Dynamic Classifier Ensembling for Bankruptcy Prediction

Somshubhra Roy
sroy22@ncsu.edu

Katherine Crosby
kacrosby@ncsu.edu

I. INTRODUCTION

Unexpected financial failures can have an extensive impact on businesses, individuals, and the global economy. In the past, financial losses could destroy a bank and its ability for it to serve other individuals. This led to large-scale economic disasters and the idea that banks need to hold greater discernment and responsibility when handling potential lenders. Since the 1930s, the study of bankruptcy prediction has grown in importance and interest. The first attempts lacked computer usage for machine learning and algorithm analysis. The Univariate analysis (Beaver, 1966) was one of the first steps in bankruptcy data analysis that uses multiple financial ratios to indicate predicted ability. Each ratio was considered separately and the resulting pattern in the data is ranked on effect overall. Later, more complex models like neural networks, random forests, and extreme gradient boosting are used to analyze components of bankruptcy prediction. Current prediction accuracy percentages lie around 82-85%. Currently, recent studies show that ensemble models like gradient boosting and bagged decision trees perform best (Maciej, 2016)

A. Correlation Between Bankruptcy and Economic Attributes:

One of the first looks into the correlation between bankruptcy and economic attributes is by use of financial ratios (Beaver, 1966). They used 6 different "groups" of ratios - each comparing different statistics of the company and if there was a possible correlation between that component and the bankruptcy probability. In this analysis, they developed the idea of a "cash-flow model" in which the potential borrower is viewed as a reservoir of assets, debt, operations flow, and expenditures. The reservoir size is also considered. To analyze the individual components, the financial ratios, they analyzed how it affected the overall "cash flow" and the potential for negative "cash flow".

After a financial ratio is observed they used Baye's Theorem to create probabilities of failure as well as success. This form of classification was the beginning of analyzing the correlation between financial attributes and bankruptcy. After this model, The Z-Score model was developed (Altman, 1968). This is a multivariate model that assigns weights to five financial ratios to predict the chances of a company going bankrupt within two years using Multiple Discriminant Analyses. Obviously, the scope was much smaller than 5-10 years of bankruptcy prediction, but it allowed economists to analyze the relationship between multiple variables at once and they could also use the predicted z-score as a variable for future algorithms. The development of machine learning and neural networks lead to great breakthroughs in developing these algorithms and finding variables that had a prominent effect or weight on the chance of bankruptcy. The evaluation also began by including other sources of information such as the effects of company size, industry sector, and the economic cycle (Lennox, 1999). The step into machine learning as compared to older approaches showed major improvements to the predictor accuracy (Adnan, 2006). Their findings show that artificially intelligent models perform the best and that MDA and logit models showed promising results in research. It was also found that using global economic factors, corporate structure, and management practices as attributes for prediction could also be effective in refining the models.

B. Data Set Description:

The data sets we used are two different collections. One from the Taiwan Economic Journal between 1999-2009 is based on the bankruptcy regulations of the Taiwan Stock Exchange. This data set has a vast set of attributes to test for: with 6820 observations for 95 input attributes including ratios, flags, and cash flow variables. The second data set is based on the data collected from the Emerging Markets Information Service. It contains information on markets around the world and analyzed bankrupt companies between 2000-2012 as well as operating companies between 2007-2013. This set specifically targets bankruptcy prediction of Polish companies. This data set contains 10000 observations of 64 attributes consisting entirely of financial attribute ratios. There are lots of overlapping attributes so the potential for cross-usage between the data sets is also feasible to check for the effect of geographic attributes and government market attributes. The datasets we have gathered are highly skewed with the majority class being "Not-Bankrupt" overshadowing the minority class "Bankrupt"

Figure 1 Class Imbalance for Dataset 1

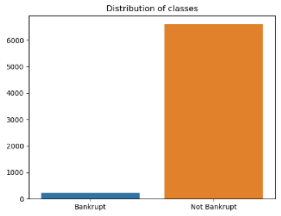

Figure 2 Class Imbalance for Dataset 2

The class imbalance present in our data is well suited to test our hypothesis under adverse conditions.

