The Company Grim Reaper: A Machine Learning Model for Predicting Bankruptcy

This report details the findings from implementing several statistical learning models for company bankruptcy prediction. It was shown that Neural Networks was most ideal for the classification task on the dataset, and that Debt to Equity Ratio was the most relevant financial indicator.

Group Members:

- Bee Shu Wen, A0220615U
- Chua Yong Ting Hanis, A0222356L
- Ong Jia Yee, A0222295H
- Tan Jia Xin Shereen A0223296A

1 Introduction | Motivation, Objectives & Overview of Statistical Approaches

Bankruptcy prediction is likely to be an important factor for banks when approached by companies for financing. It is in the interest of banks to provide financing offers only to borrowers with a high repayment ability. We believe that banks can benefit from having a bankruptcy prediction model that learns some signs of imminent bankruptcy and classifies companies, allowing unusual cases to be highlighted for inspection with greater ease.

We aimed to implement and refine several statistical learning models to predict company bankruptcy. In the process, we hope to uncover useful indicators of company bankruptcy.

Our project is focused on studying supervised learning techniques. For parametric models, we will be looking at Logistic Regression. For non-parametric models, we will look into Support Vector Machines, Neural Networks, and a variety of Tree-Based Methods.

2 Data Analysis and Preprocessing

2.1 Dataset Description

The dataset is collected from the Taiwan Economic Journal for the years 1999 to 2009. It contains 6819 observations with 95 quantitative features comprising a company's financial performance measures. The response is a binary variable indicating the bankruptcy status of the corresponding company. With only 3.3% of the observations being labeled as bankrupt, the dataset is highly imbalanced.

2.2 Data Cleaning

The column *Net Income Flag* was found to be of no predictive power since it contains only one unique number. Besides this, the dataset is clean and consistent with no missing values or duplicate rows.

2.3 Data Analysis | Outlier Detection, Correlation Investigation & Preliminary Research

Since our dataset has a lot of predictors, we were cautious of removing outliers using single dimensional box plots or univariate summary statistics, as we are aware that what seems like an outlier in one dimension may not be an outlier in the full predictor space. With this in mind, we used a popular

anomaly detection algorithm, Isolation Forest, to identify the outliers. Much like a Random Forest, it relies on a forest of trees to assign anomaly scores to each observation. For our case, the Isolation Forest flagged out 81 points as outliers, which we removed accordingly.

For predictor-predictor correlation, we found that many of the predictors are significantly correlated; we expect data preprocessing to be helpful in reducing the impact of collinearity on inference-making later on. For predictor-response correlation, we mostly found weak correlations. It is reasonable to expect that some predictors have weak but linear relationships, and other predictors have non-linear relationships with the response.

An internet research and an interview was conducted as part of our preliminary research. The top two most frequent factors mentioned by news articles and research papers are: (1) Cash Flow Statement, and (2) Debt to Equity Ratio. In our dataset, we found 7 variables (*X13, X20, X36, X75, X80, X81, X83*) related to the two factors mentioned above. An interview with an accounting student revealed that more than 50 variables in our dataset were important for bankruptcy prediction. The exact list of variables can be found in the appendix. The sole purpose of the information obtained in this preliminary research is to serve as a basis of comparison with subsequent findings on variable importance via our models. Besides this, the information will not be used in any other way, such as model building.

2.4 Data Preprocessing | Class Balancing, Feature Selection & Dimensionality Reduction

The data was first split into a ratio of 0.6:0.1:0.3 which corresponds to 4042, 673 and 2023 observations for training, validation and test sets respectively.

To address the class imbalance in our data, we looked into sampling. We opted against under-sampling techniques since it is less suitable for our dataset which has a lower number of observations, as it might lead to the loss of important information in the data. We also compared Synthetic Minority Over-sampling Technique (SMOTE), which makes use of k-nearest neighbours from the minority class to generate synthetic samples, and Adaptive Synthetic Sampling (ADASYN), which is an extension of SMOTE that is more adaptive and is able to generate data for “harder to learn” examples. We

decided on ADASYN as our over-sampling technique to balance our training data due to its suitability for higher degrees of class imbalance. With ADASYN, we arrived at a dataset where 41% are minority class observations. Specifically, the final training data consists of 2758 observations of class 1 (bankrupt) and 3940 of class 0 (non-bankrupt).

Feature selection is a necessary process to eliminate irrelevant features. Including redundant variables will reduce a model's generalization capability and affect the overall performance. In addition, feature selection can simplify models, making them easier and faster to train which is important since we have a large dataset. Three supervised feature selection techniques, the Filter method, the Wrapper method and the Embedded method were considered. In the end, we selected LASSO, an Embedded method, because it is fast, takes into consideration the interactions between features, and is much less prone to overfitting. LASSO was chosen over Ridge because Ridge only reduces the complexity of the model but does not reduce the number of variables.

With the 28 features selected by LASSO, we used Principal Component Analysis (PCA) to further concentrate the signals of bankruptcy in the first few Principal Components (PCs), so that fewer variables can be used while retaining as much variance in the dataset as possible. 15 PCs were retained so that at least 80% of the variance in the dataset is accounted for.

In some cases, we will use the 28 features selected by LASSO for model building. In other cases, we will use the 15 PCs, and the usage of these 2 sets of features depend on the model and its suitability.

