MACHINE LEARNING FOR IMBALANCED DATASET: PREDICTING COMPANY BANKRUPTCY

Abhinav Bajpai (abbajpai@iu.edu), Saranjeet Singh Saluja (sasalu@iu.edu)
Adit Sadiwala (asadiwal@iu.edu)

Department of Data Science
Indiana University - Bloomington

ABSTRACT

When modeling credit risk, a major concern is the class imbalance in the dataset used. In the case of a dataset with default or no-default assignment for companies, there is class imbalance when one class is considerably larger than the other. Generally, we observe fewer default class assignments because money lending occurs following proper due diligence and review of past performance. However, defaults do happen as external factors or the company's own performance deteriorates and hence the need for models to anticipate future default scenarios arises. In this study, we review six modeling techniques - Logistic Regression, Under-sampling (Near Miss), Under-sampling (Cluster Centroid), Grabit Model, Thresholding, and Extreme Machine Learning for handling class imbalanced data and compare their performance using the Polish companies' bankruptcy data.

Index Terms— Imbalance Dataset, Credit Risk, Bankruptcy, Default

1. INTRODUCTION

There have been financial meltdowns in the past, such as the 2008 financial crisis or the 2019-2020 economic downturn caused by pandemics which forced hundreds of companies into bankruptcy. Credit risk analysis of companies seeking debt or equity has become an industry in the past due to such incidents. There are multiple credit risk rating agencies that provide default probabilities and credit ratings to establish a company's creditworthiness. Financial institutions such as banks rely on sophisticated statistical models to analyze the financial performance of the companies to which they lend their money. They also adjust interest rates based on these analyses. The performance of past established methodologies can be enhanced using machine learning algorithms, enabling better credit judgment. As both firm-specific and macroeconomic factors influence company performance, credit risk analysis is a complex activity. In addition, the credit risk industry is well regulated, and models that approve/disapprove credit need approval by regulators. In the aftermath of 2008, regulatory authorities were very cautious about approving more sophisticated credit approval models. Credit risk modeling has largely relied on simpler and more explainable models, such as Logistic Regression, Time Series using ARIMA, and Linear Regression. Recently, we saw a shift to accepting more advanced machine learning techniques as they become more explainable using newer visualization tools and quality metrics (ex: bias measurement using fairness metrics). This shift has also allowed for the use of newer methodologies to deal with imbalanced datasets, which are encountered more often in credit risk modeling. As part of our study, we examine the current landscape of methods available to handle imbalance data, and we compare the effectiveness of these methods in accurately classifying default events.

2. RELATED WORK

Over the last decade, machine learning researchers have extensively studied the imbalanced dataset problem. Bankruptcy or default by companies offer real world imbalanced data for study because defaults are rare unless there is severe economic downturn. Sigrist and Hirnschall [1] in their study suggest a new binary classification model called the Grabit model, which is derived by applying gradient tree boosting to the Tobit model. The Tobit model is based on assumption that there exists a latent variable $Y*$ which follows a Gaussian distribution conditional on some covariates $X = (X1, . . ., Xp)^T \in \mathbb{R}p$,: $Y*|X \sim N(F(X), \sigma2)$ such that F(X) depends linearly on X. The Grabit model suggested by the authors replace the function F(X) with a more flexible function $F(\cdot) : \mathbb{R}p \rightarrow \mathbb{R}$ consisting of an ensemble of regression trees. The function is estimated by applying boosting with regression trees as base learners using the negative log-likelihood of the Tobit model as loss function L(y, F). To demonstrate improved performance over other machine learning techniques, the authors then applied the Grabit model to predict defaults on loans made to Swiss small and medium-sized enterprises (SME).

Similarly, Hengyu [2] propose a new cost sensitive Support Vector Machine (SVM) algorithm where cost weights of samples are first determined by class density using the Kernel Density Estimation (KDE) algorithm. The author then uses Support Vector Data Description (SVDD) algorithm to find out the center and radius of all minority (C_{min}, R_{min}) and majority (C_{max}, R_{max}) samples. Further utilizing the center and radius values, each sample is classified as S_{Normal}, S_{Border}, $S_{Overlap}$ and S_{Noise} and a distance based weighted scheme is then suggested for the SVM algorithm. In order to showcase the algorithm's better generalization ability, authors compare its performance against other variants of the SVM algorithm designed for imbalanced datasets.

Khoshgoftaa and Johnson [3] outline the importance of thresholding as a technique to improve classification results in imbalanced data. The probability estimates produced by classifier algorithms have thresholds that assign class labels. Changing the threshold used to assign class labels to probability estimates increases or decreases the bias towards a particular class. In the author's opinion, a 0.0 decision threshold can give a perfect true positive ratio (TPR), invalidating the model because it will certainly misclassify all negative samples as positive samples. As a result, selecting an optimal threshold will have to consider a tradeoff between class-wise performance scores. In their study, the authors propose a thresholding procedure to identify optimal classification thresholds on imbalanced data along with a demonstration of the relationship between minority class size and the optimal threshold. Their thresholding procedure is further evaluated by fitting a DNN model to a Medicare dataset with only 0.03% positive class in the dataset.