II. METHODOLOGY

A. Data Preprocessing

The distribution of target variables, bankrupt vs non-bankrupt organizations is first plotted to figure out the bias of the dataset. Then scatter plots of the target variable vs each independent variable is plotted to figure out the outliers. The distribution of each independent variable is also plotted to find out the outliers. The relative and cumulative importance of each of these attributes is plotted to find out the most important factors for Predicting bankruptcy.

B. Bias Prevention Using Oversampling

To prevent bias the rows containing outlier or absent independent variables are usually replaced by the mean of the distribution. Fortunately, our dataset did not have such issues. The minority target variable class containing the lesser distribution is oversampled using SMOTE for reducing bias. SMOTE, which stands for Synthetic Minority Over-sampling Technique, is a method for dealing with imbalanced datasets in machine learning by oversampling the minority class and creating synthetic data points using K Nearest Neighbors algorithm.

C. Data Encoding and Attribute Importance Analysis

The categorical variables are one hot encoded to convert into numerical data. The raw data is then fitted to an untuned random forest classifier with default parameters to find the average entropy importance of the individual

features. The lesser important attributes were scrapped to reduce model complexity.

D. Training

The over-sampled data was split into a training and testing set in a ratio of 7:3 with 70% of the data used for training or models and 30% for testing the performance of the trained models. Several classification models are explored including Random Forests, Gradient Boosted Random Forests, XGboost, Adaboost, and Artificial Neural Networks to figure out 5 models with consistently better F1 scores using cross-validation by splitting the training dataset into training and test set. The various hyper-parameters of each of these models including the architecture of the ANN were tuned empirically and corresponding ROC curves were plotted to find out the set of hyper-parameters that results in the highest area under the ROC curve.

III. MODELS

A. Base Classifier: Decision Tree

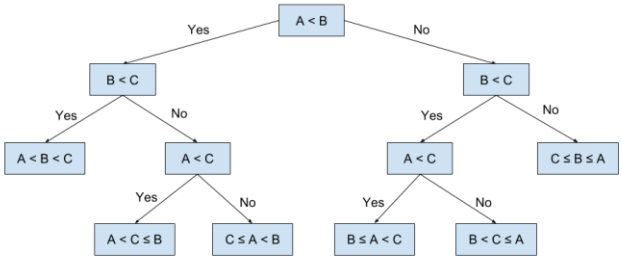

Figure 3 Decision Tree

A decision tree of depth 1 was used as a weak base classifier for our tree-based models. A decision tree is a type of machine-learning algorithm that uses a tree-like structure to make predictions. Each node in the tree represents a single decision that the model must make, and each branch represents the different possible outcomes of that decision. The leaves of the tree represent the final predicted class.

B. Random Forest Classifier

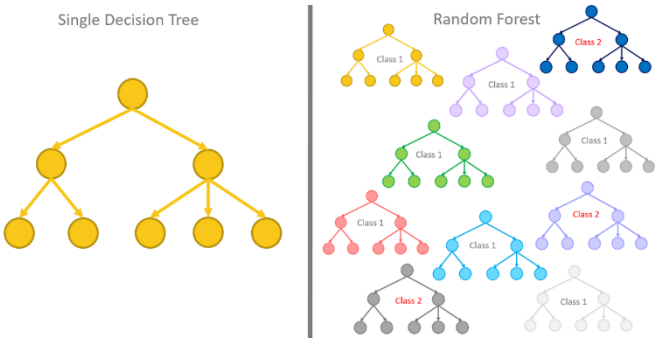

Figure 4 Random Forests

A random forest classifier is a type of supervised machine learning algorithm that is used for classification tasks. It is called a "random forest" because it trains many individual decision trees on random subsets of the data and then averages the predictions of those trees to make a final classification. In a random forest classifier, multiple decision trees are trained on different subsets of the data, and the predictions of each tree are combined to make a final prediction. This can help to reduce overfitting, improve the model's generalization ability, and make more accurate predictions on unseen data.