3 Model Building

3.1 Logistic Regression

We started off with logistic regression, a parametric model. One of its assumptions is that there is no multicollinearity in the predictors. This assumption can be easily dealt with by using the 15 PCs as predictors. After fitting the model with the 15 PCs, we went on to investigate the linearity in the predictors and the log-odds, which is another important model assumption. Through plots, we found that the assumption does not hold. As this may lead to biased or unreliable estimates, we tried to transform the

predictors in hopes of meeting the linearity assumption. The transformations we attempted include the logarithmic, power, n^{th} root and the more flexible spline transformation. However, the results were similar and the linearity assumption still fails for some predictors. With this, we conclude that although the validation set result of the logistic regression model was decent, it may not be suitable as its underlying assumption is violated. Hence, we shall look at non-parametric methods that impose less restriction.

3.2 Support Vector Machine (SVM)

We chose SVM due to its ability to enlarge the feature space and model non-linear decision boundaries using the kernel trick. We utilised two sets of features in our SVM analysis - one set of features which was before PCA was performed, and another set derived after PCA was performed. This was decided as we wanted to experiment and determine whether performing PCA which results in a lesser number of features has an impact on the model performance.

We experimented with linear, radial, and polynomial kernels. To select the optimal combination of parameters for each kernel, 10-fold Cross-Validation (CV) was performed. We will look at the regularization cost parameter, and for kernel-specific parameters, we will look at gamma for the radial kernel and degree for the polynomial kernel. The exact values considered can be found in the appendix.

The best threshold value is defined as the point where recall could be maximised while minimising the reduction in precision, and will be found with the help of a precision-recall (PR) curve based on validation set performance.

Linear Kernel (After PCA)

A cost of 10000 was selected using 10-fold CV. By setting the threshold at 0.3, 9 bankruptcies are successfully detected, the model achieves a False Positive Rate (FPR) of 0.160 and a True Positive Rate (TPR) of 0.429.

Radial Kernel (After PCA)

A cost of 100 and gamma of 0.5 was selected using 10-fold CV. By setting the threshold at 1.8, 13 bankruptcies are successfully detected, the model achieves an FPR of 0.446 and a TPR of 0.619. While

the radial kernel outperforms the linear kernel in terms of TPR, it performs considerably worse in terms of FPR.

Polynomial Kernel (After PCA)

A cost of 10000 and degree of 2 was selected using 10-fold CV. Interestingly, the PR curve for this kernel is the same as that for the previous linear kernel despite offering a higher degree of flexibility. By setting the same threshold of 0.3, both polynomial and linear kernels have the same performance. 9 bankruptcies are successfully detected, the model achieves an FPR of 0.160 and a TPR of 0.429.

Linear Kernel (Before PCA)

A cost of 1000 was selected using 10-fold CV. By setting the threshold at 2.6, 14 bankruptcies are successfully detected, the model achieves an FPR of 0.173 and a TPR of 0.667.

Radial Kernel (Before PCA)

A cost of 10 and gamma of 0.5 was selected using 10-fold CV. We set the threshold at 0.85 and just like the linear kernel, 14 bankruptcies are successfully detected, and a TPR of 0.667 is attained. However, the radial kernel performs worse with a larger FPR of 0.258.

Polynomial Kernel (Before PCA)

A cost of 1000 and degree of 2 was selected using 10-fold CV. Just as the dataset after PCA, using the dataset before PCA also returned the same PR curve for both polynomial and linear kernels. Setting the same threshold of 2.6, 14 bankruptcies are successfully detected, the model achieves an FPR of 0.258 and a TPR of 0.667.

In general, the dataset before performing PCA yields better results in terms of detecting more bankruptcies and achieving a higher TPR, while maintaining an acceptable level of FPR. Among the kernels used on the dataset before PCA, the linear and polynomial kernels outperforms the radial kernel. As the linear and polynomial kernels have the same model performance, the additional complexity of a polynomial kernel is unnecessary and hence, the simpler model with linear kernel (before PCA) is the best model for SVM.

3.3 Neural Networks (NNs)

The attractive yet dangerous part of NNs is having a myriad of hyperparameters controlling model complexity. For hyperparameter tuning, we looked at: (1) the number of hidden layers and nodes, (2) regularisation, (3) batch size, and (4) number of epochs. Other hyperparameters (detailed in the appendix) were fixed throughout. Since we are working on a classification task, the loss function was cross entropy loss.

To establish a baseline, we implemented a single layer NN with 8 hidden nodes. The model was trained with batch size 64 for 20 epochs. Evaluation on the validation set yielded a high TPR of 0.952 and a satisfactory FPR of 0.339.

Using 10-fold cross-validation and accuracy as the metric for hyperparameter tuning, we end up with a NN with 2 hidden layers of 14 nodes each. The model was trained with batch size 32 for 20 epochs. Evaluation on the validation set yielded a TPR of 0.667 and a FPR of 0.176. We hypothesised a few reasons for the lackluster performance of the tuned model: (1) the effects of class imbalance on model training were underestimated and exacerbated when using accuracy as the evaluation metric for hyperparameter tuning; (2) there was excessive focus placed on improving the accuracy as a number; (3) there was insufficient emphasis on preventing overfitting.

With the above findings, a second run of hyperparameter tuning used precision and recall as evaluation metrics, and conscious effort was made to verify that improvements in model performance were comparable to the additional complexity introduced to the model. The resulting model obtained was a single layer NN with 4 nodes in the hidden layer. The model was trained with batch size 64 for 20 epochs. Evaluation on the validation set yielded a TPR of 0.952 and a FPR of 0.337.

In summary, we have seen that NNs can be a very powerful tool for bankruptcy prediction. However, like most models, it is to be noted that: (1) the evaluation metric should be carefully selected to suit the data, and (2) overfitting should be a key concern. Future work could be to explore the use of

original features instead of principal components as inputs to the neural network, and to investigate model explaining, given that these complex models may be used to facilitate important decision-making.

