An Analysis Of Bankruptcy Predictions On Taiwanese Companies

Alexander Jeanis, Jay Nguyen, Brian Folkers

Introduction

The dataset for this project comes from the Taiwan Economic Journal. It gives information on several financial variables for different Taiwanese companies, including whether or not the companies went bankrupt. The data was collected between the years 1999 to 2009, although the years themselves are not included as a variable. Through an analysis of the data, it is possible to examine what factors may lead to a company going bankrupt or remaining in business. This project will mainly analyze the data using classification and regression models conducted in R. The main objective will be to determine what factors most affect the likelihood of bankruptcy. To this end, each section will be split into different questions and types of analysis.

Question 1: How accurately can a company defaulting be predicted and which factors are most significant in predicting a company's default?

Accurately predicting bankruptcy is of paramount importance for both investors and lending institutions. By being able to accurately predict default, lending institutions could hypothetically assign credit ratings or target interest rates given default risk, and construct our lending portfolio in an intelligent risk-weighted manner. Additionally, this analysis could lead to the creation of accounting-based lending rules such as never lend/invest in a company with Attribute_Type of >X%.

Question 2: What are the influences of the predictors on bankruptcy?

Exploring which factors contribute most heavily to bankruptcy would allow investors to make informed decisions on the companies within their investment/lending portfolios. While it is probably not practical to use correlation with bankruptcy to form a cross-sectional long/short portfolio (Long Top 10% Least Like to Declare Bankruptcy/Short Top 10% Most Like to Declare Bankruptcy), Investors can use said predictive models in choosing which companies to choose or avoid in potential stock investments/lending decisions.

Question 3: Which models can predict bankruptcy with the best accuracy and why do some of these models work better than others?

With any task, optimizing performance is of paramount importance. In this project, we would like to run our dataset through several classification models, comparing and cross-validating them on key metrics. Ideally, we would like to see 90+% classification accuracy with the least degree of overfitting possible. Additionally, we would like to identify the structural rationale for the performance of each model/model class--that is, why does it make sense that a (KNN, Tree, etc) model accomplishes the task in the best manner.

Methods

The original dataset was split into training (training_unsampled_data) and test (test_original) datasets to be used for all sections. When examining our data we identified 4 main classes of classification algorithms that we wanted to run our data on:

Logistic Regression:

This analysis will use a multiple logistic regression model with all variables. This will help to determine if there is a significant relationship between these factors and a company defaulting. A logistic regression model will work best because the response variable ”Bankruptcy” is a binary variable.

K-nearest neighbor (KNN):

We wanted to try using the K-Nearest Neighbors Algorithm as a simple start to classification with this dataset. The KNN helps to determine how many clusters of the data are most optimal by classifying each data point by how its neighbor is classified. KNN is also computationally efficient and it might have good performance due to the bankruptcy variable having a good possibility of clustering.

Naive Bayes:

The Naive Bayes classification algorithm works under the assumption that the variables in the dataset are not correlated with one another. It is helpful in determining the base accuracy of the dataset. Using Naive Bayes will allow for a classification analysis that assumes that a company's likelihood of Bankruptcy is independent of the financial indicators.

Decision Tree / Random Forest:

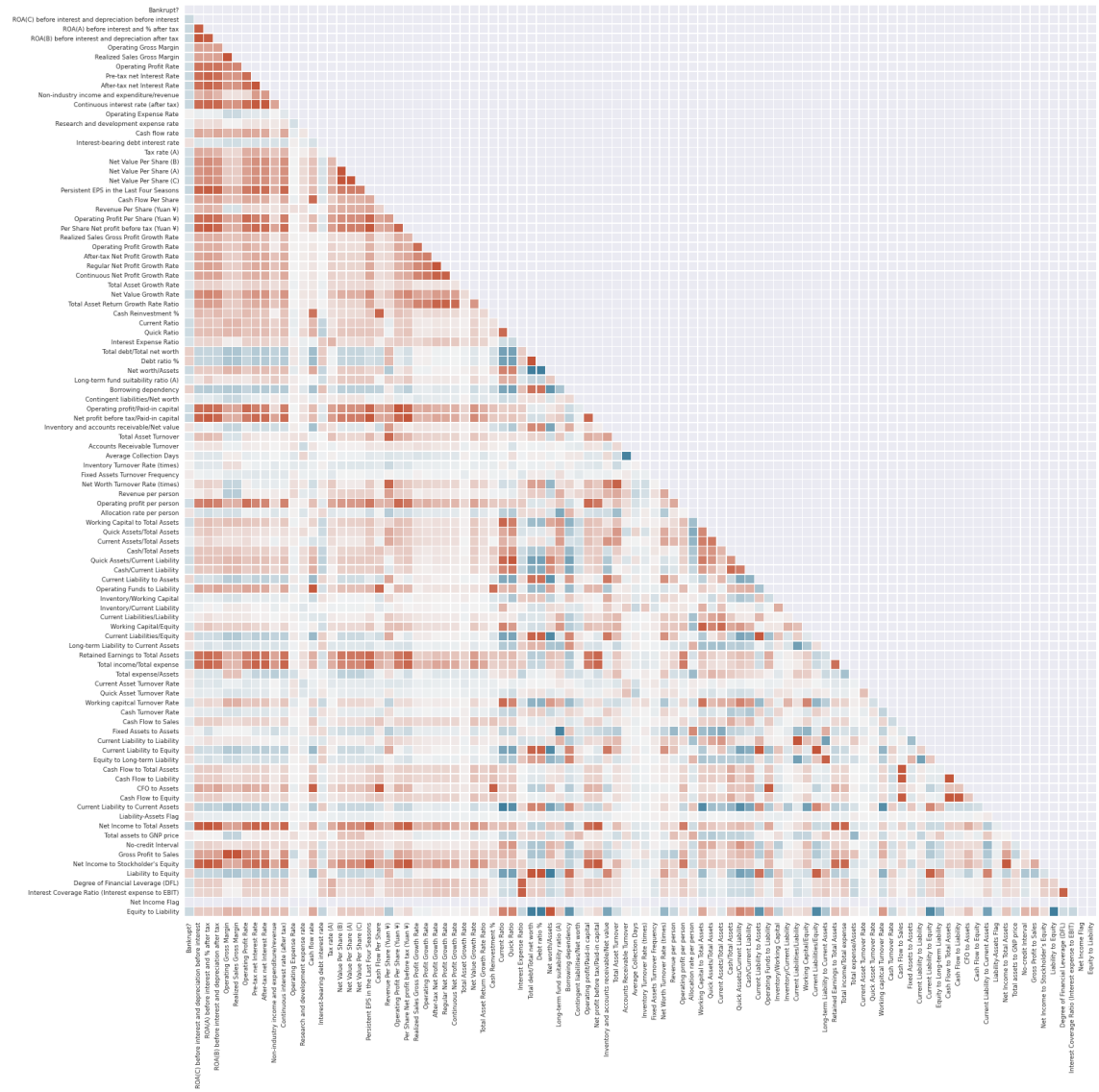
We felt that decision trees (and their extension random forest) exhibited several advantages as a class of models. First, trees are generally simple to understand and interpret as they can be easily visualized. Second, trees do well with both numerical and categorical data.

Furthering trees into random forests makes sense as random forest models exhibit the previous advantages of decision trees along with enhanced ability to handle outliers, and unbalanced / high dimensional data (which our dataset definitely is).

Data Analysis

Preprocessing/Data wrangling

Starting the preprocessing phase, we changed our response to a factor in R. We then checked to see if there is any missing data or duplicates. There ended up being no missing data or duplicates. Since there are too many predictors to produce a pairs plot, we used a heatmap of correlations to get a sense of the data.



There seems to be highly correlated predictors therefore we will look to remove some predictors. First, we will find predictors with near-zero variance to further investigate.