C. Gradient Boosting

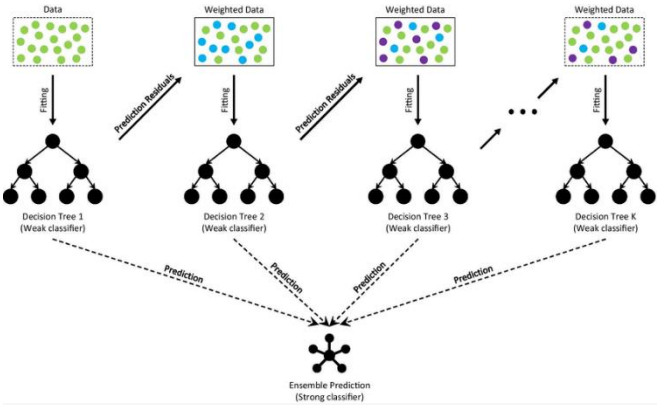

Figure 5 Gradient Boosting

A gradient-boosted classifier is a type of supervised machine learning algorithm that is used for classification tasks. It is called "gradient boosted" because it uses gradient boosting, a technique that involves training many individual weak models in a sequential manner and then combining their predictions to make a final, more accurate prediction. Gradient boosting works by training weak models, such as decision trees, on the data and then using the errors from those models to train subsequent models. This process continues until a desired number of models have been trained, at which point the predictions of all the models are combined to make a final prediction. Gradient boosting can be used to improve the performance of a model by reducing overfitting and improving the model's ability to generalize to new data. It is often used to train highly accurate models on large, complex datasets.

D. XGBoost

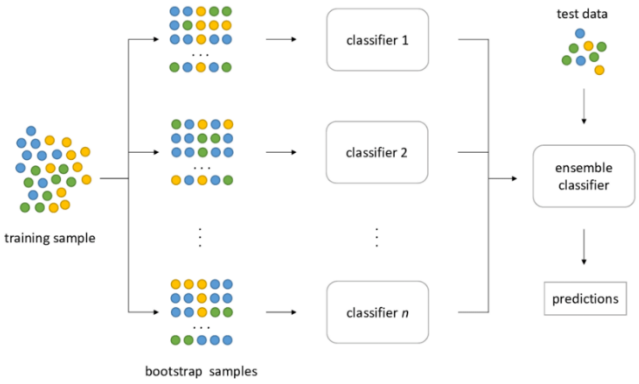

Figure 6 XGBoost

XGBoost is an implementation of gradient-boosted decision trees designed for speed and performance. It is a popular choice for many machine-learning tasks, including classification and regression. XGBoost is an optimized version of the gradient boosting algorithm that uses parallel processing and other techniques to make training faster and more efficient. It also includes a number of additional features, such as regularization and early stopping, that can help to improve the accuracy of the trained model. Because of its efficiency and performance, XGBoost has become a widely used tool in the field of machine learning and is often used in winning solutions to machine learning competitions.