The use of sampling techniques to manage imbalanced data has also been a major focus of imbalanced data research. Talha Mahboob Alam et al. [4] explore Undersampling (Random undersampling, Near Miss, Cluster Centroid) and Oversampling (Random Oversampling, Adaptive Synthetic, SMOTE, SMOTE Tomek, and K-Means SMOTE) techniques through their study. After applying these sampling techniques to solve imbalanced data issues of 3 different credit datasets, the authors test multiple hypotheses through ANOVA. As a result, the authors show that machine learning algorithms such as Random Forest, Gradient Boosting Decision Tree, etc., demonstrate improved performance.

Zhang and Qin [5] proposes a novel variant of the Extreme Learning Machine (ELM) algorithm that employs one hidden layer feedforward neural network to reduce the effect of class imbalance. The proposed algorithm Output Weight Adjustment – Extreme Learning Machine (OWA- ELM) improves minority class classification accuracy by adjusting the connection weights between hidden layer neurons and minority output neurons in ELM. Specifically, authors add an empirically determined Δ value to one of the columns of the weight matrix that significantly improves the evaluation metric G-Mean and F measure. Using 22 imbalanced datasets, the authors demonstrate improved performance and suggest that the optimal range for Δ is between 0.01 and 0.02.

3. DATA

3.1 Data Description

The sample dataset about bankrupt and operating Polish companies was published on the UCI machine learning repository by Zieba and Tomczak [6] . Approximately 10k companies' financial ratios were collected for the period 2000-2012 and default information is provided over multiple forecasting time periods – 1, 2, 3, 4 and 5 years.

The 1st Year data contains financial ratios for the first year of the forecasting period, along with a class label that indicates whether bankruptcy has occurred after five years. Similarly, the time period of bankruptcy is defined for other datasets in the table below.

Dataset Name	Bankrupt Firms	Operating Firms	Time Horizon to Bankruptcy
1 Year	271	6756	5
2 Year	400	9773	4
3 Year	495	10008	3
4 Year	515	9277	2
5 Year	410	5500	1

Table 1: Polish Bankruptcy Datasets

The datasets showcase skewed distribution of default (minority) and non-default (majority) classes. In all the five dataset the % of default class ranges between 3.9% to 6.9%. For dataset 1 Year and 2 Year (bankruptcy in 5 and 4 year), we observe the lowest percentage (3.9%) of firms that had defaulted as compared to dataset 5 Year (bankruptcy in 1 year) where 6.9 % of firms defaulted in the collected sample.

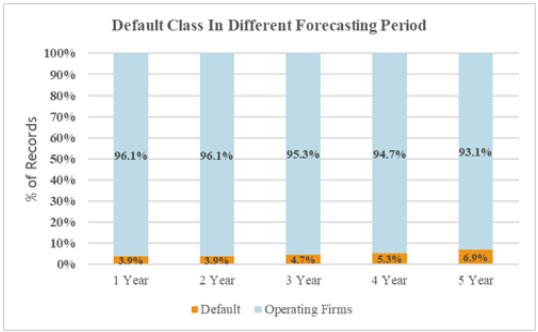

Figure 1: Default % In 5 Datasets

Each dataset contains 64 attributes, which are financial ratios derived from the financial statements of the company and show the company's financial health and performance. A complete list of the financial ratio is provided in the Appendix.

For each dataset, all attributes with more than 10% missing data were dropped, and the remaining attributes were imputed using the median and scaling was performed using Robust Scaler.

4. METHODS

4.1. Logistic Regression

In the case of a logistic regression, the probability of default, π is assumed to be a linear function of the observed covariates, i.e.

$$\pi = P(Y = 1|X = x) = \frac{exp(\beta' x)}{1 + exp(\beta' x)}$$

where $x = (1, x_1, ::: , x_p)$ is the design matrix and $\beta = (\beta_0, \beta_1, \beta_2, \beta_3 \ldots \beta p)$ is the (p+1)-dimensional vector of regression parameters. The decision boundary is linear via the linear predictor $lp = \beta x$. Thus, $lp > 0$ (equivalently $\{x: \hat{\pi} > .5\}$ where $\hat{\pi}$ is the predicted probability) is classified as 1 and 0 otherwise.

We use the Logistic Regression model as our baseline model and experiment with other techniques to see if they outperform the base model.

4.2 Under-Sampling

During an imbalanced classification task, resampling methods are used to change the composition of a training dataset. In order to balance the class distribution, undersampling techniques remove examples from the training dataset that belong to the majority class. For example, a 1:100 class distribution may be reduced to a 1:10, 1:2, or even a 1:1 distribution. The two undersampling techniques we used in our study are described below:

4.2.1 Near Miss

To balance the dataset, under-sampling was performed using Near Miss [7]. The Near Miss under-sampling technique has three variants:

1. **Near Miss 1:** Selection of Majority class point with the smallest average distance from the three nearest points of the minority class
2. **Near Miss 2:** Selection of Majority class point with the smallest average distance from the three furthest points of the minority class; and
3. **Near Miss 3**: Selection of Majority class point with the smallest average distance from each point of the minority class

For this study, we have used the Near Miss 1 algorithm. Post under-sampling with Near Miss 1, a Random Forest Classifier model was applied to all forecasting time period datasets – 1, 2, 3, 4, and 5 years.

4.2.2 Cluster Centroids

The Cluster Centroids [8] technique is another under-sampling technique for handling imbalanced data. The K-Means algorithm is used to under-sample by substituting the majority class points with the cluster centroids. Using the cluster centroids as the majority class points, this technique reduces information loss. Following cluster centroids under-sampling, a Random Forest Classifier model was used to forecast 1-, 2-, 3-, 4- and 5-year time periods.

