MA5790

Bankruptcy Prediction of Companies in Taiwanese Stock Market

By - Group 8
Sri Rayaleswar Mondrety
Venkat Sairam Veeranki

Table of Contents:

Abstract:

Bankruptcy is a legal process through which companies can undergo reorganization and seek relief from creditors. The bankruptcy procedures for large companies generally fall under Chapter 7 or 11. This project focuses on predicting the bankruptcy of companies listed on the Taiwanese stock exchange, utilizing a dataset that includes information from the 2008 financial crisis. Our interest lies in analyzing the dataset sourced from the UCI ML repository and applying various linear and nonlinear classification models to predict whether a company is likely to go bankrupt.

Introduction:

The UCI ML repository dataset spans from 1999 to 2009, capturing the aftermath of the 2008 financial crisis. It encompasses data from 6819 companies, featuring 96 predictors. This project involves modeling several classification models after preprocessing the data, which includes removing variables with near-zero variance and addressing collinearity through PCA. The dataset is subsequently split into training and testing sets. The resulting models are evaluated on the testing dataset using the ROC metric, and two optimal models are selected. The ultimate goal is to identify the most optimal model for predicting bankruptcy.

Variable Introduction and Definitions:

The dataset comprises 96 features, including a mix of financial ratio metrics such as Return on Assets (ROAs), Gross Profits, Operating and Net Income, Expenses, Cash Flows, Taxes, Growth Rates, Debt, Turnover, Revenue, Liability, and more. Two essential criteria govern the inclusion of companies in this study: they must have a minimum of three years of complete public information before undergoing a financial crisis, and comparisons are restricted to companies with a sufficient number of counterparts in the same industry.

The dataset primarily encompasses manufacturing companies, including industrial and electronic companies (346), service industries (39), and others (93). It is essential to note the dataset's imbalanced nature, with 6,819 samples, where 6,599 belong to the negative class (non-bankrupt) and only 220 to the positive class (bankrupt). This imbalance poses a challenge for generalization in predicting bankruptcy outcomes. The analysis will delve into the intricacies of these financial features and employ machine learning models to develop a predictive model for bankruptcy detection, aiding companies in making informed decisions to mitigate financial distress.

Predictor	**Description**	**Type**
Bankrupt	Binary indicator (1 for bankruptcy, 0 for non-bankruptcy)	Categorical
ROA.C..before.interest.and.depreciation. before.interest	Return on Assets before interest and depreciation	Numeric
ROA.A..before.interest.and...after.tax	Return on Assets before interest and after-tax	Numeric
ROA.B..before.interest.and.depreciation.after.tax	Return on Assets before interest and depreciation after tax	Numeric
Operating.Gross.Margin	Gross margin derived from operating activities	Numeric
Realized.Sales.Gross.Margin	Realized gross margin from sales	Numeric
Operating.Profit.Rate	Operating profit rate	Numeric
Pre.tax.net.Interest.Rate	Pre-tax net interest rate	Numeric
After.tax.net.Interest.Rate	After-tax net interest rate	Numeric
Non.industry.income.and.expenditure.revenue	Non-industry income and expenditure revenue	Numeric
Continuous.interest.rate..after.tax	Continuous interest rate after tax	Numeric
Operating.Expense.Rate	Operating expense rate	Numeric
Research.and.development.expense.rate	Research and development expense rate	Numeric
Cash.flow.rate	Cash flow rate	Numeric
Interest.bearing.debt.interest.rate	Interest-bearing debt interest rate	Numeric
Tax. rate..A.	Tax rate A	Numeric
Net.Value.Per.Share..B.	Net value per share (class B)	Numeric
Net.Value.Per.Share..A.	Net value per share (class A)	Numeric