3.4 Tree-Based Methods

With many features in our dataset, it is likely that there is a highly non-linear and complex relationship between features. Hence, the Tree-Based Method is implemented using the features selected from LASSO to handle multicollinearity and infer useful variables that can effectively detect bankruptcies.

Upon taking Precision and Recall into consideration, we adjusted our threshold. As we prioritize not having bankrupt cases go undetected, we selected a threshold that has a higher Recall score but not neglecting Precision, we ensured that Precision is minimally 0.20. The optimal threshold is found using the validation set. We apply this to all the Tree-Based Methods except Boosting.

Classification Tree

From the classification tree in Figure(1), we see that there are 11 terminal nodes, variables *X36* and *X1* are the top two most important variables.

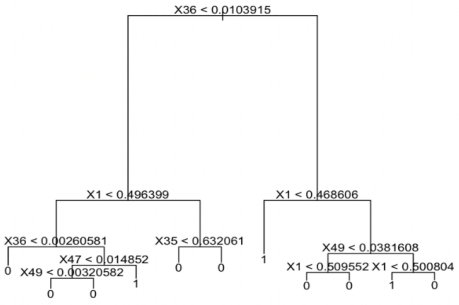


Figure (1): Classification Tree

Applying the optimal threshold = 0.50 on our validation set, the number of bankruptcies detected is 15, FPR is 0.0613 and TPR is 0.714.

Pruned Tree

We attempt to prune the classification tree to prevent overfitting and to lower variance. Using cost complexity pruning, 8 terminal nodes are found to have the lowest cross-validation error rate, Figure (2).

Fitting 8 terminal nodes back to the original classification tree, we obtained a pruned tree, Figure(3). *X36* and *X1* remain as our top two most important variables.

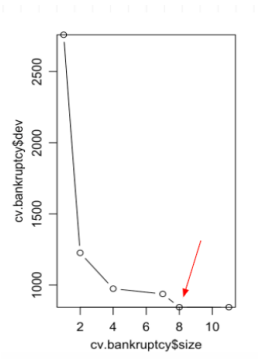


Figure (2): Lowest cv error rate

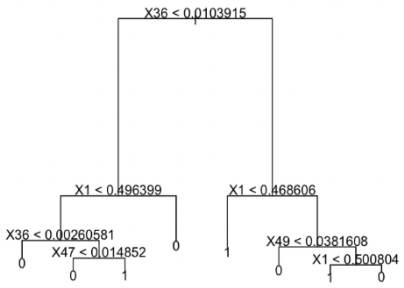


Figure (3): Pruned Classification Tree

Applying the optimal threshold = 0.80 on our validation set, the number of bankruptcies detected, FPR and TPR are the same as the ones derived from an unpruned classification tree. Pruning did not lead to any improvement.

Bagging

In hope to obtain better results, we implement Bagging. Bagging reduces the variance of a decision tree as the tree is built by taking the average of many independent decision trees. Applying the optimal threshold = 0.25 on our validation set, the number of bankruptcies detected is 14, FPR is 0.0798 and TPR is 0.667.

Not only did Bagging not lead to any improvement, it performed worse than the original classification tree. This could possibly be due to the minimal decrease in variance and increase in bias. As the entire feature space is used to create the splits in the individual trees, it is very likely that the trees are going to be similar and highly correlated with one another which results in minimal decrease in variance when taking their average. The increase in bias is because Bagging uses many decision trees.

RandomForest

RandomForest addresses the problem in Bagging by forcing each split to only consider a subset of predictors. Applying the optimal threshold = 0.20 on our validation set, the number of bankruptcies detected is 17, FPR is 0.104 and TPR is 0.810. There is an improvement from Bagging.

Boosting

Boosting, as a slow learner is expected to perform well. In Boosting, trees are grown sequentially where new trees are formed by considering the errors from previous trees. Using the default parameters, the number of trees = 5000 and shrinkage = 0.1. Since the recall and precision are similar for all the thresholds, we pick the one with the highest recall and precision which is 0.60.

Applying these on our validation set, the number of bankruptcies detected is 10, FPR is 0.00767 and TPR is 0.476. Boosting performed badly. This could be due to the presence of noise and undetected outliers, which results in the model being highly dependent on them.

In an attempt to obtain better results, we tuned the parameters. Setting the number of trees = 2000, shrinkage = 1 and the optimal threshold = 0.10 on our validation set, the number of bankruptcies detected is 12, FPR is 0.0445 and TPR is 0.571. We obtained better results.

Among the Tree-Based Methods, RandomForest performs the best.

Variable Importance

For Bagging and RandomForest, Mean Decrease Accuracy and Mean Decrease Gini are used to identify important variables. The higher the Mean Decrease Accuracy or Mean Decrease Gini, the greater the importance of that variable in the model. *X1* and *X36* are the top two most important variables.

For Boosting, relative influence is used to identify important variables. The top three most important variables with a relative influence greater than 10 are *X36, X1* and *X69*.

The common variables identified as important by Bagging, RandomForest and Boosting are *X1* and *X36*.

4 Results and Discussion

4.1 Selected Model and Performance on Test Set

We compared the performance of all the models and concluded that the single layer neural network with 4 nodes in the hidden layer performs the best for bankruptcy prediction. Evaluating model

performance on the test set, 69 out of 71 bankruptcies were detected, and a FPR of 0.328 and a TPR of 0.972 was achieved.

4.2 Most Relevant Indicator of Bankruptcy

To check that the variables identified by our models as important are valid, we returned to our preliminary research findings. Consolidating all our findings from preliminary research and from model inferences, we conclude that the most useful and important feature in predicting bankruptcy is *X36, Total debt/Total net worth: Total Liability/Equity Ratio*.