4.3 Grabit Model

In the Tobit model, censoring mechanisms are used to define data. However, Tobit is not only applicable to censored data, but also to data that comprises continuous sections with discrete parts at the borders. Examples include fractional response data, such as Loss given default, rainfall, or default prediction. The Tobit model is based on assumption that there exists a latent variable $Y*$ in an interval $[y_l, y_u]$, which follows a Gaussian distribution conditional on some covariates $X = (X1, . . ., Xp)T \in \mathbb{R}p$,: $Y*|X \sim N(F(X), \sigma 2)$ such that F(X) depends linearly on X. We can interpret Y* as the default potential, and a default occurs when the potential $Y*$ exceeds a certain threshold. Therefore, default events correspond to the case $Y* \geq y_u$, and the observed data is identified as $Y* = y_u$. Non-default values correspond to $Y* < y_u$, and the auxiliary variable is identified as Y = Y* in this case. The Tobit loss function for $Y* \geq y_u$ is given by the following formula: $L(y, F) = -\log(f_{F,\sigma}(y))$ which is asymmetric in the predictor function F(X): thus, the larger the default potential, the smaller the loss and the lower the default potential, the greater the loss. This leads to the desired behavior of predicting a higher default probability for cases that default, rather than penalizing predictions above the threshold. Since the Tobit model only considers linear dependences on features, it is unable to learn complex nonlinear relationships in data. A gradient tree boosting approach is applied to the Tobit model to overcome this challenge and better classification results are obtained.

4.4 Thresholding

Decision threshold, discrimination threshold, or simply threshold governs the conversion of a predicted probability into a class label. In the case of normalized predicted

probabilities or scores in the range of 0 to 1, the threshold value is set to 0.5. For instance, if there are two classes 0 and 1, normalized predicted probabilities and a threshold of 0.5, values less than 0.5 are assigned to class 0 and values greater than or equal to 0.5 are assigned to class 1. For imbalanced dataset this threshold of .5 can be moved to optimize the evaluation matrix. In our study, we fitted a Deep Neural Network model on the training datasets, followed by a prediction probability calculation on the test dataset. We then experiment with different threshold value settings through a grid search that maximizes the F1 score. The resulting labels and metrics are used for comparison with other classifiers.

4.5 Extreme Machine Learning

Extreme learning machine (ELM) is a single hidden layer feedforward neural network. When we initialize input weights and biases, ELM does not require us to adjust them for learning. This results in a very fast training time and high generalization. However, ELM does not perform well with imbalanced data. Therefore, we implemented Enhanced ELM with thresholding. We added delta after applying the activation function to enhanced ELM. Adding a very small delta to our neural network does not negatively affect it. For classification, we will be using the Moore-Penrose inverse matrix. To find the most optimal model parameters, we use thresholding to maximize F1-Score.

Evaluation Matrix

Classification accuracy is defined as the number of correct predictions divided by the total number of predictions. When the distribution of examples across classes is severely skewed, this intuitive metric breaks down and can be misleading. Therefore, we consider the performance using the following metrics:

1. **Precision** : Precision calculates the accuracy of the minority class. It is calculated by dividing correctly predicted positive examples by the number of positive examples that were predicted.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

2. **Recall :** The recall metric measures how many correct positive predictions were made out of all possible positive predictions. As opposed to precision, which only identifies the correct positive predictions out of all positive predictions, recall identifies the missed positive predictions. As a result, recall provides some information about how well a positive class has been covered.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

3. **F1 Score :** The goal of biased datasets is to increase recall without reducing precision. This can be challenging, since increasing recall often results in reduced precision. By using an F-measure, both precision and recall can be captured in one measurement and can help compare classifiers.

$$\text{F-measure} = \frac{2\ X\ Precision\ X\ Recall}{\text{Precision} + \text{Recall}}$$

Additionally, we provide the ROC Curve (Appendix) as an additional diagnostic tool to compare the different classifiers. Calculating the area under the curve (AUC) gives a single score for the classifier model across all threshold values. The score is a value between 0 and 1, where 1 indicates a perfectly accurate classifier.

5. RESULTS

5.1. Logistic Regression (Baseline)

The logistic regression on the imbalanced data does give a high accuracy, however, the precision and recall scores (and hence the F1 score) are very low. The worst performance is observed for the Year 3 dataset, where the F-1 Score is a mere 3%, implying that the model is unable to identify observations of minority class. The best performance is observed in the Year 5 dataset where a recall of 0.22 is achieved. Overall, the model has failed to identify the minority class adequately.

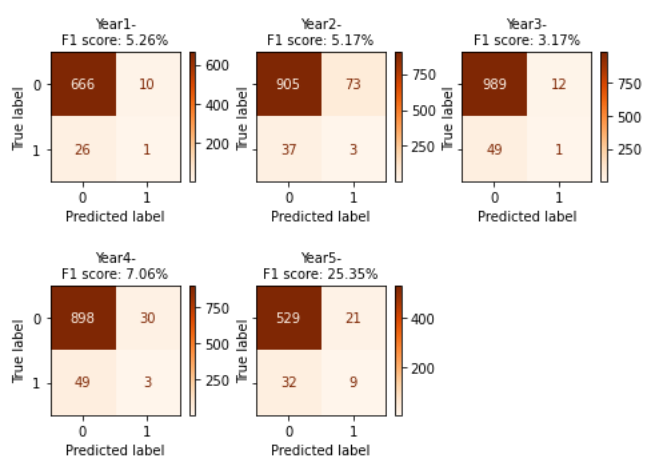

Figure 2: Logistic Regression Confusion Matrix

Dataset	Accuracy	Precision	Recall	F-1
Year 1	95%	9%	4%	5%
Year 2	89%	4%	8%	5%
Year 3	94%	8%	2%	3%

Dataset	Accuracy	Precision	Recall	F-1
Year 4	92%	9%	6%	7%
Year 5	91%	30%	22%	25%

Table 2: Logistic Regression Evaluation Metrics

5.2 Near Miss

The Near Miss under-sampling technique on the imbalanced data does give a higher recall value but this is achieved at the cost of lower precision score. The best performance of precision score .12 is observed in the Year 5 dataset. Overall, the model fails to classify both default and non-default events accurately.