E. AdaBoost

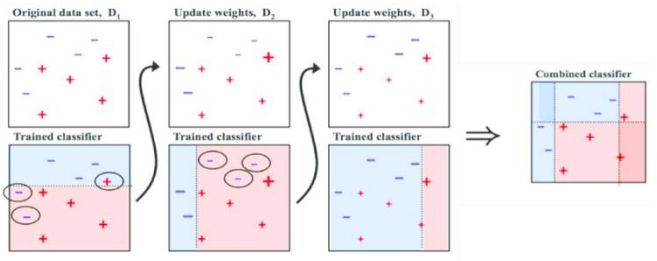

Figure 7 Adaboost

AdaBoost, short for Adaptive Boosting, is a type of ensemble learning algorithm that is used for classification tasks. It is called "adaptive" because it adjusts the weights of the training examples at each iteration so that the misclassified examples are given more weight and the correctly classified examples are given less weight. This helps the algorithm to focus on the examples that are harder to classify and improves the overall performance of the model. AdaBoost works by training a series of weak models, such as decision trees, on the data. At each iteration, the weights of the training examples are adjusted based on the performance of the previous model. Then, a new model is trained on the updated weights, and the process continues until a desired number of models have been trained. The final prediction is made by combining the predictions of all the trained models using a weighted majority vote. This can help to improve the performance of the model by reducing overfitting and improving the model's ability to generalize to new data.

F. Artificial Neural Network

An ANN, or artificial neural network, is a type of machine learning algorithm that is modelled after the structure and function of the human brain. It is called a "neural network" because it is composed of many interconnected nodes, which are called neurons. The most common type of ANN is the feedforward neural network, which consists of an input layer, one or more hidden layers, and an output layer. The input is passed through the network, layer by layer, and the final output is produced at the output layer. Our training set was again split into a training and validation set in a 7:3 ratio for the fission of our custom ANN. Our designed ANN has 5 fully connected layers each with a dropout of 0.5 to prevent overfitting. We have used the sigmoid activation function instead of the standard Rectified linear unit function in each layer since the number of parameters in each layer is much less than in standard models and sigmoids were very prevalent in early iterations of primitive multilayer perceptrons. The learning rate has been reduced linearly in each epoch starting with an initial learning rate of 0.01. The other parameters such as regularisation constant, batch normalization constant, and batch normalization momentum have been kept to the default values of Alexnet since its architecture with dropout is somewhat similar to ours. The model was trained for 200 epochs with a batch size of 200. The architecture, loss and accuracy plots of the model on each dataset are provided below.