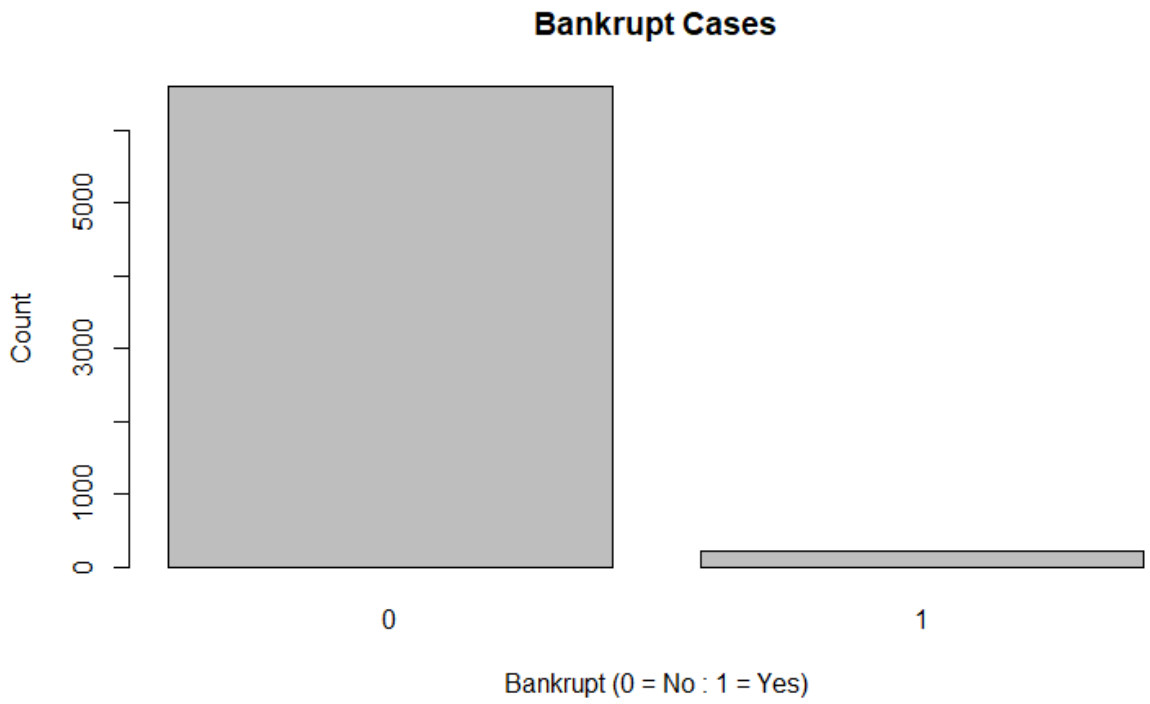
```
> summary(data[,(zeroVarCols)])
   Bankrupt?       Liability-Assets Flag Net Income Flag
 Min.   :0.00000   Min.   :0.000000      Min.   :1
 1st Qu.:0.00000   1st Qu.:0.000000      1st Qu.:1
 Median :0.00000   Median :0.000000      Median :1
 Mean   :0.03226   Mean   :0.001173      Mean   :1
 3rd Qu.:0.00000   3rd Qu.:0.000000      3rd Qu.:1
 Max.   :1.00000   Max.   :1.000000      Max.   :1
>
```

Bankrupt is our response variable. With bankruptcy having little to no variance, we may be dealing with a very imbalanced dataset. We will investigate that later. The Net Income Flag can be removed because it is 1 throughout the whole dataset with no changes. This adds no value to our models. Liability Assets seems to have a similar distribution to Bankrupt. We will investigate this further.

```
> nrow(data[data$`Liability-Assets Flag` == 1 & data$`Bankrupt?` == 1,])
[1] 6
> table(data$`Liability-Assets Flag`)

   0    1
6811    8
>
```

We can see that 6/8 of the companies flagged with having more liability than assets ended up being bankrupt. This is great for interpretation, but it will not help in our models due to having a near-zero variance, so we will be removing this column.

To further evaluate the bankruptcy variable, we generated a plot to see the distribution of data.



There is a large imbalance in the data. We will need to deal with this because it is harder to evaluate our models when the model can predict No and be correct 93% of the time. This is misleading and shows that we will also need to use F1 score instead of accuracy to evaluate model performance. Lastly, we removed predictors with correlation >.999. Due to the nature of Finance metrics, these predictors may be derived from the same metric or are ratios of one another. We then split our dataset into a training and testing set with a 80/20 split. With our training set, we performed upsampling to balance out the Bankrupt response variable. This helps with our model as it makes the bankrupt clusters more apparent, though it may introduce more bias and overfitting.

This is the distribution of our training set after upsampling leaving us with 10512 observations and 93 variables in the training set and 1364 observations and 93 variables.

The original dataset was split into training (training_Unsampled_data) and test (test_original) datasets to be used for all sections.

Logistic Regression:

Running logistic regression, we can see it produced different significant variables.

```
Coefficients:
                                                  Estimate Std. Error z value Pr(>|z|)
(Intercept)                                       4.718e+08  1.970e+08   2.395 0.016600 *
Liability.Assets.Flag1                            4.895e+00  2.342e+00   2.090 0.036613 *
Realized.Sales.Gross.Margin                       4.444e+00  4.767e+00   0.932 0.351255
Pre.tax.net.Interest.Rate                         9.037e+01  5.948e+01   1.519 0.128703
Non.industry.income.and.expenditure.revenue      -8.224e+01  4.686e+01  -1.755 0.079259 .
Operating.Expense.Rate                           -1.059e-11  1.218e-11  -0.870 0.384564
Research.and.development.expense.rate             2.258e-11  1.461e-11   1.545 0.122304
Cash.flow.rate                                   -7.212e+01  1.542e+01  -4.677 2.91e-06 ***
Interest.bearing.debt.interest.rate               8.621e-10  5.465e-10   1.578 0.114662
Tax.rate..A.                                      5.885e-01  2.616e-01   2.249 0.024491 *
Net.Value.Per.Share..B.                          -1.875e+00  2.858e+00  -0.656 0.511747
Cash.Flow.Per.Share                              -3.073e+00  6.276e+00  -0.490 0.624444
Revenue.Per.Share..Yuan...                       -7.429e-08  2.960e-05  -0.003 0.997998
Realized.Sales.Gross.Profit.Growth.Rate           5.783e+01  9.147e+00   6.322 2.58e-10 ***
Operating.Profit.Growth.Rate                     -3.311e+01  2.197e+01  -1.507 0.131763
After.tax.Net.Profit.Growth.Rate                  1.599e+01  4.739e+00   3.374 0.000742 ***
Continuous.Net.Profit.Growth.Rate                -2.097e+01  1.050e+01  -1.998 0.045754 *
Total.Asset.Growth.Rate                           2.300e-11  1.473e-11   1.561 0.118534
Net.Value.Growth.Rate                             2.189e-09  6.390e-06   0.000 0.999727
Total.Asset.Return.Growth.Rate.Ratio             -1.483e+01  4.538e+01  -0.327 0.743859
Cash.Reinvestment..                              -1.711e+01  2.769e+00  -6.178 6.48e-10 ***
```

Significant Variables

```
 [1] "Liability.Assets.Flag1"                    "Realized.Sales.Gross.Margin"
 [3] "Pre.tax.net.Interest.Rate"                 "Non.industry.income.and.expenditure.revenue"
 [5] "Operating.Expense.Rate"                    "Research.and.development.expense.rate"
 [7] "Cash.flow.rate"                            "Interest.bearing.debt.interest.rate"
 [9] "Tax.rate..A."                              "Realized.Sales.Gross.Profit.Growth.Rate"
[11] "Operating.Profit.Growth.Rate"              "After.tax.Net.Profit.Growth.Rate"
[13] "Continuous.Net.Profit.Growth.Rate"         "Total.Asset.Growth.Rate"
[15] "Cash.Reinvestment.."                       "Current.Ratio"
[17] "Quick.Ratio"                               "Total.debt.Total.net.worth"
[19] "Net.worth.Assets"                          "Long.term.fund.suitability.ratio..A."
[21] "Borrowing.dependency"                      "Operating.profit.Paid.in.capital"
[23] "Net.profit.before.tax.Paid.in.capital"     "Inventory.and.accounts.receivable.Net.value"
[25] "Total.Asset.Turnover"                      "Accounts.Receivable.Turnover"
[27] "Inventory.Turnover.Rate..times."           "Fixed.Assets.Turnover.Frequency"
[29] "Net.Worth.Turnover.Rate..times."           "Operating.profit.per.person"
[31] "Working.Capital.to.Total.Assets"           "Quick.Assets.Total.Assets"
[33] "Current.Assets.Total.Assets"               "Cash.Total.Assets"
[35] "Current.Liability.to.Assets"               "Operating.Funds.to.Liability"
[37] "Inventory.Working.Capital"                 "Inventory.Current.Liability"
[39] "Retained.Earnings.to.Total.Assets"         "Total.income.Total.expense"
[41] "Current.Asset.Turnover.Rate"               "Quick.Asset.Turnover.Rate"
[43] "Cash.Turnover.Rate"                        "Cash.Flow.to.Sales"
[45] "Current.Liability.to.Liability"            "Current.Liability.to.Equity"
[47] "Equity.to.Long.term.Liability"             "Cash.Flow.to.Total.Assets"
[49] "Cash.Flow.to.Liability"                    "CFO.to.Assets"
[51] "Cash.Flow.to.Equity"                       "Current.Liability.to.Current.Assets"
[53] "Net.Income.to.Total.Assets"                "Total.assets.to.GNP.price"
[55] "Degree.of.Financial.Leverage..DFL."
```

The second graphic displays all the significant variables that we can use to compare to the important variables achieved from Random Forest.

From the logistic model, we got the following confusion matrix

```
                0           1
0 0.848973607 0.003665689
1 0.122434018 0.024926686
```