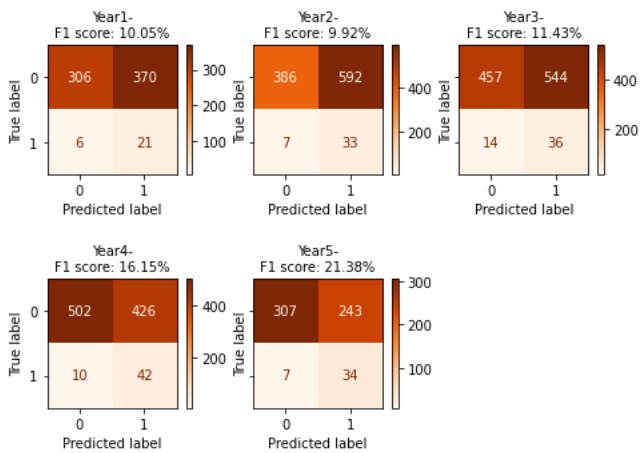

Figure 3: Under-Sampling **(**Near Miss) Confusion Matrix

Dataset	Accuracy	Precision	Recall	F-1
Year 1	47%	5%	78%	10%
Year 2	41%	5%	83%	10%
Year 3	47%	6%	72%	11%
Year 4	56%	9%	81%	16%
Year 5	58%	12%	83%	21%

Table 3: Under-Sampling **(**Near Miss) Evaluation Metrics

While the Near Miss algorithm performs reasonably well (in terms of recall) compared to the baseline model, it gives a lot of False Positives. Due to the bias introduced by the algorithm, which considers points close to the minority class, the model is unable to distinguish the minority class from the majority class.

5.3 Cluster Centroid

The Cluster Centroids under-sampling technique on the imbalanced data performs much better in all metrics as compared to our baseline model and the Near Miss model. There is also a significant increase in the precision score as compared to the previous approaches. The worst performance is observed for the Year 4 dataset, where the recall score is a lowly 0.19 while the best performance is observed in the Year 1 dataset where a recall of nearly 0.48 is achieved. The results are a much better performance from Near-Miss under-sampling technique where model is able to reduce misclassification of non-default events significantly.

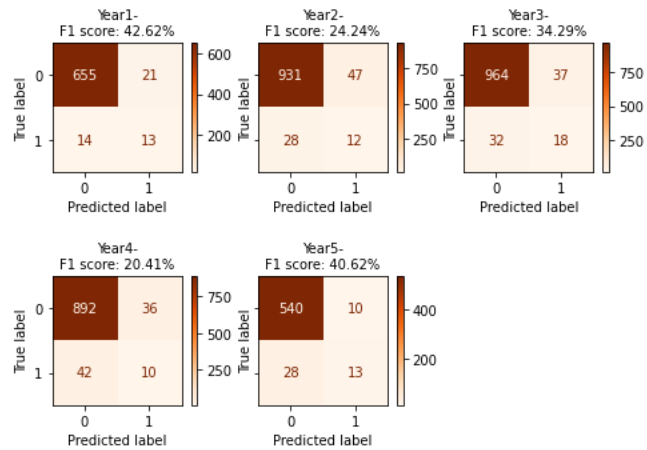

Figure 4: Under-Sampling **(**Cluster Centroid) Confusion Matrix

Dataset	Accuracy	Precision	Recall	F-1
Year 1	95%	38%	48%	43%
Year 2	93%	20%	30%	24%
Year 3	93%	33%	36%	34%
Year 4	92%	22%	19%	20%
Year 5	94%	57%	32%	41%

Table 4: Under-Sampling **(**Cluster Centroid) Evaluation Metrics

5.4 Grabit Model

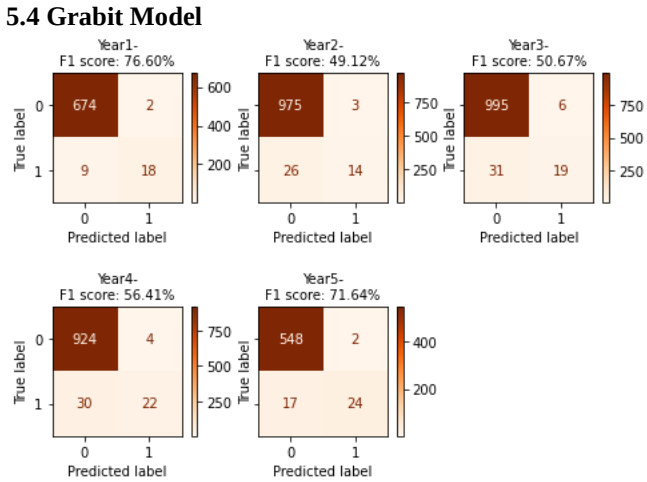

Figure 5: Grabit Model Confusion Matrix

Dataset	Accuracy	Precision	Recall	F-1
Year 1	98%	90%	67%	77%
Year 2	97%	82%	35%	49%
Year 3	96%	76%	38%	51%
Year 4	97%	85%	42%	56%
Year 5	97%	92%	59%	72%

Table 5: Grabit Model Evaluation Metrics

The Grabit model shows much more improved performance as compared to all the other previous models. The best performance is observed in the Year 1 dataset with a recall Score of 0.67. The model's high precision score across all datasets indicates that it can accurately identify companies that will not default. A comparatively lower recall score, on the other hand, indicates that it is unable to identify all the companies which will default.