4.3 Limitations

One limitation of our dataset is the lack of information on the stage of growth of the company and the industry it is in. Companies in different industries have different debt and risk levels. Industries such as the stock market sector tend to be more volatile and risky than the others. Hence, an indicator that is identified as important in predicting bankruptcy for one industry might not be the same for another industry. The same goes to the stage of growth of the company. For startups, their financial status might not be as stable as an established business but that does not mean that there is no potential growth and this emphasizes the need to look beyond financial indicators to have a more holistic assessment. Therefore, we acknowledge that using a one fit all approach is unrealistic and there are still many factors to be taken into consideration before predicting whether a company will go bankrupt.

5 Conclusion

With the aim of improving workflow in banks, especially in terms of providing financial loans to companies, our group explored 4 different machine learning models in hope of building one that can accurately predict whether a company will go bankrupt. Among the 4 models, Neural Network performed the best and when tested on the test dataset, we see a very good result. Additionally, we were able to identify one key indicator in predicting bankruptcy, this indicator is *Total debt/Total net worth: Total Liability/Equity Ratio*. Rounding up, we have achieved both prediction and inference for our project.

Appendix

2.3.3 Preliminary Research

The second part of preliminary research consists of an interview with an accounting student. A list of financial indicators from our dataset was provided and they were asked which variables were important from their background in accounting. More than 50 variables were shortlisted and are as follows: X8, X11, X13, X14, X16, X17, X18, X19, X20, X27, X29, X30, X31, X33, X34, X35, X36, X37, X38, X39, X40, X41, X42, X50, X54, X55, X56, X57, X58, X59, X60, X61, X62, X63, X64, X65, X66, X67, X68, X71, X72, X73, X74, X75, X77, X78, X79, X81, X82, X83, X84, X85, X90, X91, X92, X93, X94 and X95.

2.4.2 Feature Selection

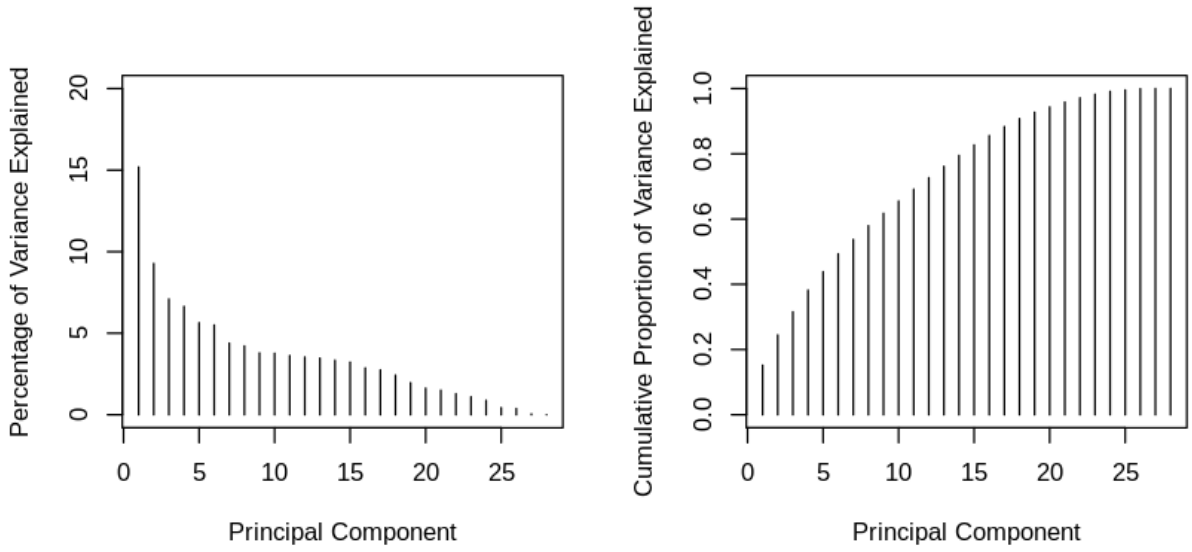
The following list comprises the 28 features selected from LASSO:

X1 - ROA(C) before interest and depreciation before interest: Return On Total Assets(C)
X4 - Operating Gross Margin: Gross Profit/Net Sales
X12 - Research and development expense rate: (Research and Development Expenses)/Net Sales
X16 - Net Value Per Share (B): Book Value Per Share(B)
X20 - Cash Flow Per Share
X27 - Regular Net Profit Growth Rate: Continuing Operating Income after Tax Growth
X35 - Interest Expense Ratio: Interest Expenses/Total Revenue
X36 - Total debt/Total net worth: Total Liability/Equity Ratio
X37 - Debt ratio %: Liability/Total Assets
X38 - Net worth/Assets: Equity/Total Assets
X39 - Long-term fund suitability ratio (A): (Long-term Liability + Equity)/Fixed Assets
X41 - Contingent liabilities/Net worth: Contingent Liability/Equity
X45 - Total Asset Turnover
X47 - Average Collection Days: Days Receivable Outstanding
X49 - Fixed Assets Turnover Frequency
X50 - Net Worth Turnover Rate (times): Equity Turnover
X51 - Revenue per person: Sales Per Employee
X52 - Operating profit per person: Operation Income Per Employee
X53 - Allocation rate per person: Fixed Assets Per Employee
X58 - Quick Assets/Current Liability
X61 - Operating Funds to Liability
X69 - Total income/Total expense
X73 - Working Capital Turnover Rate: Working Capital to Sales
X75 - Cash Flow to Sales
X82 - CFO to Assets
X83 - Cash Flow to Equity
X87 - Total assets to GNP price
X93 - Interest Coverage Ratio (Interest expense to EBIT)

2.4.3 Dimensionality Reduction

The number of principal components to retain is selected taking into consideration the following:

- Plot of Variance Explained by each PC (left)
- Plot of Cumulative Proportion of Variance Explained by PCs (right)



3.2 Support Vector Machine (SVM)

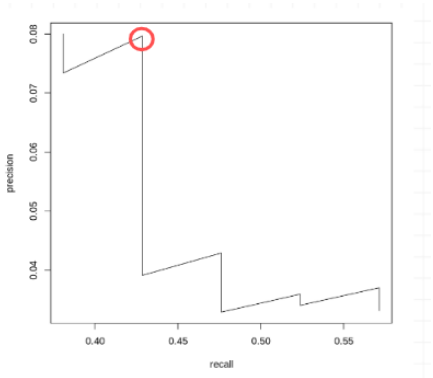
The specific values considered for parameter tuning are as follows:

- Cost: 0.1, 1, 10, 100, 1000, 10000
- Gamma: 0.5, 1, 2, 3 (for radial kernel)
- Degree: 2 to 6 (for polynomial kernel)

For each SVM model, we present below the:

- precision-recall curve
- precision and recall values at every threshold value
- confusion matrix

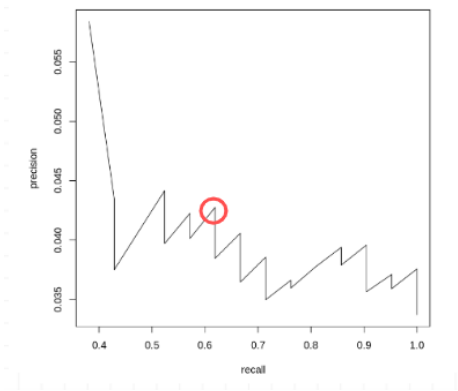
Linear Kernel (after PCA)



threshold	rec	prec
<dbl>	<dbl>	<dbl>
0.00	0.3809524	0.08000000
0.05	0.3809524	0.07843137
0.10	0.3809524	0.07766990
0.15	0.3809524	0.07619048
0.20	0.3809524	0.07619048
0.25	0.3809524	0.07339450
0.30	0.4285714	0.07964602
0.35	0.4285714	0.07964602
0.40	0.4285714	0.07894737
0.45	0.4285714	0.07894737
0.50	0.4285714	0.07826087

	pred	
true	0	1
0	548	104
1	12	9

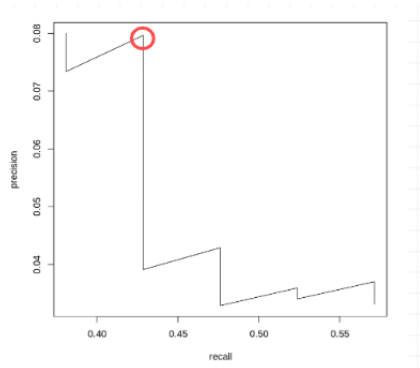
Radial Kernel (after PCA)



1.55	0.5238095	0.03971119
1.60	0.5714286	0.04225352
1.65	0.5714286	0.04109589
1.70	0.5714286	0.04040404
1.75	0.5714286	0.04013378
1.80	0.6190476	0.04276316
1.85	0.6190476	0.04180064
1.90	0.6190476	0.04113924
1.95	0.6190476	0.04000000
2.00	0.6190476	0.03915663

true	pred 0	pred 1
0	361	291
1	8	13

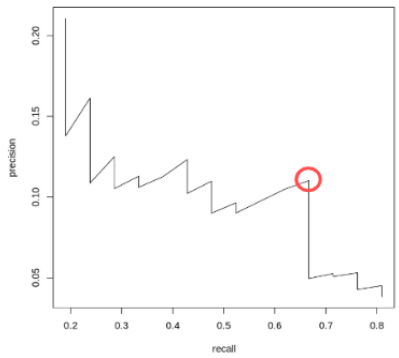
Polynomial Kernel (after PCA)



threshold <dbl>	rec <dbl>	prec <dbl>
0.00	0.3809524	0.08000000
0.05	0.3809524	0.07843137
0.10	0.3809524	0.07766990
0.15	0.3809524	0.07619048
0.20	0.3809524	0.07619048
0.25	0.3809524	0.07339450
0.30	0.4285714	0.07964602
0.35	0.4285714	0.07964602
0.40	0.4285714	0.07894737
0.45	0.4285714	0.07894737
0.50	0.4285714	0.07826087

true	pred 0	pred 1
0	548	104
1	12	9

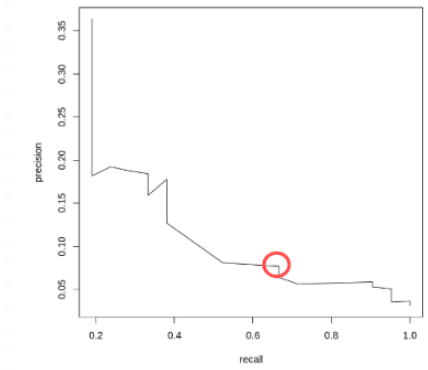
Linear Kernel (before PCA)



2.40	0.5238095	0.09401709
2.45	0.5238095	0.09166667
2.50	0.5238095	0.09016393
2.55	0.6190476	0.10483871
2.60	0.6666667	0.11023622
2.65	0.6666667	0.10852713
2.70	0.6666667	0.10447761
2.75	0.6666667	0.10370370
2.80	0.6666667	0.10000000
2.85	0.6666667	0.09790210

true	pred 0	pred 1
0	539	113
1	7	14

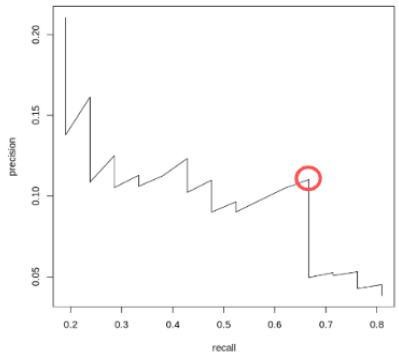
Radial Kernel (before PCA)