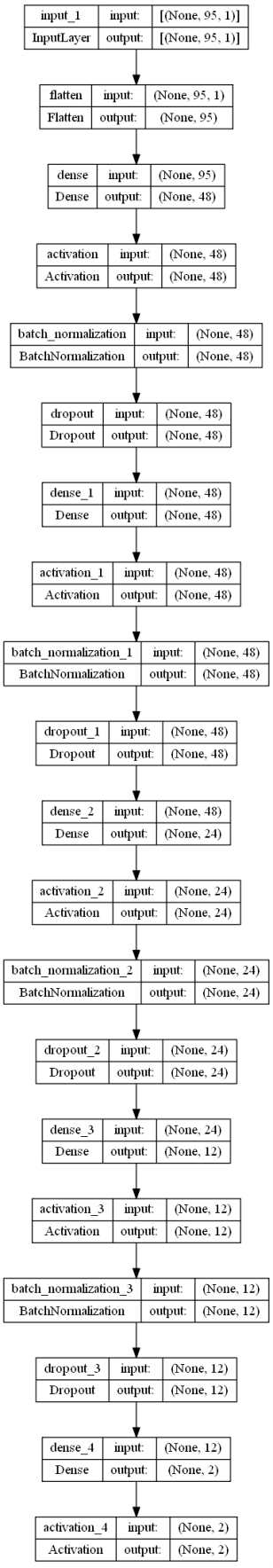

Figure 8 ANN architecture

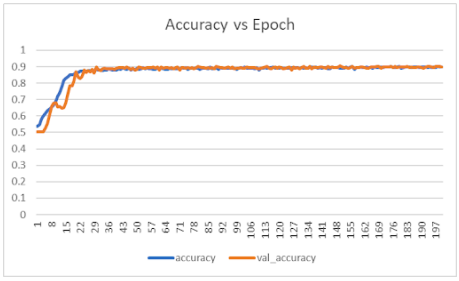

Figure 9 Accuracy Vs. Epoch Dataset1

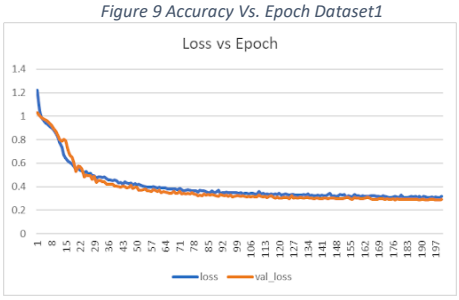

Figure 10 Loss Vs. Epoch Dataset2

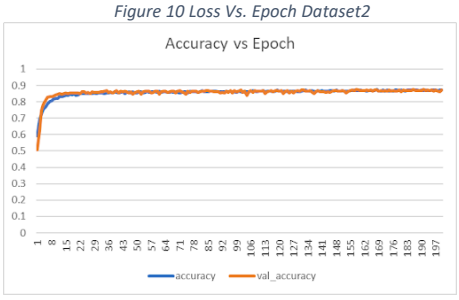

Figure 9 Accuracy Vs. Epoch Dataset2

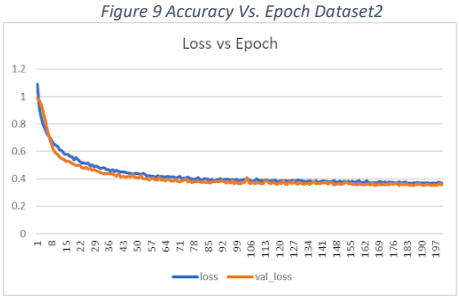

Figure 10 Loss Vs. Epoch Dataset2

G. Ensemble Classifier

These 4 models were then ensembled together and weight is attributed to the prediction score of each model. We set s1, s2, s3, s4, and s5 be the classification score corresponding to each of the 4 selected models. We set $\alpha,\beta,\gamma,\delta,$ and θ be the weight attributed to each model. Therefore, the prediction score of the ensemble model $s = \alpha*s1+\beta*s2+\gamma*s3+\delta*s4+\theta*s5$. Since the prediction confidence score of a model is a measure of probability, s has to belong to the interval [0,1]. Thus, our constraint is $\alpha+\beta+\gamma+\delta+\theta = 1$. The optimal ratio of $\alpha,\beta,\gamma,\delta,$ and θ can be iteratively found using a grid search to explore if the ROC- AUC score can be pushed higher than individual models. However, because of 5 coefficients, the complexity of the optimization function is of the order $O(5*k*(n^5))$ where O(k) is the complexity of individual models and n is the number of sequences between [0,1] we go through for the cross-validation grid search. Generalized to m models our model complexity would be $O(m*k*(n^m))$. To reduce this complexity instead of iteratively going through each possible combination of alpha, beta, gamma, delta, and theta we apply our constraint $\alpha+\beta+\gamma+\delta+\theta = 1$ and only select values that fit our constraint. Using this dynamic programming method, we were able to reduce our model complexity to $O(m*k)$ since the number of combinations we go through is a predetermined constant.