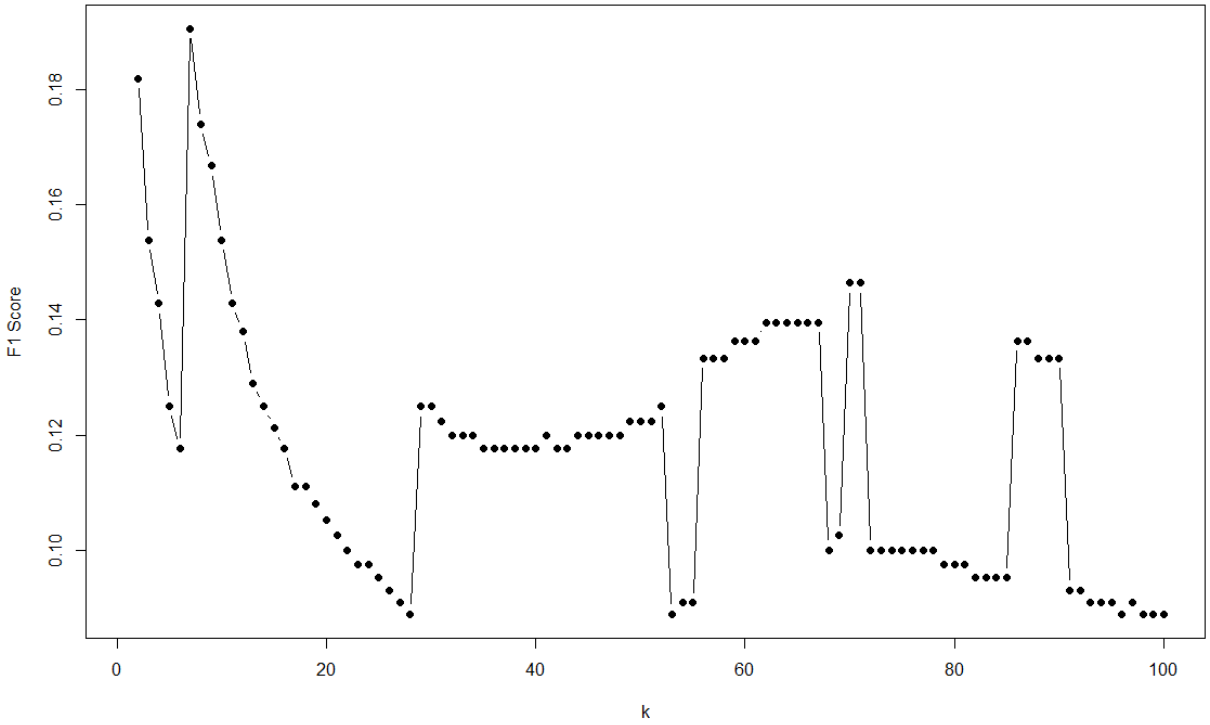
This produces a 28.33% f1 score. We can see that this model struggles to make a decision boundary but this also may be a good score relative to other models we will be testing.

K-nearest neighbor (KNN):

We ran our KNN model with K = sqrt(10512)/2 = 51. We expect a decently performing model but not the best due to an expected bankruptcy cluster in the data.

```
knn_model    0    1
        0 0.56 0.01
        1 0.40 0.02
> |
```

After running, we achieved a f1 score of 10.83% and an accuracy of 58%. This is an unexpected result and may be fixed with parameter tuning for K.



We tested for K up to 100 due to the size of our dataset. We can see the optimal value is the smaller K values, but this is not accurate due to the majority of the dataset being not bankrupt. The “nearest neighbor” will always most likely be nonbankrupt. We can see that at 69 and 70 for K the model performs the best relatively. Though, in general, this model did not perform well. With a 14% f1 score.

Naive Bayes:

We began our construction of the Naive Bayes Model by creating a 10-fold cross validated Naive Bayes with train(data.train,data.train$Bankrupt,method='nb',trControl=trainControl(method='cv',number=10) As seen by the confusion matrix, the Naive Bayes model exhibits 59.9% accuracy and has an F1 score of ~0.328

```
Cross-Validated (10 fold) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction    0    1
         0 50.1 39.9
         1  0.2  9.8

 Accuracy (average) : 0.5989
```

By adding preprocessing in the form of centering and scaling, with preProc = c("center", "scale"),we were able to improve the accuracy slightly to 60.6% and the F1 Score Slighly to ~34.7%.

