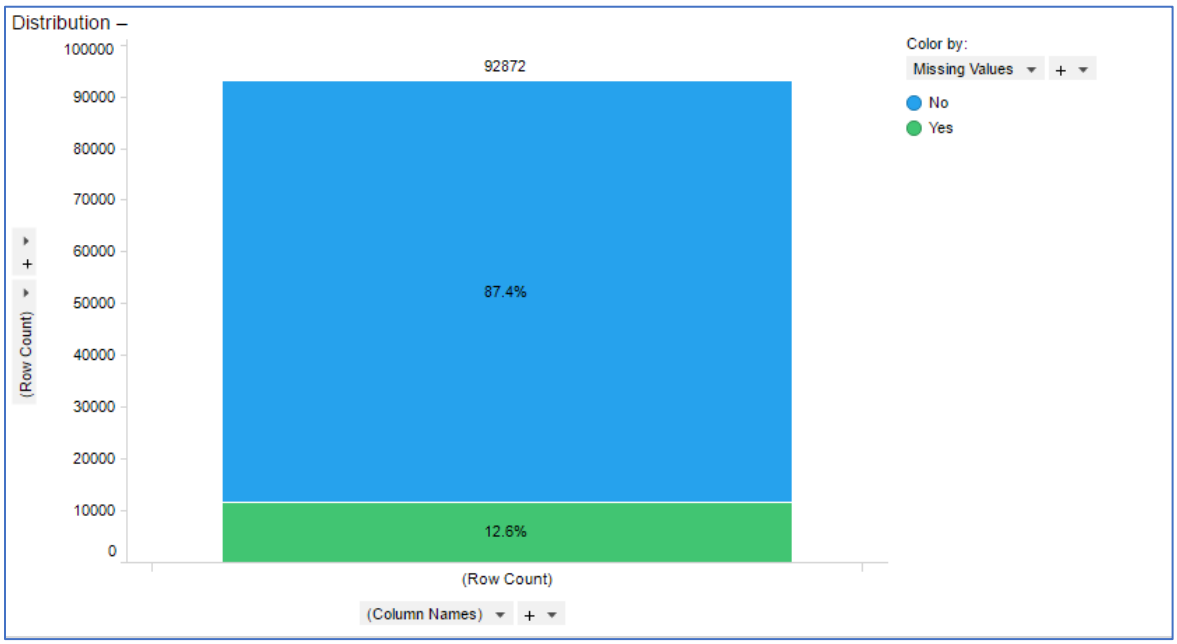
Machine Learning Tools and Corporate Bankruptcy Prediction

The purpose of this report is to discuss the effectiveness of machine learning tools to perform corporate credit risk analysis, predicting corporate bankruptcies and to examine whether these models outperform traditional prediction tools like Logistic Regression, Multiple Discriminant Analysis (MDA) or Altman's Z Score. For this project, we use predictor variable data (EBIT, Asset Growth Rate, Sales Growth Rate, EPS etc.) from companies between 1979 and 2013 to develop and test predictive models. After rebalancing the data (0.6% bankruptcy rate in the data set), we created some new features and then used the data through 3 different models which are *Logistic Regression, Random Forest and Support Vector Machine*. The models were trained on a sub-set of the data (70%) and then tested on the remaining data to determine its performance at predicting bankruptcy. We will discuss our model building process in detail in the remainder of this report and conclude with the *finding that modern machine learning models tend to outperform traditional ones like logistic regression, thus significantly improving the corporate credit risk assessment and prediction process.*