0.70	0.3809524	0.14035088
0.75	0.3809524	0.12698413
0.80	0.5238095	0.08088235
0.85	0.6666667	0.07692308
0.90	0.6666667	0.06392694
0.95	0.7142857	0.05639098
1.00	0.9047619	0.05864198
1.05	0.9047619	0.05307263
1.10	0.9523810	0.05063291

true	pred 0	1
0	484	168
1	7	14

Polynomial Kernel (before PCA)



2.40	0.5238095	0.09401709
2.45	0.5238095	0.09166667
2.50	0.5238095	0.09016393
2.55	0.6190476	0.10483871
2.60	0.6666667	0.11023622
2.65	0.6666667	0.10852713
2.70	0.6666667	0.10447761
2.75	0.6666667	0.10370370
2.80	0.6666667	0.10000000
2.85	0.6666667	0.09790210

true	pred 0	1
0	539	113
1	7	14

3.3 Neural Networks

The hyperparameters fixed throughout are listed below:

- optimizer: Adam
- hidden layer activations: Rectified Linear Unit (ReLU)
- output layer activation: Sigmoid

Threshold value (for conversion to class predictions) was unchanged at 0.5 for all neural network models.

Further details on hyperparameter tuning are described in the following paragraphs.

For the first run of hyperparameter tuning, 10-fold cross validation was performed on the training set and the best hyperparameter was chosen by cross-validated accuracy. To find the best number of nodes for the first hidden layer, we fitted and evaluated model performance with 4 to 14 hidden nodes (in increments of 2), and arrived at 14 nodes, which gave an accuracy of 87.7%. Using 14 nodes for the first layer, we proceeded to find the best number of nodes for a second hidden layer, and arrived at 14 nodes, which gave an accuracy of 89.6%, a 1.9% improvement from using a single hidden layer.

Next, we attempted regularization through the implementation of a dropout layer (between the first and second hidden layer) of various degrees between 0.2 and 0.4. In application, this means that some of the

outputs from the first hidden layer are set to zero when it is passed to the second hidden layer, thus not all information (signal or noise) gathered from the inputs will be used to learn the weight parameters.

After experimenting with different degrees of dropout and observing a deterioration in performance by up to 2%, we then looked at batch sizes - 32, 64, 128, and 256 - and found that using a batch size of 32 helped to improve cross-validated performance slightly. Lastly, we trained the model for a longer number of epochs and plotted loss over time (epochs) to check for overfitting. There was one particular fold of training data where the rate of decrease in validation loss diverged quickly from the rate of decrease in training loss within the first 20 epochs. For the remaining 9 folds, validation loss decreased steadily with training loss for the first 20 epochs. Since the validation loss did not sufficiently decrease to justify increasing the training epochs, we will continue to use 20 epochs.

We obtain the following conclusions for the second run of hyperparameter tuning done:

- using 4 hidden nodes for the single layer neural network is best since increasing the number of nodes only gave 1-2% improvement to recall/precision
- using a second hidden layer (of various number of nodes) deteriorated recall and precision
- similarly, regularization deteriorated performance; it was likely unnecessary and harmful due to the fact that the data has been extensively preprocessed and we are using principal components as inputs to the neural network

3.4 Tree-Based Methods

For each tree-based model, we present below the:

- precision and recall values at every threshold value, with the optimal threshold value highlighted
- confusion matrix

Lastly, we attach plots of variable importance from Bagging, Boosting, and RandomForest.

Classification Tree

	threshold	recall_results	precision_results	f2_results
1	0.10	1.0000000	0.08108108	0.3061224
2	0.15	1.0000000	0.08108108	0.3061224
3	0.20	0.8571429	0.11111111	0.3658537
4	0.25	0.8571429	0.11111111	0.3658537
5	0.30	0.8571429	0.11111111	0.3658537
6	0.35	0.7619048	0.14953271	0.4188482
7	0.40	0.7619048	0.14953271	0.4188482
8	0.45	0.7619048	0.14953271	0.4188482
9	0.50	0.7142857	0.27272727	0.5395683
10	0.55	0.7142857	0.27272727	0.5395683
11	0.60	0.7142857	0.27272727	0.5395683
12	0.65	0.7142857	0.27272727	0.5395683
13	0.70	0.7142857	0.27272727	0.5395683
14	0.75	0.7142857	0.27272727	0.5395683
15	0.80	0.7142857	0.27272727	0.5395683
16	0.85	0.3809524	0.25000000	0.3448276
17	0.90	0.3809524	0.25000000	0.3448276

		Actual	
		Bankrupt	No Bankrupt
Predict	Bankrupt	15	40
	No Bankrupt	6	612

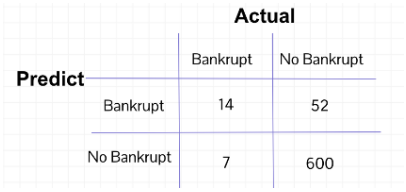
Pruned Classification Tree

	threshold	recall_results	precision_results	f2_results
1	0.10	1.0000000	0.07581227	0.2908587
2	0.15	1.0000000	0.07581227	0.2908587
3	0.20	1.0000000	0.07581227	0.2908587
4	0.25	1.0000000	0.07581227	0.2908587
5	0.30	0.7142857	0.27272727	0.5395683
6	0.35	0.7142857	0.27272727	0.5395683
7	0.40	0.7142857	0.27272727	0.5395683
8	0.45	0.7142857	0.27272727	0.5395683
9	0.50	0.7142857	0.27272727	0.5395683
10	0.55	0.7142857	0.27272727	0.5395683
11	0.60	0.7142857	0.27272727	0.5395683
12	0.65	0.7142857	0.27272727	0.5395683
13	0.70	0.7142857	0.27272727	0.5395683
14	0.75	0.7142857	0.27272727	0.5395683
15	0.80	0.7142857	0.27272727	0.5395683
16	0.85	0.3809524	0.25000000	0.3448276
17	0.90	0.3809524	0.25000000	0.3448276