Net.Value.Per.Share..C.	Net value per share (class C)	Numeric
Persistent.EPS.in.the.Last.Four.Seasons	Persistent earnings per share in the last four seasons	Numeric
Cash.Flow.Per.Share	Cash flow per share	Numeric
Revenue.Per.Share..Yuan...	Revenue per share in Yuan	Numeric
Operating.Profit.Per.Share..Yuan...	Operating profit per share in Yuan	Numeric
Per.Share.Net.profit.before.tax..Yuan...	Per-share net profit before tax in Yuan	Numeric
Realized.Sales.Gross.Profit.Growth.Rate	Growth rate of realized gross profit from sales	Numeric
Operating.Profit.Growth.Rate	Growth rate of operating profit	Numeric
After.tax.Net.Profit.Growth.Rate	Growth rate of after-tax net profit	Numeric
Regular.Net.Profit.Growth.Rate	Regular growth rate of net profit	Numeric
Continuous.Net.Profit.Growth.Rate	Continuous growth rate of net profit	Numeric
Total.Asset.Growth.Rate	Growth rate of total assets	Numeric
Net.Value.Growth.Rate	Growth rate of net value	Numeric
Total.Asset.Return.Growth.Rate.Ratio	Ratio of the growth rate of total asset return	Numeric
Cash.Reinvestment..	Cash reinvestment rate	Numeric
Current.Ratio	Current ratio	Numeric
Quick.Ratio	Quick ratio	Numeric
Interest.Expense.Ratio	Interest expense ratio	Numeric
Total.debt.Total.net.worth	Ratio of total debt to total net worth	Numeric
Debt.ratio.	Debt ratio	Numeric
Net.worth.Assets	Net worth to total assets ratio	Numeric
Long.term.fund.suitability.ratio.A	Long-term fund suitability ratio (class A)	Numeric
Borrowing.dependency	Borrowing dependency	Numeric
Contingent.liabilities.Net.worth	Ratio of contingent liabilities to net worth	Numeric
Operating.profit.Paid.in.capital	Operating profit paid in capital	Numeric
Net.profit.before.tax.Paid.in.capital	Net profit before tax paid in capital	Numeric
Inventory.and.accounts.receivable.Net.value	Net value of inventory and accounts receivable	Numeric
Total.Asset.Turnover	Total asset turnover	Numeric
Accounts.Receivable.Turnover	Accounts receivable turnover	Numeric
Average.Collection.Days	Average collection days	Numeric
Inventory.Turnover.Rate..times.	Inventory turnover rate in times	Numeric
Fixed.Assets.Turnover.Frequency	Fixed assets turnover frequency	Numeric
Net.Worth.Turnover.Rate..times.	Net worth turnover rate in times	Numeric
Revenue.per.person	Revenue per person	Numeric
Operating.profit.per.person	Operating profit per person	Numeric
Allocation.rate.per.person	Allocation rate per person	Numeric