Data Pre-Processing

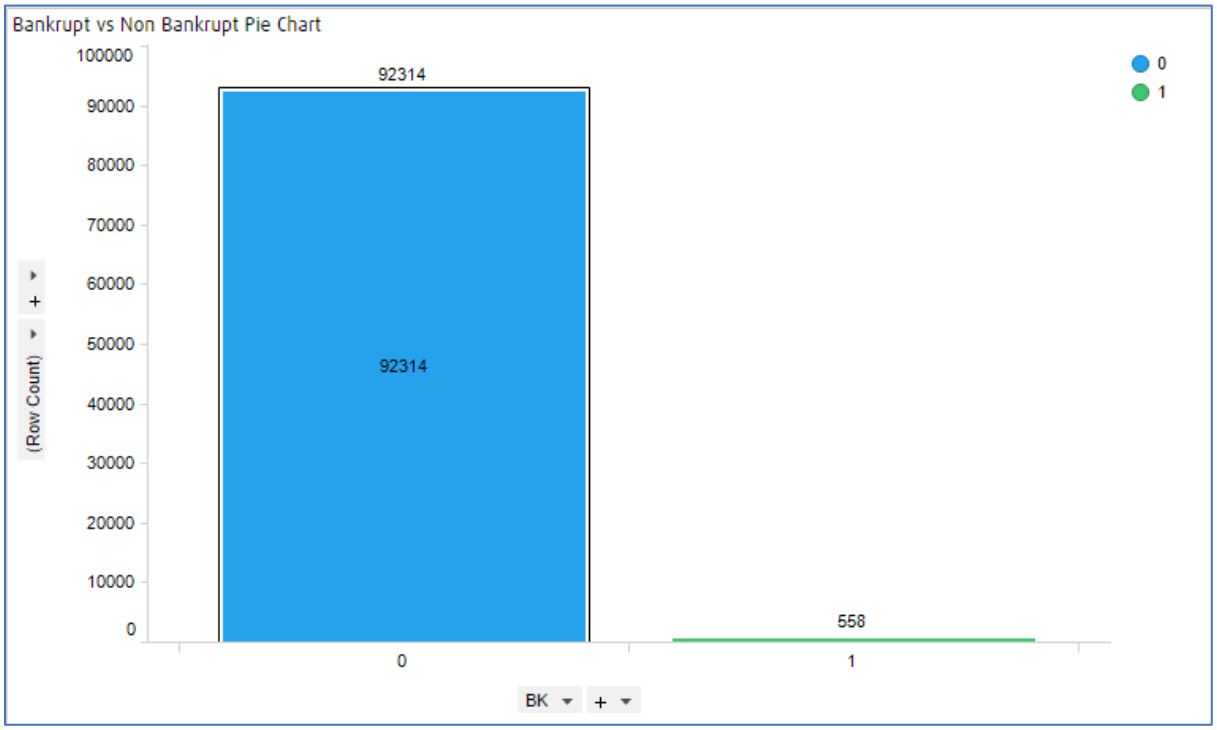
We started our model development process by performing data exploration activities and noticed two major issues:

1) **Missing Values:** There are a lot of missing values in our dataset, in fact, over 12.5% instances have missing data (as can be seen in the figure below):



Due to the large number of records impacted by this, we could not drop such instances, instead decided to impute the missing values using the **MICE** package in R (ppm method).

2) **Imbalanced Data Set:** The data set was extremely imbalanced, with only 0.6% of the instances corresponding to the bankrupt class. Most machine learning algorithms work best when there are an equal number of observations in each class, thus having a data set that is so imbalanced would negatively impact the performance of our models. To tackle this, we decided to up-sample the data (using the ROSE package in R) by randomly selecting an observation from the minority class with replacement, for a portion of the observations in the majority class. After up-sampling, bankruptcy instances accounted for approximately 25% of our data (the number of instances became 123,085).



The last step in the data pre-processing phase was to create new features. This was done through the use of dummy variables - creating sub-categories for each of the features - and by taking log-transformations. The table below summarizes these feature engineering activities:

Feature	**Engineering Activity**	**Feature**	**Engineering Activity**
Date Year -Fiscal	Created 4 dummy variables to group values for this feature year_1980: for year before 1989 year_1990: year 1990-1999 year_2000: year 2000-2009 year_2010: year 2010-current	Tobin's Q	Take log transformation for Tobin's Q (can't take log for other numerical predictors, because others include negative values)