5.5 Thresholding

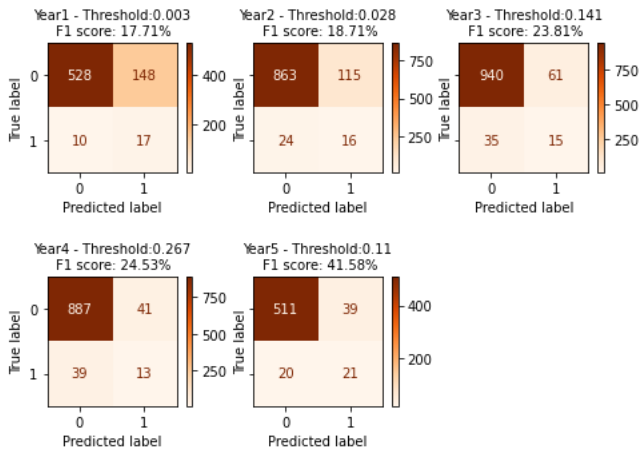

Figure 6: Thresholding Confusion Matrix

Dataset	Accuracy	Precision	Recall	F-1
Year 1	78%	10%	63%	18%
Year 2	86%	12%	40%	19%
Year 3	91%	20%	30%	24%
Year 4	92%	24%	25%	25%
Year 5	90%	35%	51%	42%

Table 6: Thresholding Evaluation Metrics

The performance of the Thresholding approach is poor across both precision and recall metrics for all datasets. For Year 1 dataset we observe a comparatively better recall value of .63 but there is a very sharp decline in the precision value. The model fails to classify default and non-default events.

5.6 Extreme Machine Learning

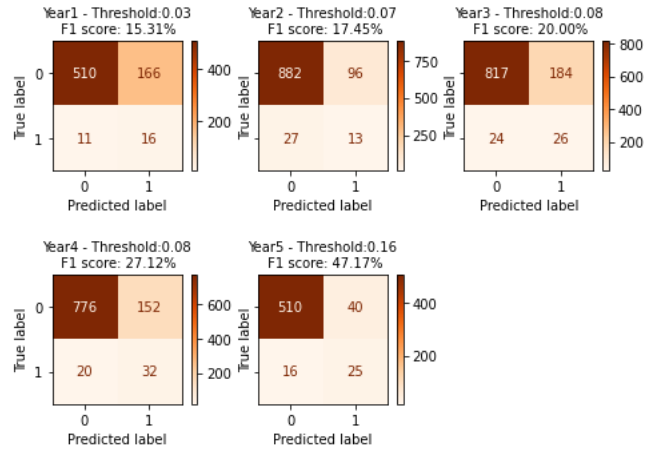

Figure 7: Extreme Machine Learning Confusion Matrix

Dataset	Accuracy	Precision	Recall	F-1
Year 1	75%	9%	59%	15%
Year 2	88%	12%	33%	17%
Year 3	80%	12%	52%	20%
Year 4	82%	17%	62%	27%
Year 5	91%	38%	61%	47%

Table 7: Extreme Machine Learning Evaluation Metrics

The Extreme machine learning model results are comparable with Grabit model when we compare the recall scores of both the approaches. However, the ELM model achieves this performance at the cost of a significantly decreased precision score, which leads to a greater misclassification of non-default events in the confusion matrix results.

6. DISCUSSION & CONCLUSION

During our study, we explored 6 different techniques for handling imbalanced dataset. We were able to put these techniques through a comprehensive evaluation by using 5 imbalanced datasets. If the same technique is found to perform well across all five datasets, then it is likely that these results are not random, but that the technique performs optimally when identifying defaults as well as non-defaults. The following figure shows the comparison of the six techniques across all five datasets using the F-1 score as an evaluation metric that factors both recall and precision. In the Year 5 and Year 1 datasets, the model's performance is better, while it declines in the Year 2, 3 and 4 datasets, possibly due to the absence of two operating profit related features that were dropped due to missing data in these three datasets. Furthermore, we find that Grabit outperforms all other modeling techniques. The Grabit model has the best F1 score across all 5 datasets with values between .49 and .77. The next best performance is achieved by using cluster