Working.Capital.to.Total.Assets	Working capital to total assets ratio	Numeric
Quick.Assets.Total.Assets	Ratio of quick assets to total assets	Numeric
Current.Assets.Total.Assets	Ratio of current assets to total assets	Numeric
Cash.Total.Assets	Ratio of cash to total assets	Numeric
Quick.Assets.Current.Liability	Ratio of quick assets to current liability	Numeric
Cash.Current.Liability	Ratio of cash to current liability	Numeric
Current.Liability.to.Assets	Ratio of current liability to total assets	Numeric
Operating.Funds.to.Liability	Ratio of operating funds to liability	Numeric
Inventory.Working.Capital	Ratio of inventory to working capital	Numeric
Inventory.Current.Liability	Ratio of inventory to current liability	Numeric
Current.Liabilities.Liability	Ratio of current liabilities to liability	Numeric
Working.Capital.Equity	Working capital to equity ratio	Numeric
Current.Liabilities.Equity	Ratio of current liabilities to equity	Numeric
Long.term.Liability.to.Current.Assets	Ratio of long-term liability to current assets	Numeric
Retained.Earnings.to.Total.Assets	Ratio of retained earnings to total assets	Numeric
Total.income.Total.expense	Ratio of total income to total expense	Numeric
Total.expense.Assets	Ratio of total expense to total assets	Numeric
Current.Asset.Turnover.Rate	Current asset turnover rate	Numeric
Quick.Asset.Turnover.Rate	Quick asset turnover rate	Numeric
Working.capitcal.Turnover.Rate	Working capital turnover rate	Numeric
Cash.Turnover.Rate	Cash turnover rate	Numeric
Cash.Flow.to.Sales	Cash flow to sales ratio	Numeric
Fixed.Assets.to.Assets	Ratio of fixed assets to total assets	Numeric
Current.Liability.to.Liability	Ratio of current liability to total liability	Numeric
Current.Liability.to.Equity	Ratio of current liability to equity	Numeric
Equity.to.Long.term.Liability	Ratio of equity to long-term liability	Numeric
Cash.Flow.to.Total.Assets	Ratio of cash flow to total assets	Numeric
Cash.Flow.to.Liability	Ratio of cash flow to liability	Numeric
CFO.to.Assets	Cash flow from operations to total assets	Numeric
Cash.Flow.to.Equity	Ratio of cash flow to equity	Numeric
Current.Liability.to.Current.Assets	Ratio of current liability to current assets	Numeric
Liability.Assets.Flag	Binary indicator (1 if total liability exceeds total assets, 0 otherwise)	Categorical
Net.Income.to.Total.Assets	Ratio of net income to total assets	Numeric
Total.assets.to.GNP.price	Ratio of total assets to GNP price	Numeric
No.credit.Interval	No-credit interval	Numeric
Gross.Profit.to.Sales	Ratio of gross profit to sales	Numeric

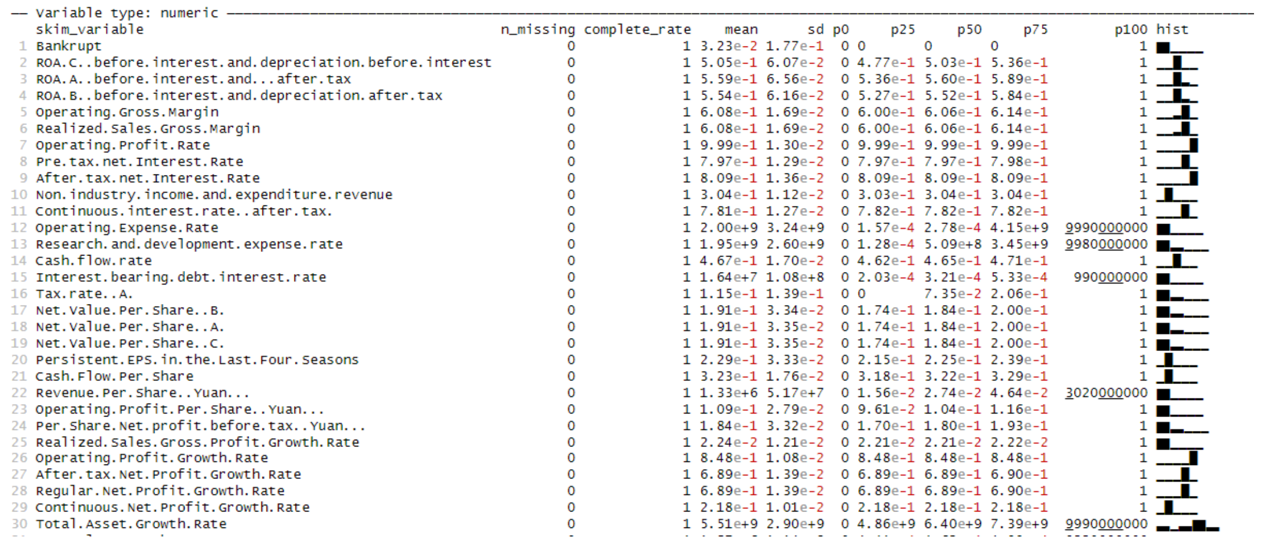
Net.Income.to.Stockholder.s.Equity	Ratio of net income to stockholder's equity	Numeric
Liability.to.Equity	Ratio of liability to equity	Numeric
Degree.of.Financial.Leverage..DFL	Degree of financial leverage	Numeric
Interest.Coverage.Ratio..Interest.expense.to.EBIT.	Interest coverage ratio (Interest expense to EBIT)	Numeric
Net.Income.Flag	Binary indicator (1 if net income is negative for the last two years, 0 otherwise)	Categorical
Equity.to.Liability	Ratio of equity to liability	Numeric