EPS	Create 5 dummy variables for EPS feature EPS1: EPS <= -4 EPS2: EPS > -4 and EPS <= -0.5 EPS3: EPS > -0.5 and EPS < 0 EPS4: EPS >= 0 and EPS <= 1 EPS5: EPS > 1	Liquidity	Created 6 buckets for Liquidity feature Liquidity1: Liquidity <= -0.2 Liquidity2: Liquidity > -0.2 and Liquidity <=0 Liquidity3: Liquidity > 0 and Liquidity <= 0.15 Liquidity4: Liquidity > 0.15 and Liquidity <= 0.35 Liquidity5: Liquidity > 0.35
Profitability	Create 6 new dummy variables Profitability1: Profitability <= -5 Profitability2: Profitability > -5 and Profitability <= -0.4 Profitability3: Profitability > -0.4 and Profitability <= 0 Profitability4: Profitability > 0 and Profitability <= 0.15 Profitability5: Profitability > 0.15 and Profitability <= 0.4 Profitability6: Profitability > 0.4	Productivity	Create 6 dummy variables Productivity1: Productivity <= -0.6 Productivity2: Productivity > -0.6 and Productivity <= -0.1 Productivity3: Productivity > -0.1 and Productivity <= 0 Productivity4: Productivity > 0 and Productivity <= 0.07 Productivity5: Productivity > 0.07 and Productivity <= 0.15 Productivity6: Productivity > 0.15
Leverage Ratio	Created 5 dummy variables for Leverage Ratio Leverage1: Leverage Ratio <= -1 Leverage2: Leverage Ratio > -1 and Leverage Ratio <= 0 Leverage3: Leverage Ratio > 0 and Leverage Ratio <= 0.3 Leverage4: Leverage Ratio > 0.3 and Leverage Ratio <= 1	Asset Turnover	Create 5 buckets / dummy variables AssetTurnover1: Asset Turnover <= 0 AssetTurnover2: Asset Turnover > 0 and Asset Turnover <= 0.45 AssetTurnover3: Asset Turnover > 0.45 and Asset Turnover <= 1 AssetTurnover4: Asset Turnover > 1 and Asset Turnover <= 1.65

	Leverage5: Leverage Ratio > 1		AssetTurnover5: Asset Turnover > 1.65
Operational Margin	Create 5 variables for Operational Margin Margin1: Operational Margin <= -0.5 Margin2: Operational Margin > -0.5 and Operational Margin <= 0 Margin3: Operational Margin > 0 and Operational Margin <= 0.07 Margin4: Operational Margin > 0.07 and Operational Margin <= 0.12 Margin5: Operational Margin > 0.12	Return on Equity	Created 4 new Dummy Variables Return1: Return on Equity <= -0.3 Return2: Return on Equity > -0.3 and Return on Equity <= 0 Return3: Return on Equity > 0 and Return on Equity <= 0.06 Return4: Return on Equity > 0.06
Market to Book Ratio	Creating 5 Dummy variables Market1: Market Book Ratio <= 0 Market2: Market Book Ratio > 0 and Market Book Ratio <= 15 Market3: Market Book Ratio > 15 and Market Book Ratio <= 60 Market4: Market Book Ratio > 60 and Market Book Ratio <= 230 Market5: Market Book Ratio > 230	Assets Growth	Created buckets for Assets Growth: AssetsGrowth1: Assets Growth <= -0.2 AssetsGrowth2: Assets Growth > -0.2 and Assets Growth <= 0 AssetsGrowth3: Assets Growth > 0 and Assets Growth <= 0.18 AssetsGrowth4: Assets Growth > 0.18
Sales Growth	Created buckets/dummy variables for Assets Growth: Sales1: Sales Growth <= -0.16 Sales2: Sales Growth > -0.16 and Sales Growth <= 0	Employee Growth	Created 4 new dummy variables: Employee1: Employee Growth <= -0.1 Employee2: Employee Growth > -0.1 and Employee Growth <=0 Employee3: Employee Growth > 0 and

	Sales3: Sales Growth > 0 and Sales Growth <= 0.2 Sales4: Sales Growth > 0.2		Employee Growth <=0.13 Employee4: Employee Growth > 0.13