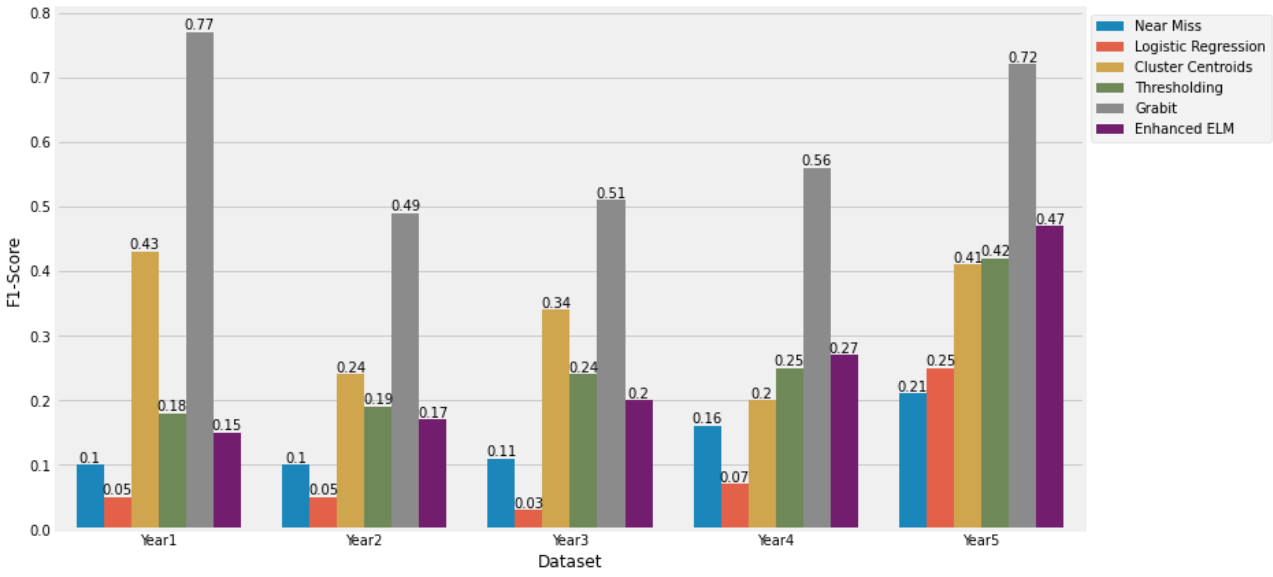

Figure 8: Comprehensive Comparison of F1 Score

centroid undersampling followed by ELM and Thresholding techniques. The baseline Logistic Regression and Near Miss undersampling techniques perform the poorest.

As a result of this project, we explored an important issue related to handling imbalanced datasets in machine learning, a problem we believe we will encounter in our future careers in data analytics and machine learning. The six techniques covered in this article are just a few that we found during our literature review. However, due to lack of time, we could not experiment with them. In the future, we would like to investigate all of them, especially the ones that use Support Vector Machine. It seems intuitively that they are well suited to solving the imbalanced dataset problem. The techniques outlined in this paper have all shown superior performance compared with the baseline Logistic Model. The performance of these additional methods can also be improved by better data processing and hyperparameter tuning. Also, we experimented with the robustness test we had proposed, but we did not achieve meaningful results, and due to the lack of literature in this area, we were unable to find any conceptual support to modify our approach and test the impact of variation on probability of default by shocking existing variables (+/-10%). Banks perform this test with statistical models with fewer variables, but our dataset contains over 60 variables, so we would like to explore the idea of robustness testing in machine learning models in our future work.

7. REFERENCES

1. Sigrist, F. and C. Hirnschall, *Grabit: Gradient tree-boosted Tobit models for default prediction.* Journal of Banking and Finance, 2019. **102**: p. 177-192.
2. Hengyu, Z., *A New Cost-sensitive SVM Algorithm for Imbalanced Dataset*. 2021, IEEE. p. 402-407.
3. Johnson, J.M. and T.M. Khoshgoftaar, *Deep Learning and Thresholding with Class-Imbalanced Big Data*. 2019, IEEE. p. 755-762.
4. Alam, T.M., et al., *An Investigation of Credit Card Default Prediction in the Imbalanced Datasets.* IEEE Access, Access, IEEE, 2020. **8**: p. 201173-201198.
5. Zhang, X. and L. Qin, *An Improved Extreme Learning Machine for Imbalanced Data Classification.* IEEE Access, Access, IEEE, 2022. **10**: p. 8634-8642.
6. Zięba, M., J.M. Tomczak, and S.K. Tomczak, *Ensemble boosted trees with synthetic features generation in application to bankruptcy prediction.* Expert Systems with Applications, 2016. **58**: p. 93-101.
7. Mani, I. and I. Zhang. *kNN approach to unbalanced data distributions: a case study involving information extraction*. in *Proceedings of workshop on learning from imbalanced datasets*. 2003. ICML.
8. Yen, S.J. and Y.S. Lee, *Cluster-based under-sampling approaches for imbalanced data distributions.* Expert Systems with Applications, 2009. **36**(3 PART 1): p. 5718-5727.