IV. FINDINGS

A. Random Forest Classifier

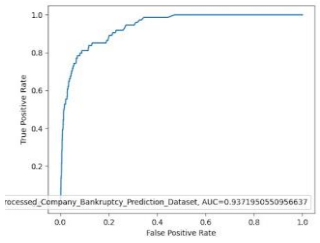

Figure 13a ROC curve Dataset 1

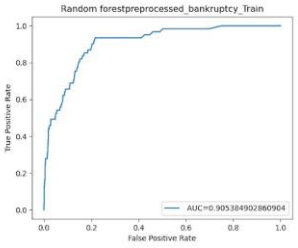

Figure 11b ROC curve Dataset 2

B. Gradient Boost Classifier

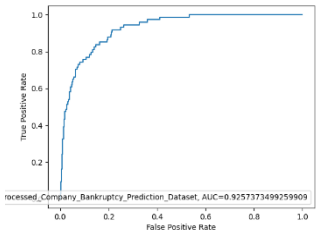

Figure 14a ROC curve Dataset 1

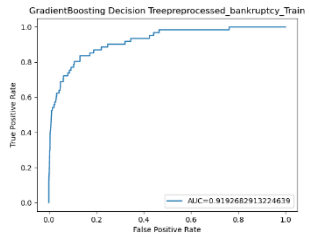

Figure 14b ROC curve Dataset 2

C. XGBoost Classifier

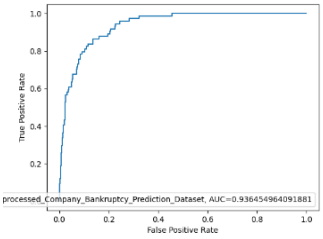

Figure 15a ROC curve Dataset 1

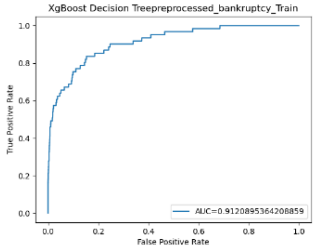

Figure 15b ROC curve Dataset 2

D. ADABoost Classifier

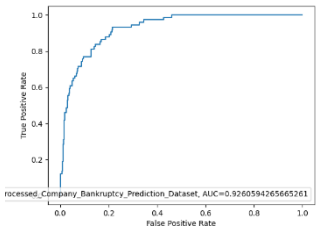

Figure 16a ROC curve Dataset 1

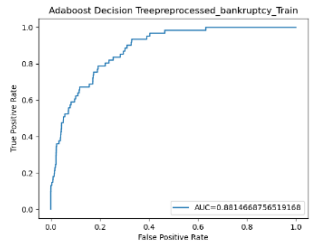

Figure 16b ROC curve Dataset 2

E. ANN Classifier

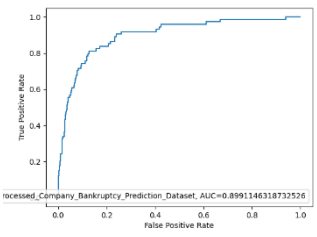

Figure 17a ROC curve Dataset 1

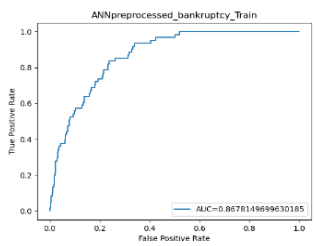

Figure 17b ROC curve Dataset 2

F. Accuracy Findings

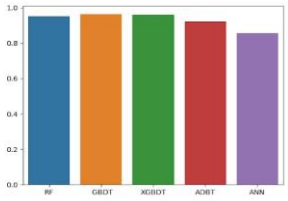

Figure 18a Accuracy Plots Dataset 1

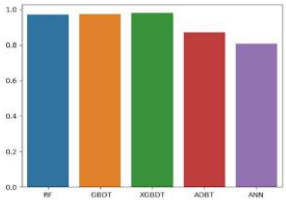

Figure 18b Accuracy Plots Dataset 2

We can see here that the performance of our ANN is worse than that of our tree-based classifiers which is a condition well documented for ANNs However newer architectures like TabNet has successfully overcome this issue. One interesting thing we can note here is that because of its flexibility and use of smote our ANN is consistently better at classifying the minority class resulting in higher recall.