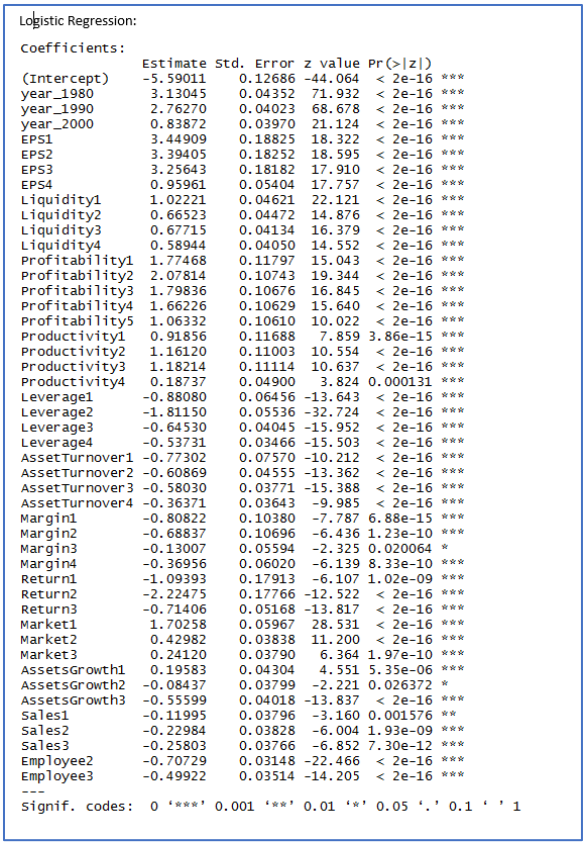
The dataset was then divided into train (70%) and test (30%) through random sampling. The former was used to train our machine learning models (Random Forest, SVM and Logistic Regression), while the later was used for hyperparameter tuning, testing and model evaluation.

Model Methodology

To begin our model building process, we set a seed to ensure reproducible results. As mentioned earlier, the **train** dataset of 87,085 instances was used to train the models and for hyperparameter tuning, while the **test** dataset was used to validate model performance and do a comparative analysis on our models.

Logistic Regression

To fit a logistic regression model, we use the **glm** function to which we feed all our predictor and response (BR) variables. By setting the family argument to 'binomial', we tell glm to fit a logistic regression model (instead of the of the many other models that can be fit to the glm). We then summarize the results of the regression to see the estimate, standard errors, z-score and p-values on each of our coefficients. As seen in the output (of the summary function) below, it seems as if all the coefficients are significant at the 5% significance level.



Logistic Regression:

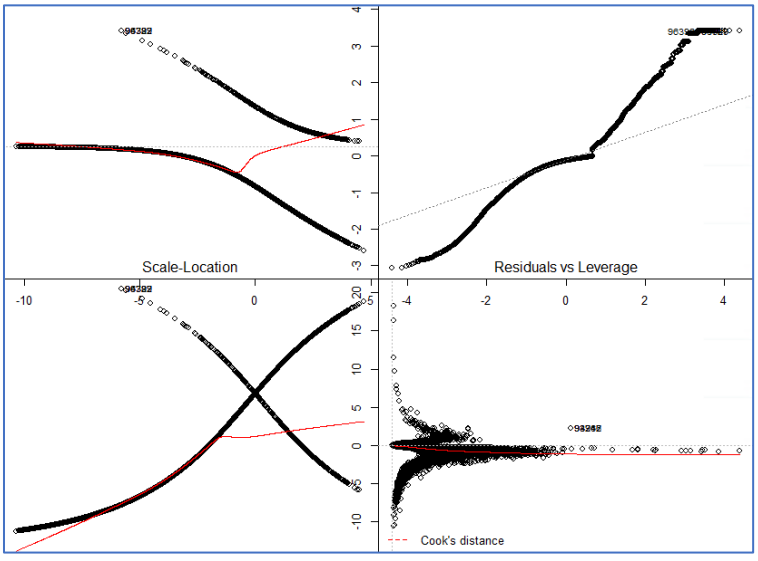
```
Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    -5.59011    0.12686 -44.064  < 2e-16 ***
year_1980       3.13045    0.04352  71.932  < 2e-16 ***
year_1990       2.76270    0.04023  68.678  < 2e-16 ***
year_2000       0.83872    0.03970  21.124  < 2e-16 ***
EPS1            3.44909    0.18825  18.322  < 2e-16 ***
EPS2            3.39405    0.18252  18.595  < 2e-16 ***
EPS3            3.25643    0.18182  17.910  < 2e-16 ***
EPS4            0.95961    0.05404  17.757  < 2e-16 ***
Liquidity1      1.02221    0.04621  22.121  < 2e-16 ***
Liquidity2      0.66523    0.04472  14.876  < 2e-16 ***
Liquidity3      0.67715    0.04134  16.379  < 2e-16 ***
Liquidity4      0.58944    0.04050  14.552  < 2e-16 ***
Profitability1  1.77468    0.11797  15.043  < 2e-16 ***
Profitability2  2.07814    0.10743  19.344  < 2e-16 ***
Profitability3  1.79836    0.10676  16.845  < 2e-16 ***
Profitability4  1.66226    0.10629  15.640  < 2e-16 ***
Profitability5  1.06332    0.10610  10.022  < 2e-16 ***
Productivity1   0.91856    0.11688   7.859 3.86e-15 ***
Productivity2   1.16120    0.11003  10.554  < 2e-16 ***
Productivity3   1.18214    0.11114  10.637  < 2e-16 ***
Productivity4   0.18737    0.04900   3.824 0.000131 ***
Leverage1      -0.88080    0.06456 -13.643  < 2e-16 ***
Leverage2      -1.81150    0.05536 -32.724  < 2e-16 ***
Leverage3      -0.64530    0.04045 -15.952  < 2e-16 ***
Leverage4      -0.53731    0.03466 -15.503  < 2e-16 ***
AssetTurnover1 -0.77302    0.07570 -10.212  < 2e-16 ***
AssetTurnover2 -0.60869    0.04555 -13.362  < 2e-16 ***
AssetTurnover3 -0.58030    0.03771 -15.388  < 2e-16 ***
AssetTurnover4 -0.36371    0.03643  -9.985  < 2e-16 ***
Margin1        -0.80822    0.10380  -7.787 6.88e-15 ***
Margin2        -0.68837    0.10696  -6.436 1.23e-10 ***
Margin3        -0.13007    0.05594  -2.325 0.020064 *
Margin4        -0.36956    0.06020  -6.139 8.33e-10 ***
Return1        -1.09393    0.17913  -6.107 1.02e-09 ***
Return2        -2.22475    0.17766 -12.522  < 2e-16 ***
Return3        -0.71406    0.05168 -13.817  < 2e-16 ***
Market1         1.70258    0.05967  28.531  < 2e-16 ***
Market2         0.42982    0.03838  11.200  < 2e-16 ***
Market3         0.24120    0.03790   6.364 1.97e-10 ***
AssetsGrowth1   0.19583    0.04304   4.551 5.35e-06 ***
AssetsGrowth2  -0.08437    0.03799  -2.221 0.026372 *
AssetsGrowth3  -0.55599    0.04018 -13.837  < 2e-16 ***
Sales1         -0.11995    0.03796  -3.160 0.001576 **
Sales2         -0.22984    0.03828  -6.004 1.93e-09 ***
Sales3         -0.25803    0.03766  -6.852 7.30e-12 ***
Employee2      -0.70729    0.03148 -22.466  < 2e-16 ***
Employee3      -0.49922    0.03514 -14.205  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

To select the best model and model predictors, we used the **stepwise** method (on the output of the glm method earlier). The **MASS** library in R was used to run this method and a combination of both the backward and forward procedures (in AIC) were used to select the best predictors. The performance of the best logit model can be seen in the images below (along with the error plots). A cutoff value of **0.6** was used for class prediction.

Logistic Regression

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 17779   221
         1  9232  8768

               Accuracy : 0.7374
                 95% CI : (0.7328, 0.742)
    No Information Rate : 0.7503
    P-Value [Acc > NIR] : 1

                  Kappa : 0.4748

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.9754
            Specificity : 0.6582
         Pos Pred Value : 0.4871
         Neg Pred Value : 0.9877
             Prevalence : 0.2497
         Detection Rate : 0.2436
   Detection Prevalence : 0.5000
      Balanced Accuracy : 0.8168

       'Positive' Class : 1
```

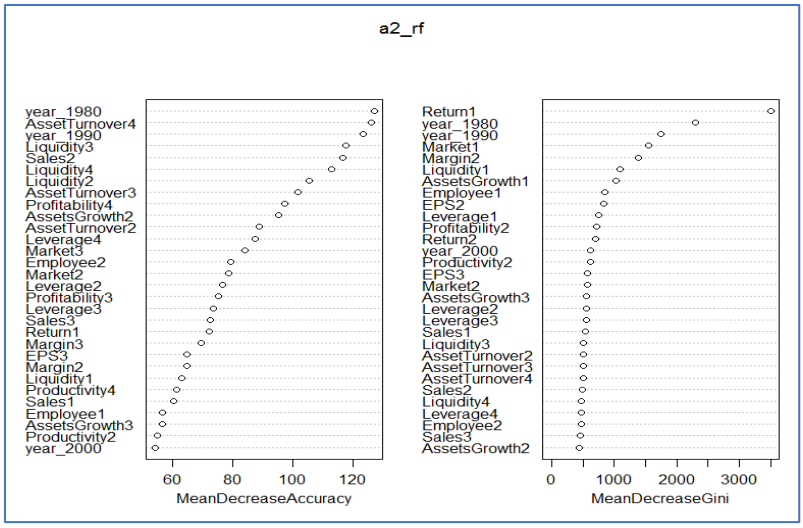
The accuracy of the logit model is 73.74%, with a **recall** of more than 97% (true positive rate) and **specificity** (true negative rate) of just under 66%. The logistic regression model is good at predicting bankruptcy in cases where a company is going to go bankrupt.