APPENDIX

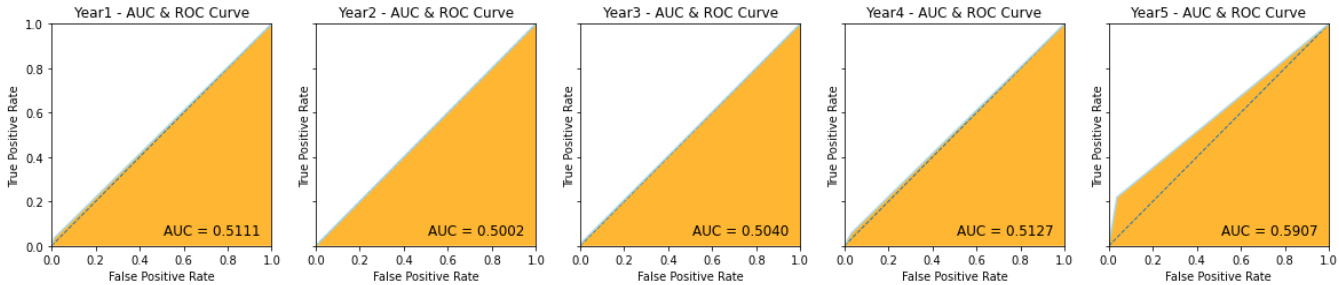

Figure 9: Logistic Regression ROC

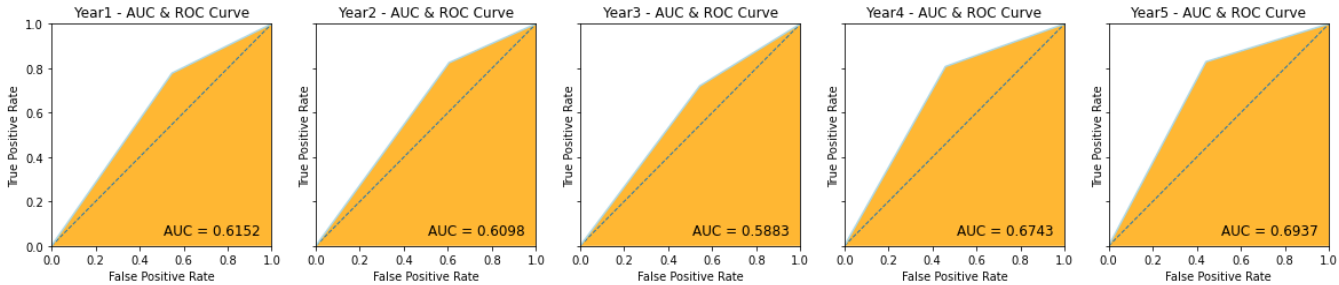

Figure 10: Under-Sampling (Near Miss) ROC

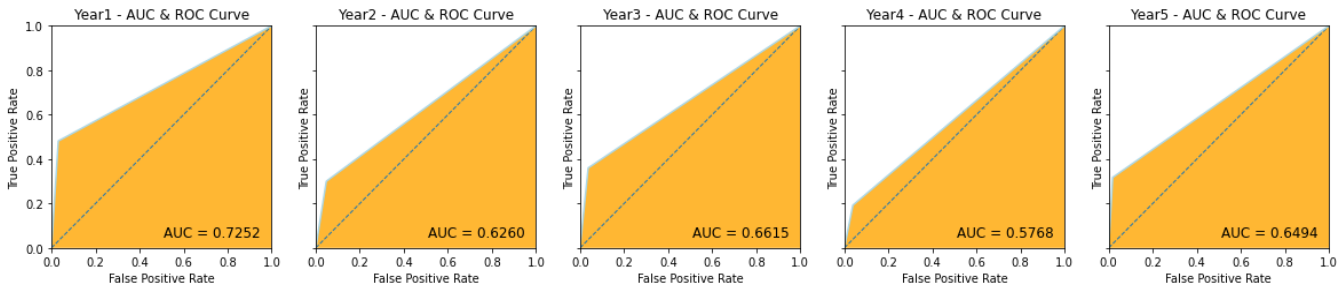

Figure 11: Under-Sampling (Cluster Centroid) ROC

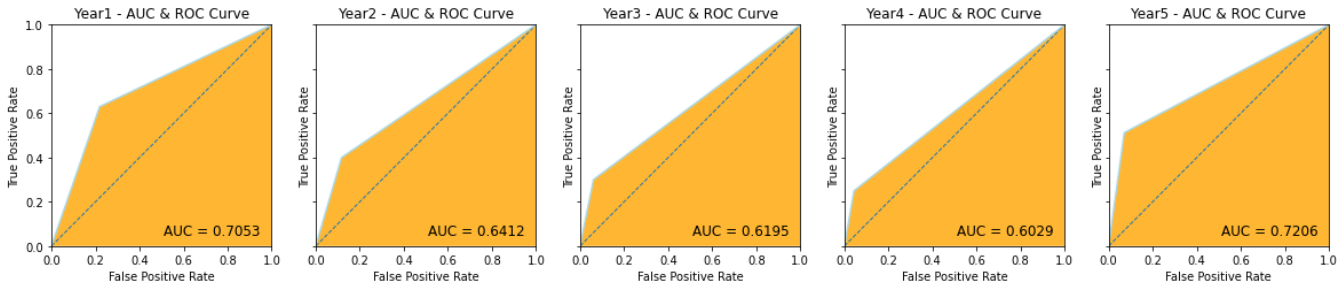

Figure 12: Thresholding ROC

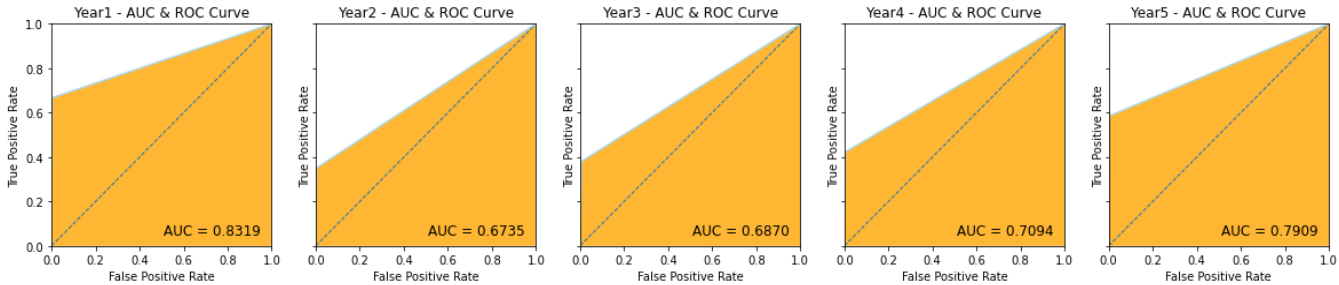

Figure 13: Grabit Model ROC

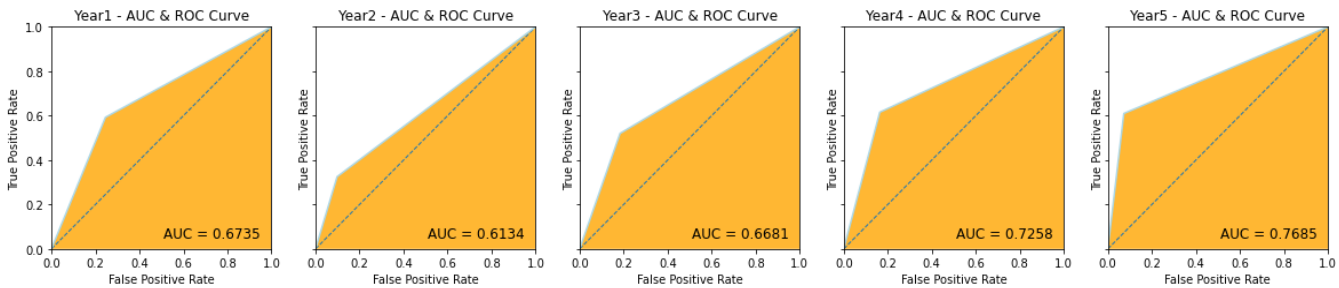

Figure 14: Extreme Learning Model ROC

*Fig 9-14 displays the AUC score of various methodologies used in this study.

Data Set Attributes

S.No.	ID	Financial Ratio	S.No.	ID	Financial Ratio
1	X1	net profit / total assets	**33**	X33	operating expenses / short-term liabilities
2	X2	total liabilities / total assets	**34**	X34	operating expenses / total liabilities
3	X3	working capital / total assets	**35**	X35	profit on sales / total assets
4	X4	current assets / short-term liabilities	**36**	X36	total sales / total assets
5	X5	[(cash + short-term securities + receivables - short-term liabilities) / (operating expenses - depreciation)] * 365	**37**	X37	(current assets - inventories) / long-term liabilities
6	X6	retained earnings / total assets	**38**	X38	constant capital / total assets
7	X7	EBIT / total assets	**39**	X39	profit on sales / sales
8	X8	book value of equity / total liabilities	**40**	X40	(current assets - inventory - receivables) / short-term liabilities
9	X9	sales / total assets	**41**	X41	total liabilities / ((profit on operating activities + depreciation) * (12/365))
10	X10	equity / total assets	**42**	X42	profit on operating activities / sales
11	X11	(gross profit + extraordinary items + financial expenses) / total assets	**43**	X43	rotation receivables + inventory turnover in days
12	X12	gross profit / short-term liabilities	**44**	X44	(receivables * 365) / sales
13	X13	(gross profit + depreciation) / sales	**45**	X45	net profit / inventory