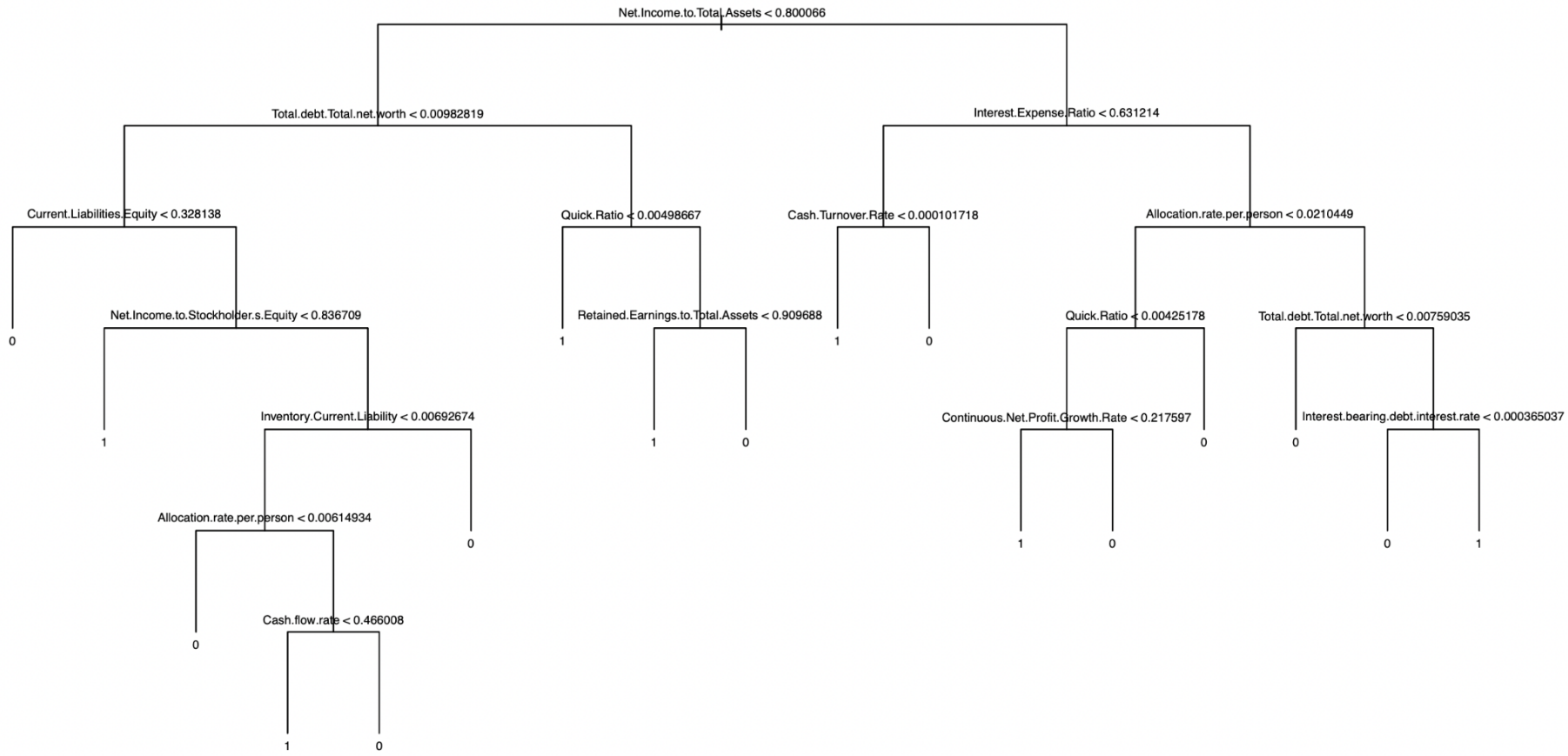
```
Cross-Validated (10 fold) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction    0    1
         0 50.1 39.3
         1  0.1 10.5

 Accuracy (average) : 0.6059
```

Decision Tree / Random Forest classification:

We began our examination of the performance of tree based methods with a basic tree constructed with the following code [data_tree = tree(Bankrupt ~ ., data = data.train)]. We can see from the following summary (below) that the tree selects 17 out of the {XX} variables as terminal nodes. This basic model displays a residual mean deviance: ~0.392 (4118 / 10500)and a misclassification error rate of 0.06745 (709 / 10512) on the training dataset.

```
Classification tree:
tree(formula = Bankrupt ~ ., data = data.train)
Variables actually used in tree construction:
 [1] "Net.Income.to.Total.Assets"         "Total.debt.Total.net.worth"         "Current.Liabilities.Equity"         "Net.Income.to.Stockholder.s.Equity"
 [5] "Inventory.Current.Liability"        "Allocation.rate.per.person"         "Cash.flow.rate"                     "Quick.Ratio"
 [9] "Retained.Earnings.to.Total.Assets"  "Interest.Expense.Ratio"             "Cash.Turnover.Rate"                 "Continuous.Net.Profit.Growth.Rate"
[13] "Interest.bearing.debt.interest.rate"
Number of terminal nodes:  17
Residual mean deviance:  0.3924 = 4118 / 10500
Misclassification error rate: 0.06745 = 709 / 10512
```