Table 1: Predictor Description

Drawing from the diverse set of 96 features, including financial ratio metrics like Return on Assets, Gross Profits, and various operational and financial indicators, the objective is to scrutinize the intricate relationship between these predictor variables and the likelihood of bankruptcy in companies. The analysis will involve a meticulous preprocessing phase, encompassing transformations and the potential elimination of certain predictors, to ensure the data is conducive for robust model development. Subsequently, the investigation will apply a comprehensive approach, incorporating both linear and nonlinear classification models, along with continuous models. These models aim to predict the binary outcome of bankruptcy status, as well as the real-valued variable indicating the severity of financial distress. The overarching goal is to equip companies with an effective predictive tool that aids in making timely and informed decisions to navigate potential financial crises

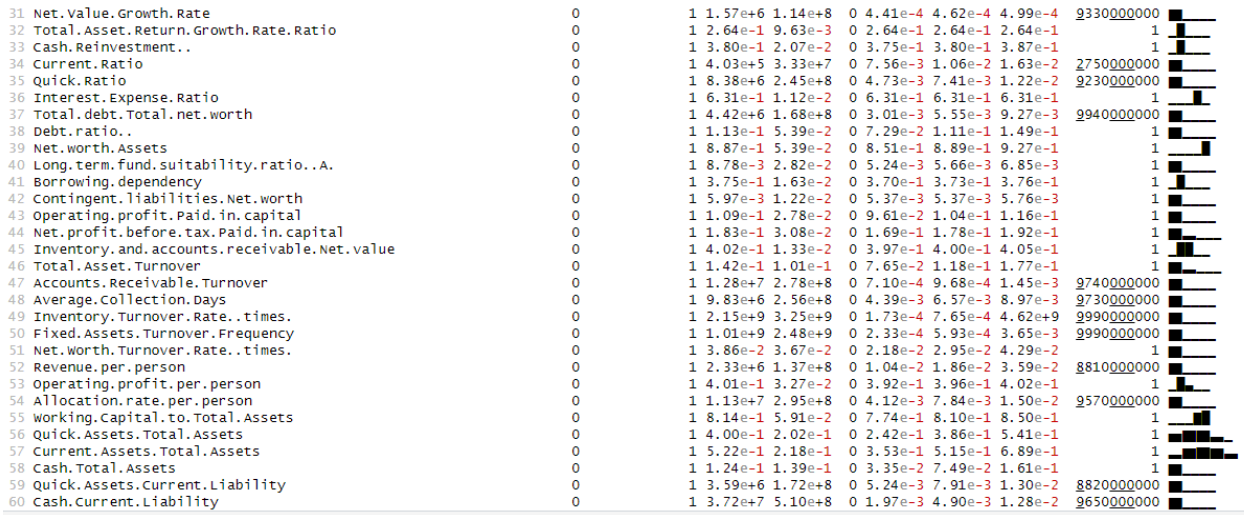
Preprocessing of the Predictors

The data had most of the predictor variables on a scale of 0 to 1, except for a couple of variables that had different scales. We checked for missing values, and there are none in the dataset.