14	X14	(gross profit + interest) / total assets	**46**	X46	(current assets - inventory) / short-term liabilities
15	X15	(total liabilities * 365) / (gross profit + depreciation)	**47**	X47	(inventory * 365) / cost of products sold
16	X16	(gross profit + depreciation) / total liabilities	**48**	X48	EBITDA (profit on operating activities - depreciation) / total assets
17	X17	total assets / total liabilities	**49**	X49	EBITDA (profit on operating activities -

S.No.	ID	Financial Ratio	S.No.	ID	Financial Ratio
					depreciation) / sales
18	X18	gross profit / total assets	**50**	X50	current assets / total liabilities
19	X19	gross profit / sales	**51**	X51	short-term liabilities / total assets
20	X20	(inventory * 365) / sales	**52**	X52	(short-term liabilities * 365) / cost of products sold)
21	X21	sales (n) / sales (n-1)	**53**	X53	equity / fixed assets
22	X22	profit on operating activities / total assets	**54**	X54	constant capital / fixed assets
23	X23	net profit / sales	**55**	X55	working capital
24	X24	gross profit (in 3 years) / total assets	**56**	X56	(sales - cost of products sold) / sales
25	X25	(equity - share capital) / total assets	**57**	X57	(current assets - inventory - short-term liabilities) / (sales - gross profit - depreciation)
26	X26	(net profit + depreciation) / total liabilities	**58**	X58	total costs /total sales
27	X27	profit on operating activities / financial expenses	**59**	X59	long-term liabilities / equity
28	X28	working capital / fixed assets	**60**	X60	sales / inventory
29	X29	logarithm of total assets	**61**	X61	sales / receivables
30	X30	(total liabilities - cash) / sales	**62**	X62	(short-term liabilities *365) / sales
31	X31	(gross profit + interest) / sales	**63**	X63	sales / short-term liabilities
32	X32	(current liabilities * 365) / cost of products sold	**64**	X64	sales / fixed assets