G. Ensemble Classifier

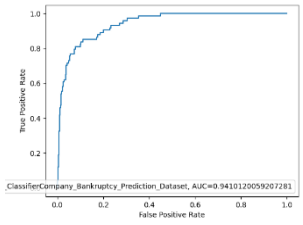

Figure 19a ROC curve Dataset 2

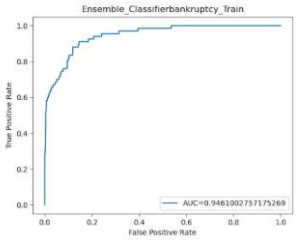

Figure 19b ROC curve Dataset 2

Here we see the AUC score for the parametric ensemble model was 94.1% for dataset 1 with the following optimal parameters: α: 0.0, β: 0.8, γ: 0.2, δ: 0.0, θ: 0.0 which is 1% greater than the maximum AUC score of the individual classifiers (Random Forest with an AUC score of (93.1%) in this case). We also see the AUC score for the parametric ensemble model was 94.6% for dataset 2 α: 0.3, β: 0.0, γ: 0.0, δ: 0.7, θ: 0.0 which is more than 3% greater than the maximum AUC score of the individual classifiers (XGBoost with an AUC score of 91.1% in this case)

V. Summary & Discussion

From our results, our hypothesis proved that a dynamic parameterized ensemble of classifiers can indeed boost the performance metrics over individual models. Our ensemble model iteratively selects the proportion of weights given to the prediction score of each individual model so as to maximize the Area under the ROC curve. Our future roadmap is to introduce undersampling of the majority class, using a weighted entropy loss function to combat class imbalance. Application of Principal Component Analysis (PCA)for further dimensionality reduction and introducing nested ensembling of dynamic depth. Our models were trained on a machine with the following specifications: CPU: i7-11800H 8 core 16 threads 4.3Mhz RAM: 16 GB 3.6 Mhz GPU: RTX 3060 8GB

VI. Citations

1. Adnan, Aziz, and Humayon Dar. 2006. Predicting corporate bankruptcy: Where we stand? Corporate Governance 6: 18–33.
2. Altman, Edward, Haldeman Robert, and Paul Narayanan. 1977. ZETA analysis A new model to identify bankruptcy risk of corporations. Journal of Banking and Finance 1:24-54.
3. Baldwin, Jane, and William Glezen. 1992. Bankruptcy Prediction Using Quarterly Financial Statement Data. Journal of Accounting,Auditing and Finance 7: 269–85.
4. Barboza, Flavio, Herbert Kimura, and Edward Altman. 2017. Machine learning models and bankruptcy prediction. Experts System withApplications 83: 405–17.
5. Beaver, William. 1966. Financial Ratios as Predictors of Failure. Journal of Accounting Research, Empirical Research in Accounting 4: 71–111.
6. Boser, Bernhard, Isabelle Guyon, and Vladimir Vapnik. 1992. A training algorithm for optimal margin classifiers. In COLT '92:Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, PA, USA, 27–29 July 1992. New York: ACM,pp. 144–52.
7. Brédart, Xavier. 2014. Bankruptcy Prediction model using Neural networks. Accounting and Finance Research 3: 124–28.
8. Brédart, Xavier, and Loredana Cultrera. 2016. Bankruptcy prediction: The case of Belgian SMEs. Review of Accounting and Finance 15:101–19.
9. Campbell, John, Jens Hilscher, and Jan Szilagyi. 2008. In search of distress risk. The Journal of Finance 63: 2899–939.
10. Chen, Tianqi, and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In KDD '16 Proceedings of the 22nd ACM SIGKDDInternational Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016. New York: ACM, pp.785–94
11. Lennox, Clive. 1999. Identifying failing companies: A revaluation of the logit, probit and da approaches. Journal of Economics and Business 51: 347.
12. Maciej Zieba, S. K. (2016). Ensemble Boosted Trees with Synthetic Features Generation in Application to Bankruptcy Prediction. Wrocław University of Science and Technology, Department of Computer Science, Department of Operations Research, Faculty of Computer Science and Management, Wrocław. J. Risk Financial Manag. 2022,15, 35 9 of 10.
13. Natras, R.; Soja, B.; Schmidt, M. Ensemble Machine Learning of Random Forest, AdaBoost and XGBoost for Vertical Total Electron Content Forecasting. Remote Sens. 2022, 14, 3547.