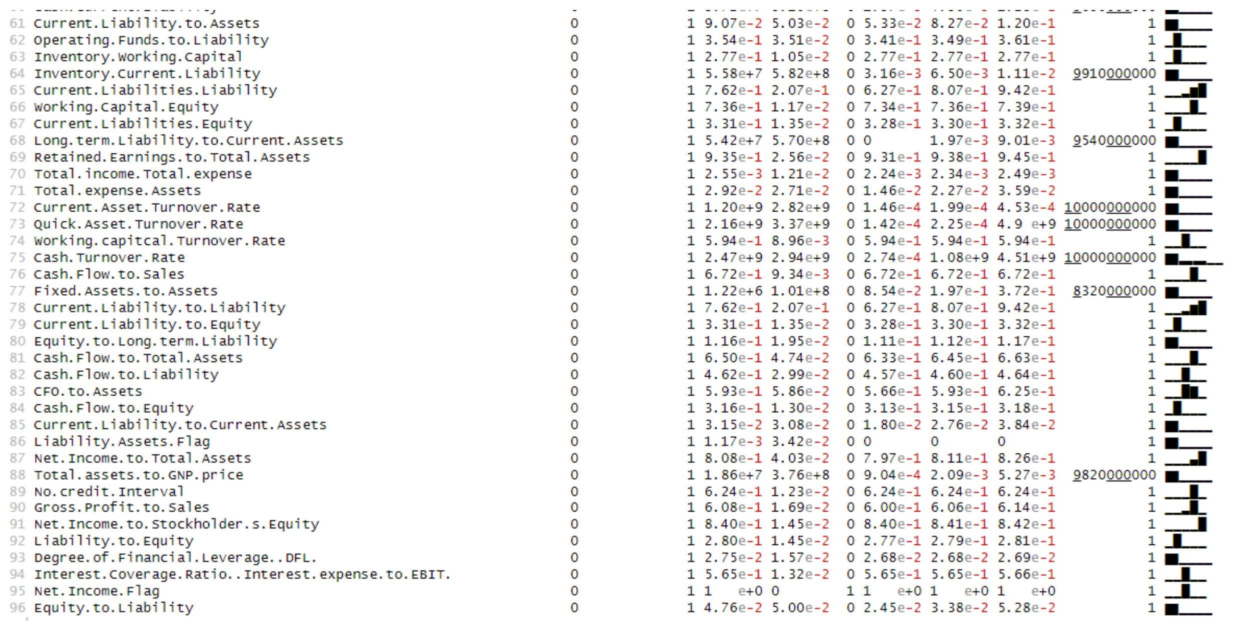


```
── Variable type: numeric ──────────────────────────────────────────────────────────────────────────────────────
   skim_variable                                         n_missing complete_rate     mean       sd p0     p25      p50      p75       p100 hist
 1 Bankrupt                                                      0             1 3.23e-2 1.77e-1  0 0        0        0                1
 2 ROA.C..before.interest.and.depreciation.before.interest       0             1 5.05e-1 6.07e-2  0 4.77e-1 5.03e-1 5.36e-1          1
 3 ROA.A..before.interest.and...after.tax                        0             1 5.59e-1 6.56e-2  0 5.36e-1 5.60e-1 5.89e-1          1
 4 ROA.B..before.interest.and.depreciation.after.tax             0             1 5.54e-1 6.16e-2  0 5.27e-1 5.52e-1 5.84e-1          1
 5 Operating.Gross.Margin                                        0             1 6.08e-1 1.69e-2  0 6.00e-1 6.06e-1 6.14e-1          1
 6 Realized.Sales.Gross.Margin                                   0             1 6.08e-1 1.69e-2  0 6.00e-1 6.06e-1 6.14e-1          1
 7 Operating.Profit.Rate                                         0             1 9.99e-1 1.30e-2  0 9.99e-1 9.99e-1 9.99e-1          1
 8 Pre.tax.net.Interest.Rate                                     0             1 7.97e-1 1.29e-2  0 7.97e-1 7.97e-1 7.98e-1          1
 9 After.tax.net.Interest.Rate                                   0             1 8.09e-1 1.36e-2  0 8.09e-1 8.09e-1 8.09e-1          1
10 Non.industry.income.and.expenditure.revenue                   0             1 3.04e-1 1.12e-2  0 3.03e-1 3.04e-1 3.04e-1          1
11 Continuous.interest.rate..after.tax.                          0             1 7.81e-1 1.27e-2  0 7.82e-1 7.82e-1 7.82e-1          1
12 Operating.Expense.Rate                                        0             1 2.00e+9 3.24e+9  0 1.57e-4 2.78e-4 4.15e+9 9990000000
13 Research.and.development.expense.rate                         0             1 1.95e+9 2.60e+9  0 1.28e-4 5.09e+8 3.45e+9 9980000000
14 Cash.flow.rate                                                0             1 4.67e-1 1.70e-2  0 4.62e-1 4.65e-1 4.71e-1          1