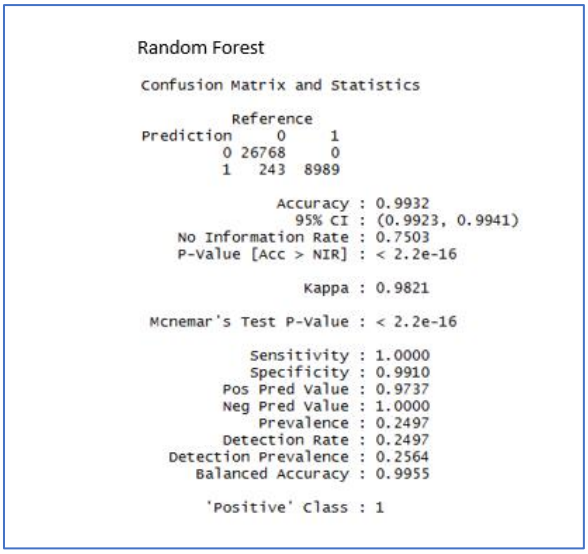
Random Forest

We ran the Random Forest Model twice on this dataset using the **Radom Forest** library in R. Initially the model was run on all the features (with BK as the response variable) on the train dataset. We ran the

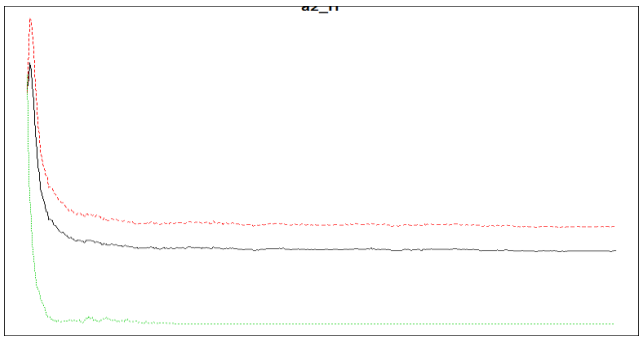
variable importance plot on the output, which provided us with the importance score of each of these features (see image below).



The variables at the top (chart on the left) are the most important variables, as shown by the decreased accuracy of the model, if these variables are removed. Based on these scores, some of the least important variables were dropped from the model (Market4 and Employee3) and the RF model was re-run using the remaining predictors on the train data. After tuning hyperparameters, we came up with the final model which was run on the test data. The images below show the output of this model:



The **accuracy** of this model is 99.32% with a perfect **sensitivity** and a 99.1% **specificity**. The results of the RF model seem very promising as its able to (almost) perfectly predict corporate bankruptcy.



Support Vector Machine (SVM)

SVM was the final model that we ran on the data. We initially ran it on all the predictors on the **train** data, setting the type as 'C-Classification' and using a 'linear' kernel. One of the outputs of the model was weighted vectors of each feature. We ranked these to highlight the least important features which were then removed from the model (Margin3, Market3, Market4, AssetsGrowth2, Sales1, Sales2 and Employee1). After removing these variables, final model was built on the train dataset using the remaining features (model tuning was also done during this stage which will be discussed in the Hyperparameter tuning section).

After hyperparameter tuning, the final model was run using the test dataset. Predictions based on this model, confusion matrix and error rates can be seen in the image below:

```
SVM

Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 25582  2295
         1  1429  6694

               Accuracy : 0.8966
                 95% CI : (0.8934, 0.8997)
    No Information Rate : 0.7503
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7148

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.7447
            Specificity : 0.9471
         Pos Pred Value : 0.8241
         Neg Pred Value : 0.9177
             Prevalence : 0.2497
         Detection Rate : 0.1859
   Detection Prevalence : 0.2256
      Balanced Accuracy : 0.8459

       'Positive' Class : 1
```

The **accuracy** of this model is close to 90% with a high **specificity** of 94.71% and a 74.47% **sensitivity**. The SVM model seems to be good at picking out firms that do not go bankrupt but fails to catch almost 25% of the bankrupt firms.