		Actual	
		Bankrupt	No Bankrupt
Predict	Bankrupt	15	40
	No Bankrupt	6	612

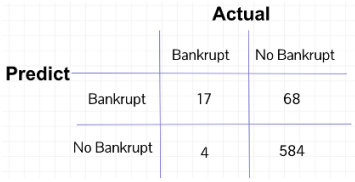
Bagging

	threshold	recall_results	precision_results	f2_results
1	0.10	0.9047619	0.1407407	0.4337900
2	0.15	0.8095238	0.1619048	0.4497354
3	0.20	0.7142857	0.1807229	0.4491018
4	0.25	0.6666667	0.2121212	0.4666667
5	0.30	0.5714286	0.2352941	0.4444444
6	0.35	0.4761905	0.2702703	0.4132231
7	0.40	0.4285714	0.3461538	0.4090909
8	0.45	0.4285714	0.4090909	0.4245283
9	0.50	0.4285714	0.4500000	0.4326923
10	0.55	0.3809524	0.4444444	0.3921569
11	0.60	0.3333333	0.4375000	0.3500000
12	0.65	0.2380952	0.5000000	0.2659574
13	0.70	0.1904762	0.6666667	0.2222222
14	0.75	0.1904762	0.6666667	0.2222222
15	0.80	0.1428571	1.0000000	0.1724138
16	0.85	0.0952381	1.0000000	0.1162791
17	0.90	0.0952381	1.0000000	0.1162791



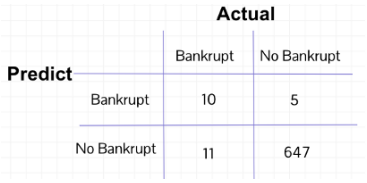
Random Forest

	threshold	recall_results	precision_results	f2_results
1	0.10	0.9523810	0.1257862	0.4115226
2	0.15	0.9523810	0.1886792	0.5263158
3	0.20	0.8095238	0.2000000	0.5029586
4	0.25	0.7142857	0.2205882	0.4934211
5	0.30	0.6666667	0.2916667	0.5303030
6	0.35	0.6190476	0.3611111	0.5416667
7	0.40	0.5238095	0.3928571	0.4910714
8	0.45	0.4285714	0.4285714	0.4285714
9	0.50	0.3809524	0.4444444	0.3921569
10	0.55	0.3333333	0.5000000	0.3571429
11	0.60	0.3333333	0.5000000	0.3571429
12	0.65	0.2857143	0.7500000	0.3260870
13	0.70	0.1904762	0.8000000	0.2247191
14	0.75	0.1428571	1.0000000	0.1724138
15	0.80	0.0952381	1.0000000	0.1162791
16	0.85	0.0952381	1.0000000	0.1162791
17	0.90	0.0952381	1.0000000	0.1162791

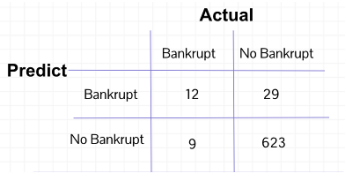


Boosting

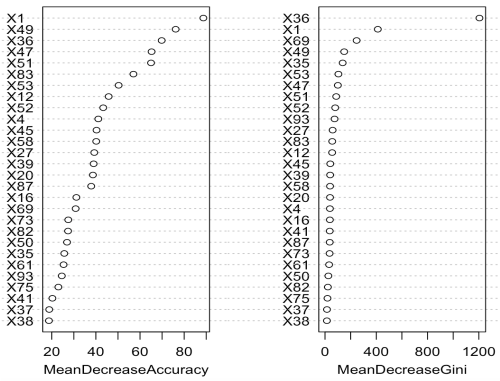
	threshold	recall_results	precision_results	f2_results
1	0.10	0.4761905	0.4347826	0.5029586
2	0.15	0.4761905	0.4347826	0.5029586
3	0.20	0.4761905	0.4761905	0.5029586
4	0.25	0.4761905	0.5000000	0.5029586
5	0.30	0.4761905	0.5000000	0.5029586
6	0.35	0.4761905	0.5263158	0.5029586
7	0.40	0.4761905	0.5555556	0.5029586
8	0.45	0.4761905	0.5555556	0.5029586
9	0.50	0.4761905	0.5555556	0.5029586
10	0.55	0.4761905	0.6250000	0.5029586
11	0.60	0.4761905	0.6666667	0.5029586
12	0.65	0.4285714	0.6428571	0.5029586
13	0.70	0.4285714	0.6923077	0.5029586
14	0.75	0.4285714	0.6923077	0.5029586
15	0.80	0.4285714	0.6923077	0.5029586
16	0.85	0.4285714	0.6923077	0.5029586
17	0.90	0.4285714	0.6923077	0.5029586