15 Interest.bearing.debt.interest.rate                           0             1 1.64e+7 1.08e+8  0 2.03e-4 3.21e-4 5.33e-4  990000000
16 Tax.rate..A.                                                  0             1 1.15e-1 1.39e-1  0 0       7.35e-2 2.06e-1          1
17 Net.Value.Per.Share..B.                                       0             1 1.91e-1 3.34e-2  0 1.74e-1 1.84e-1 2.00e-1          1
18 Net.Value.Per.Share..A.                                       0             1 1.91e-1 3.35e-2  0 1.74e-1 1.84e-1 2.00e-1          1
19 Net.Value.Per.Share..C.                                       0             1 1.91e-1 3.35e-2  0 1.74e-1 1.84e-1 2.00e-1          1
20 Persistent.EPS.in.the.Last.Four.Seasons                       0             1 2.29e-1 3.33e-2  0 2.15e-1 2.25e-1 2.39e-1          1
21 Cash.Flow.Per.Share                                           0             1 3.23e-1 1.76e-2  0 3.18e-1 3.22e-1 3.29e-1          1
22 Revenue.Per.Share..Yuan...                                    0             1 1.33e+6 5.17e+7  0 1.56e-2 2.74e-2 4.64e-2 3020000000
23 Operating.Profit.Per.Share..Yuan...                           0             1 1.09e-1 2.79e-2  0 9.61e-2 1.04e-1 1.16e-1          1
24 Per.Share.Net.profit.before.tax..Yuan...                      0             1 1.84e-1 3.32e-2  0 1.70e-1 1.80e-1 1.93e-1          1
25 Realized.Sales.Gross.Profit.Growth.Rate                       0             1 2.24e-2 1.21e-2  0 2.21e-2 2.21e-2 2.22e-2          1
26 Operating.Profit.Growth.Rate                                  0             1 8.48e-1 1.08e-2  0 8.48e-1 8.48e-1 8.48e-1          1
27 After.tax.Net.Profit.Growth.Rate                              0             1 6.89e-1 1.39e-2  0 6.89e-1 6.89e-1 6.90e-1          1
28 Regular.Net.Profit.Growth.Rate                                0             1 6.89e-1 1.39e-2  0 6.89e-1 6.89e-1 6.90e-1          1
29 Continuous.Net.Profit.Growth.Rate                             0             1 2.18e-1 1.01e-2  0 2.18e-1 2.18e-1 2.18e-1          1
30 Total.Asset.Growth.Rate                                       0             1 5.51e+9 2.90e+9  0 4.86e+9 6.40e+9 7.39e+9 9990000000
```