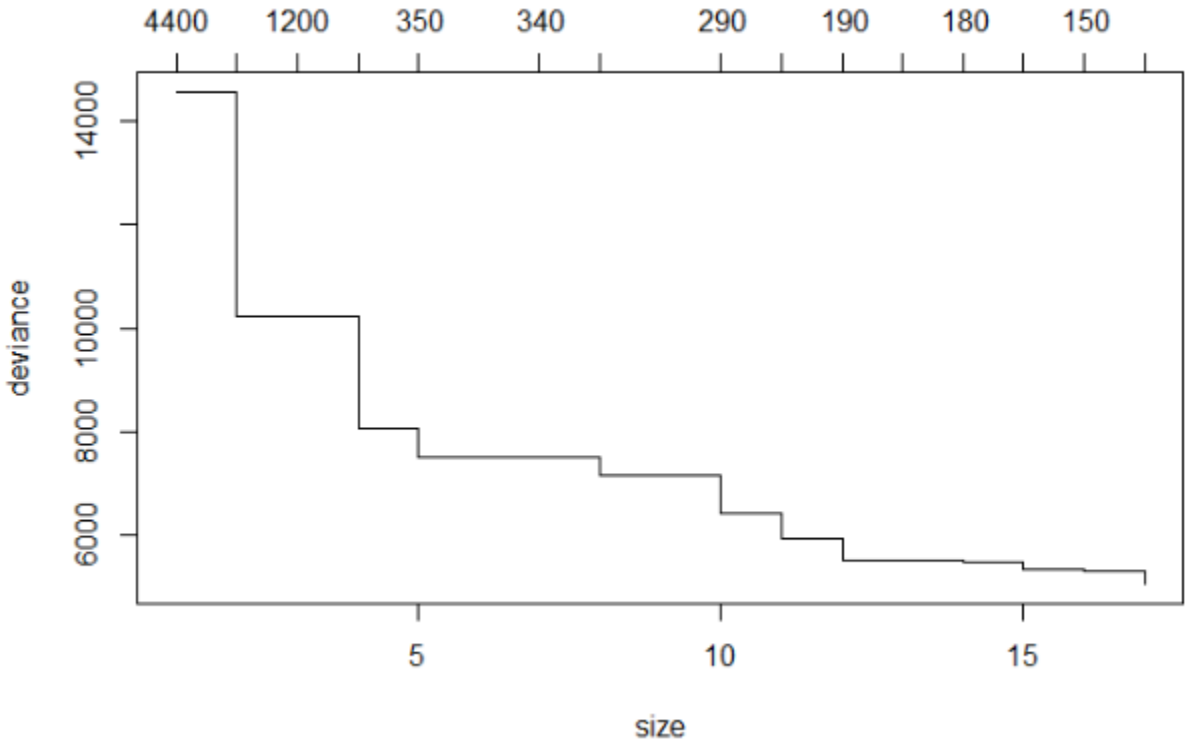
By plotting the tree we can see a visual representation of the tree's decision-making process:


Net.Income.to.Total.Assets < 0.800066
Total.debt.Total.net.worth < 0.00982819
Interest.Expense.Ratio < 0.631214
Current.Liabilities.Equity < 0.328138
Quick.Ratio < 0.00498667
Cash.Turnover.Rate < 0.000101718
Allocation.rate.per.person < 0.0210449
0
Net.Income.to.Stockholder.s.Equity < 0.836709
1
Retained.Earnings.to.Total.Assets < 0.909688
1
0
1
0
Quick.Ratio < 0.00425178
Total.debt.Total.net.worth < 0.00759035
Inventory.Current.Liability < 0.00692674
Continuous.Net.Profit.Growth.Rate < 0.217597
0
Interest.bearing.debt.interest.rate < 0.000365037
1
0
0
1
Allocation.rate.per.person < 0.00614934
0
Cash.flow.rate < 0.466008
1
0

Now, by using [data_tree_pred = predict(data_tree, data.test, type = 'class')] we can test our tree against our withheld testing data. By observing the confusion matrix of outcomes we can see that our classifier was correct ~85.0% of the time, with roughly 96.6% of our classification errors coming from false positives (0.145 / 0.150). Additionally, The basic tree model also exhibits an paltry F1 Score of 29.1%

	Reference	
Prediction	0	1
0	0.818914956	0.005131965
1	0.145161290	0.030791789

Now that we've examined a regular tree, let's run a cross-validation for choosing tree complexity with cv.tree(data_tree) to examine the effect that additional terminal nodes have on model performance. From the plot below, we can see that the largest decreases in deviance occur from K=1 to 5, and we should use the full 17 terminal nodes to achieve the best performance.