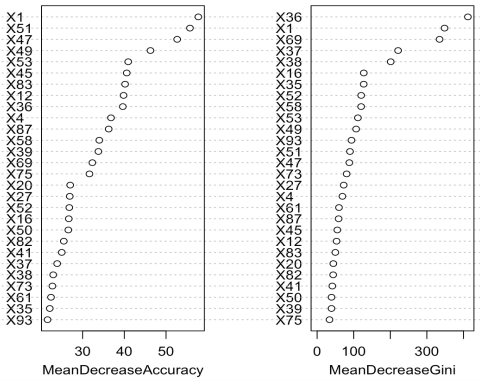
	threshold	recall_results	precision_results	f2_results
1	0.10	0.5714286	0.2926829	0.4672897
2	0.15	0.5714286	0.3636364	0.4672897
3	0.20	0.5238095	0.4230769	0.4672897
4	0.25	0.5238095	0.4583333	0.4672897
5	0.30	0.4761905	0.4347826	0.4672897
6	0.35	0.4761905	0.5000000	0.4672897
7	0.40	0.4761905	0.5263158	0.4672897
8	0.45	0.4761905	0.5263158	0.4672897
9	0.50	0.4761905	0.5882353	0.4672897
10	0.55	0.4761905	0.5882353	0.4672897
11	0.60	0.4285714	0.6000000	0.4672897
12	0.65	0.4285714	0.6000000	0.4672897
13	0.70	0.4285714	0.6923077	0.4672897
14	0.75	0.3809524	0.6666667	0.4672897
15	0.80	0.3333333	0.6363636	0.4672897
16	0.85	0.2857143	0.6000000	0.4672897
17	0.90	0.2380952	0.7142857	0.4672897



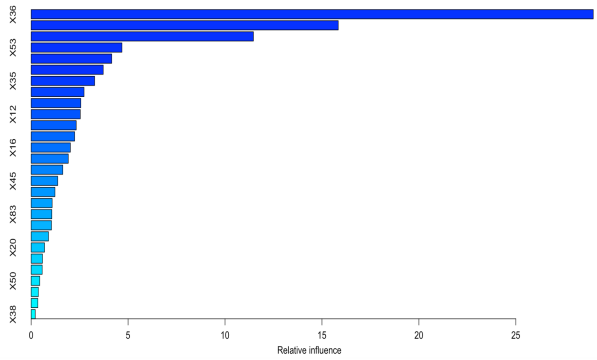
Variables Importance



Bagging Variables Importance



RandomForest Variables Importance



Boosting relative influence plot

var	rel.inf
X36	28.9942145
X1	15.8406659
X69	11.4643850

Boosting relative influence figures

Declaration of use of ChatGPT

ChatGPT was used to generate a draft outline (shown below) and refine the vocabulary used for this report. Several modifications were made to produce the final outline for this report.

I. Introduction
A. Background information on the problem being addressed
B. Research questions or objectives
C. Overview of the machine learning approach used

II. Data Analysis and Preprocessing
A. Description of the dataset(s) used
B. Data cleaning and preprocessing techniques
C. Data visualization and analysis

III. Model Building
A. Description of the machine learning model used
B. Model training and evaluation methods
C. Hyperparameter tuning and optimization techniques

IV. Results and Discussion
A. Performance metrics of the model(s) evaluated
B. Comparison of results to previous studies or benchmarks
C. Analysis of model strengths and weaknesses
D. Discussion of potential applications and future work

V. Conclusion
A. Summary of the main findings
B. Implications and contributions of the study
C. Limitations and potential areas for improvement

VI. References
A. List of sources cited in the report

VII. Appendices
A. Additional figures or tables
B. Detailed descriptions of model architecture or training methods

References

Curtis, G. (n.d.). *Warning Signs of a Company in Trouble*. Investopedia. https://www.investopedia.com/articles/financialcareers/07/warning_signs.asp

Fernando, J. (n.d.). *Debt-to-Equity (D/E) Ratio Formula and How to Interpret It*. Investopedia. https://www.investopedia.com/terms/d/debtequityratio.asp

Hamel, G. (n.d.). *What Are the Causes of Business Bankruptcy?* Small Business - Chron.com. https://smallbusiness.chron.com/causes-business-bankruptcy-49407.html

Investopedia. (n.d.). *Financial Ratios to Spot Companies Headed for Bankruptcy*. Investopedia. https://www.investopedia.com/articles/active-trading/081315/financial-ratios-spot-companies-headed-bankruptcy.asp

McClure, B. (n.d.). *Financial Ratios to Spot Companies in Financial Distress*. Investopedia. https://www.investopedia.com/articles/financial-theory/10/spotting-companies-in-financial-distress.asp

Abhigyan. (2020, July 5). *Feature Selection For Dimensionality Reduction(Embedded Method)*. Medium. https://medium.com/analytics-vidhya/feature-selection-for-dimensionality-reduction-embedded-method-e05c74014aa

López, F. (2021, March 1). *SMOTE: Synthetic Data Augmentation for Tabular Data*. Towards Data Science. https://towardsdatascience.com/smote-synthetic-data-augmentation-for-tabular-data-1ce28090debc

Sauravkaushik8 Kaushik. (2016, December 1). *Introduction to Feature Selection methods with an example (or how to select the right variables?)*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/

chris2016. (2019, December 5). *How to make a precision recall curve in R*. R-bloggers. https://www.r-bloggers.com/2019/12/how-to-make-a-precision-recall-curve-in-r/

Corporate Finance Institute. (n.d.). *Boosting*. Corporate Finance Institute. https://corporatefinanceinstitute.com/resources/data-science/boosting/

Martinez-Taboada, F., & Redondo, J. I. (2020, April 2). *Variable importance plot (mean decrease accuracy and mean decrease Gini)*. Public Library of Science. https://plos.figshare.com/articles/figure/Variable_importance_plot_mean_decrease_accuracy_and_mean_decrease_Gini_/12060105/1