The results of the three models are summarized in the table below:

	Accuracy	Sensitivity	Specificity	AUC	F1 score	Error Rate
Logistic	0.7374	0.9754	0.6582	0.9458	0.6497	0.2626
Random Forest	0.9932	1	0.991	0.9955	0.9867	0.0068
SVM	0.8967	0.7447	0.9473	0.846	0.7824	0.1034

In the case of corporate bankruptcies, it is essential that any machine learning model should have high precision and recall (sensitivity). This is because the cost of missing a bankruptcy prediction could potentially be very high (for example, the model can be used for credit analysis or investment in fixed income securities issued by the company). Incorrectly concluding that a company is financially secure (when it is not) could then result in a large loss if an investment was subsequently made in the company. On the flip side, incorrectly predicting that a company would go bankrupt would at worse result in a missed investment opportunity or early exit. The best measure to evaluate such models would be the F1 score, which is a combination of these 2 measures:

F1 – (2*Recall * Precision) / (Recall + Precision)

The RF model seems to be the best choice, since it has the highest F1 score (it is outperforming the other two on every measure in the table above).

Hyperparameter Tuning

As mentioned previously, we randomly split our data into two partitions (test and train) and used the train partition to tune the hyperparameters of the model. For the RF and SVM models, we tried using grid search and auto tuning (respectively) for this purpose. Due to the size of the data these models ran for over 3 days without giving any results, so we instead decided to manually try different hyperparameter values on the models to try and optimize it. The table below shows the hyperparameters that we varied, and their final values used in the models:

Model	Hyperparameter's varied	Final Hyperparameter Values
Logistic	Direction: 'both', 'forward' and backward Cutoff value	Direction: both Cutoff: 0.6
Random Forest	Importance Proximity Cutoff Type	Importance: True Proximity: False Cutoff: c (0.6, 0.4) Type: 'classification'
SVM	Type Kerner Cost	Type: 'C-classification' Kernel: 'linear' Cost: 0.1

Results

The goal of this experiment was to determine whether machine learning based tools can do a better job at predicting corporate bankruptcy, than traditional methods like Altman's Z-Score (MDA). We also wanted to assess whether methods like Random Forest and SVM are better predictors (in this case) than older techniques like logistic regression. Since this is a classification exercise, we used confusion matrix-based measures to evaluate model performance. Specifically, due to the importance of identifying **true positives** correctly, we used the F1 measure. To compare the performance of our models with the traditional MDA based methods, we calculated the Altman's Z-Score on each instance of our dataset and then defined bankruptcy to be the case where the score was below the cutoff value of **1.81**. The result of this exercise was compared against the response variable in the dataset to calculate the accuracy of this method (see the attached Excel file).

Based on the results of our experiment, we can conclude that machine learning (ML) algorithms perform significantly better than traditional methods like Altman's Z. The accuracy of each of our ML based methods was higher than the accuracy of Altman's Z, which proved to be 50% on our dataset. We can also conclude that modern methods out-perform logistic regression based on both the accuracy and F1 score. The output of our ML models also shows that there are some identifiable factors that increase the risk of bankruptcy (bold ones are the most important):

- ✓ **Negative EPS**
- ✓ **Low profitability (profitability ratio below 0.4)**
- ✓ Lower Liquidity
- ✓ Lower Productivity
- ✓ High Leverage Ratio
- ✓ High Asset Turnover Ratio
- ✓ Market to book ratios below 0
- ✓ **Age of company (older companies have a higher tendency to go bankrupt)**

These ML models can have several important uses in the financial world to improve on current corporate credit risk assessment practices. On the capital markets side, they can be used by **bond rating agencies** to better assess the credit risk of a firm and in turn provide more accurate ratings. On the buy side, Hedge Funds that specialize in distressed security investments can use such models to determine whether the market is accurately pricing the securities and deduce whether the debt securities are over or under priced. On the wholesale side, banks can use these models to optimize the rates that they charge their corporate customers based on a more accurate assessment of their credit risk.

Before we put our model into production, we should conduct further testing on other datasets as well, so ascertain that similar results are obtained there. We should also test our models on newer data, as this dataset is several years old.