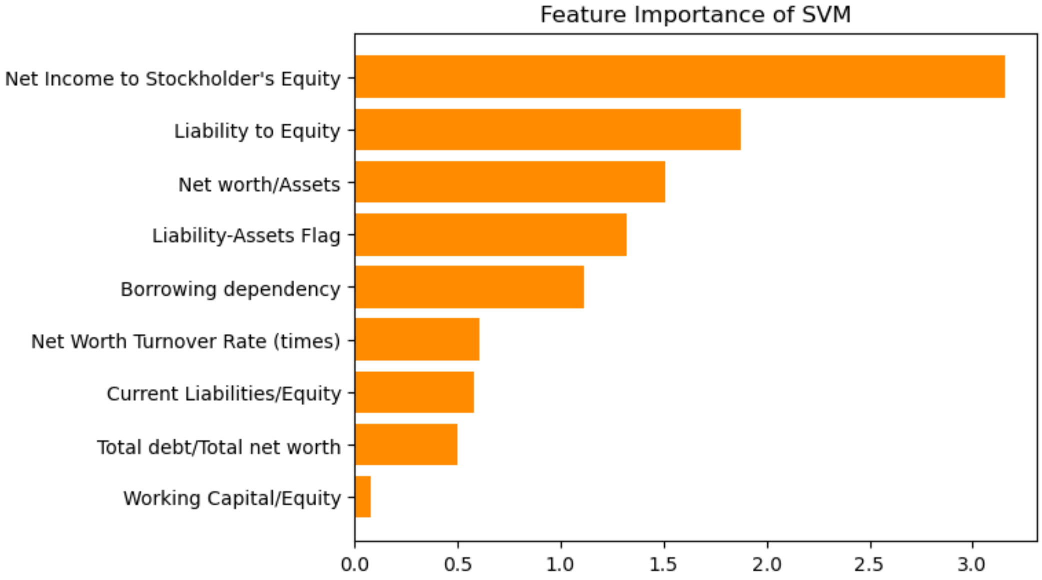


Figure 15 Feature Importance of SVM

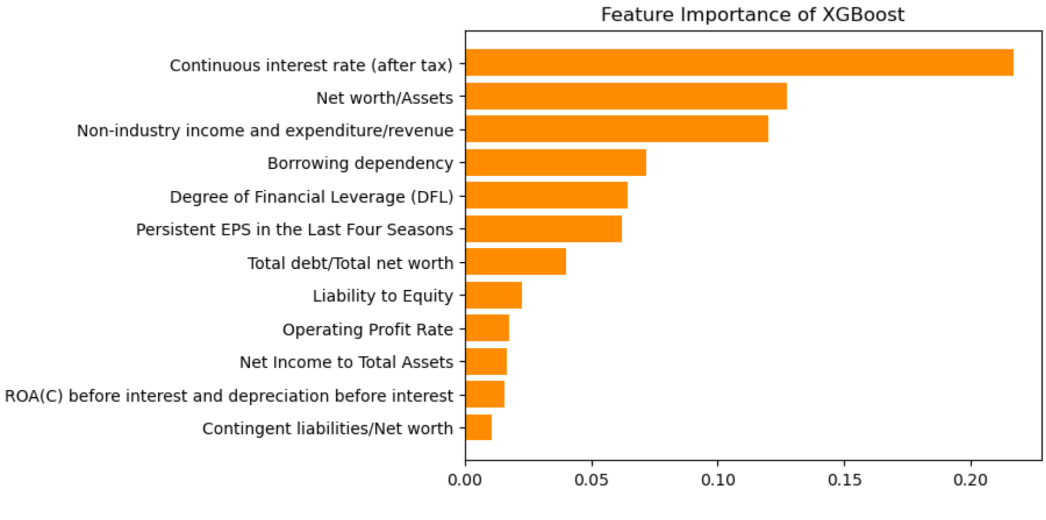


Figure 16 Feature Importance of XGBoost

We selected the six variables that appeared most frequently in the top-ten most important variables of the five models. Then we summarize them into two main categories, one is the financial soundness category, including continuous interest rate, net worth/assets and total debt/total net worth, and the other is the profitability category, including persistent EPS in the last four seasons, net income to total assets and borrowing dependency.

4 Conclusion

This study focuses on the performance of five common classifiers in bankruptcy prediction problem in Taiwan, and the results indicate that both Logistic regression and XGBoost exhibit outstanding model performance. Logistic regression demonstrates a stronger predictive ability for positive samples, while XGBoost, in addition to pursuing accurate predictions for positive samples, also exhibits superior Precision performance. The recommendation provided in this study is to choose between logistic regression and XGBoost based on specific requirements for model metrics in real- world business scenarios.

In the future, the research can be expanded to bankruptcy prediction problem in other regions, and it is expected to have certain reference significance for related research.

References

[1] https://www.graphpad.com/guides/prism/latest/statistics/stat_pca_graphs_tab.htm

[2] https://stats.stackexchange.com/questions/363312/normalization-standardization-should-one-do-this-before-oversampling-undersampl

[3] https://scikit-learn.org/stable/auto_examples/compose/plot_digits_pipe.html#sphx-glr-auto-examples-compose-plot-digits-pipe-py

[4] 李扬,李竟翔,马双鸽.不平衡数据的企业财务预警模型研究[J].数理统计与管理,2016,35(05):893-906.

[5] https://stats.stackexchange.com/questions/210700/how-to-choose-between-roc-auc-and-f1-score

[6] https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/

[7] https://scikit-learn.org/stable/auto_examples/compose/plot_compare_reduction.html#sphx-glr-auto-examples-compose-plot-compare-reduction-py

[8] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[9] https://towardsdatascience.com/optimizing-hyperparameters-the-right-way-3c9cafc279cc

[10] https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc_crossval.html