Random Forest:

We need to try to further the concept of single trees by creating a random forest model With [randomForest(Bankrupt~., data.train, proximity=TRUE)] After running this model on our test dataset, the random forest showed excellent classification accuracy at 96.6%, and an F1 Score of 34.3% Interestingly enough, when looking at the confusion matrix, we can see that majority of this model's errors come from false negatives (80.4%) rather than false positives like the single decision tree.

	Reference	
Prediction	0	1
0	0.957478006	0.027126100
1	0.006598240	0.008797654

By the importance of variables within the tree, we are able to ascertain which factors are most important to predicting bankruptcy. In the case of basic, non-bagged model, we can see that the Net Income to Total Assets Ratio is the greatest predictor followed closely by Conitouus Interest Rate After Tax. These two measures make sense as they basically represent the amount of free cash flow that a company is generating per dollar of assets along with the severity of the interest rate on a company's debt. Thus it makes intuitive sense that if a company has a low net income relative to assets and a high-interest rate they will be unable to meet their debt obligations and thus go bankrupt.

Basic Random Forest Model, Importance Measures

	MeanDecreaseGini
Net.Income.to.Total.Assets	237.303
Continuous.interest.rate..after.tax.	235.069
Retained.Earnings.to.Total.Assets	201.593
Persistent.EPS.in.the.Last.Four.Seasons	192.525
Total.income.Total.expense	178.661
Quick.Ratio	171.209
Per.Share.Net.profit.before.tax..Yuan…	167.521
Total.debt.Total.net.worth	166.819
Net.worth.Assets	160.569
Net.profit.before.tax.Paid.in.capital	159.293

We can also try bagging the tree to further improve performance. We begin by creating our bagged model with [randomForest(Bankrupt~., data.train, importance=TRUE, mtry=n_pred)] After running the model on testing data, we see a slightly improved performance of 97.1%, but a massive improvement of F1 Score to 53.5%. Looking at the confusion matrix, we can see that most of these improvement comes by cutting down false negatives which are now slightly less than 2% of overall errors representing 65% of total errors.

	Reference	
Prediction	0	1
0	0.95381232	0.01906158
1	0.01026393	0.01686217

As expected, our Bagged Random Forest chose the same two variables as most important (Net Income / Total Assets & Interest Rate After Tax) solidifying their position as the two most important measures in the dataset by far for reasons mentioned earlier.

Bagged Random Forest Model, Importance Measures

	MeanDecreaseGini
Net.Income.to.Total.Assets	1133.896
Continuous.interest.rate..after.tax.	816.242
Total.debt.Total.net.worth	477.264
Interest.Expense.Ratio	351.173
Persistent.EPS.in.the.Last.Four.Seasons	196.929
Retained.Earnings.to.Total.Assets	170.547
Cash.Turnover.Rate	128.945
Inventory.Working.Capital	112.993
Quick.Ratio	92.347
Inventory.Current.Liability	86.661

Conclusion

By applying the knowledge we received in class this semester, we were able to successfully apply statistical learning concepts to answer all of our questions. Looking at model selection, the (bagged) random forest model performed the best by far on the test dataset with a 53.5% F1 Score. Next up, a basic Random Forest Model achieved the second best results 34.3% F1 Score. We can see models that are restricted by dimensions such as KNN perform poorly with a 14% F1 score. With more time, in the next iteration of this data analysis, we can utilize random forest to select important feature to improve the performance of all models especially one's like KNN. This will reduce dimensions and help deal with multi-collinearity in the data. Alternatively, we can run a PCA to help with feature selection though, this turns the models into a black box which would not help us interpret each predictor.

Thus, our best models can predict bankruptcy with upwards of 53.5% F1 Score. Next, looking towards predictive factors, our best models identified Net Income / Total Assets and Continuous Interest Rate After Tax as the two largest factors in predicting bankruptcy. The importance of both factors makes sense as the rate that a company pays on its debt along with its ability to generate income from assets are intrinsically related to its ability to make debt payments and thus avoid bankruptcy. That is, if a company is not generating income and has large interest payments on its debt, then it will logically teeter towards bankruptcy. Finally, when analyzing our model performance, it is not unexpected that Random Forest performed best given the high dimensionality of our data. In the future, to further our research, we would

suggest dividing more into the relationships between each individual predictor, given that some of them share the components (such as numerator or denominator), and possibly use PCA to reduce these factors down, allowing us to attain greater accuracy with non-random forest models. With being said, we enjoyed testing different models on the dataset, tweaking parameters, and confirming our hypotheses during the course of this project.

Acknowledgments

Alexander Jeanis - Introduction / Random Forest / Naive Bayes / Conclusion

Jay Nguyen - Data Wrangling / Logistic Regression / KNN / Methods

Brian Folkers - Introduction / Logistic Regression / Methods