Fig 1: Descriptive statistics



```
31 Net.Value.Growth.Rate                         0    1 1.57e+6 1.14e+8  0 4.41e-4 4.62e-4 4.99e-4  9330000000
32 Total.Asset.Return.Growth.Rate.Ratio          0    1 2.64e-1 9.63e-3  0 2.64e-1 2.64e-1 2.64e-1           1
33 Cash.Reinvestment..                           0    1 3.80e-1 2.07e-2  0 3.75e-1 3.80e-1 3.87e-1           1
34 Current.Ratio                                 0    1 4.03e+5 3.33e+7  0 7.56e-3 1.06e-2 1.63e-2  2750000000
35 Quick.Ratio                                   0    1 8.38e+6 2.45e+8  0 4.73e-3 7.41e-3 1.22e-2  9230000000
36 Interest.Expense.Ratio                        0    1 6.31e-1 1.12e-2  0 6.31e-1 6.31e-1 6.31e-1           1
37 Total.debt.Total.net.worth                    0    1 4.42e+6 1.68e+8  0 3.01e-3 5.55e-3 9.27e-3  9940000000
38 Debt.ratio..                                  0    1 1.13e-1 5.39e-2  0 7.29e-2 1.11e-1 1.49e-1           1
39 Net.worth.Assets                              0    1 8.87e-1 5.39e-2  0 8.51e-1 8.89e-1 9.27e-1           1
40 Long.term.fund.suitability.ratio..A.          0    1 8.78e-3 2.82e-2  0 5.24e-3 5.66e-3 6.85e-3           1
41 Borrowing.dependency                          0    1 3.75e-1 1.63e-2  0 3.70e-1 3.73e-1 3.76e-1           1
42 Contingent.liabilities.Net.worth              0    1 5.97e-3 1.22e-2  0 5.37e-3 5.37e-3 5.76e-3           1
43 Operating.profit.Paid.in.capital              0    1 1.09e-1 2.78e-2  0 9.61e-2 1.04e-1 1.16e-1           1
44 Net.profit.before.tax.Paid.in.capital         0    1 1.83e-1 3.08e-2  0 1.69e-1 1.78e-1 1.92e-1           1
45 Inventory.and.accounts.receivable.Net.value   0    1 4.02e-1 1.33e-2  0 3.97e-1 4.00e-1 4.05e-1           1
46 Total.Asset.Turnover                          0    1 1.42e-1 1.01e-1  0 7.65e-2 1.18e-1 1.77e-1           1
47 Accounts.Receivable.Turnover                  0    1 1.28e+7 2.78e+8  0 7.10e-4 9.68e-4 1.45e-3  9740000000
48 Average.Collection.Days                       0    1 9.83e+6 2.56e+8  0 4.39e-3 6.57e-3 8.97e-3  9730000000
49 Inventory.Turnover.Rate..times.               0    1 2.15e+9 3.25e+9  0 1.73e-4 7.65e-4 4.62e+9  9990000000
50 Fixed.Assets.Turnover.Frequency               0    1 1.01e+9 2.48e+9  0 2.33e-4 5.93e-4 3.65e-3  9990000000
51 Net.Worth.Turnover.Rate..times.               0    1 3.86e-2 3.67e-2  0 2.18e-2 2.95e-2 4.29e-2           1
52 Revenue.per.person                            0    1 2.33e+6 1.37e+8  0 1.04e-2 1.86e-2 3.59e-2  8810000000
53 Operating.profit.per.person                   0    1 4.01e-1 3.27e-2  0 3.92e-1 3.96e-1 4.02e-1           1
54 Allocation.rate.per.person                    0    1 1.13e+7 2.95e+8  0 4.12e-3 7.84e-3 1.50e-2  9570000000
55 Working.Capital.to.Total.Assets               0    1 8.14e-1 5.91e-2  0 7.74e-1 8.10e-1 8.50e-1           1
56 Quick.Assets.Total.Assets                     0    1 4.00e-1 2.02e-1  0 2.42e-1 3.86e-1 5.41e-1           1
57 Current.Assets.Total.Assets                   0    1 5.22e-1 2.18e-1  0 3.53e-1 5.15e-1 6.89e-1           1
58 Cash.Total.Assets                             0    1 1.24e-1 1.39e-1  0 3.35e-2 7.49e-2 1.61e-1           1
59 Quick.Assets.Current.Liability                0    1 3.59e+6 1.72e+8  0 5.24e-3 7.91e-3 1.30e-2  8820000000
60 Cash.Current.Liability                        0    1 3.72e+7 5.10e+8  0 1.97e-3 4.90e-3 1.28e-2  9650000000
```



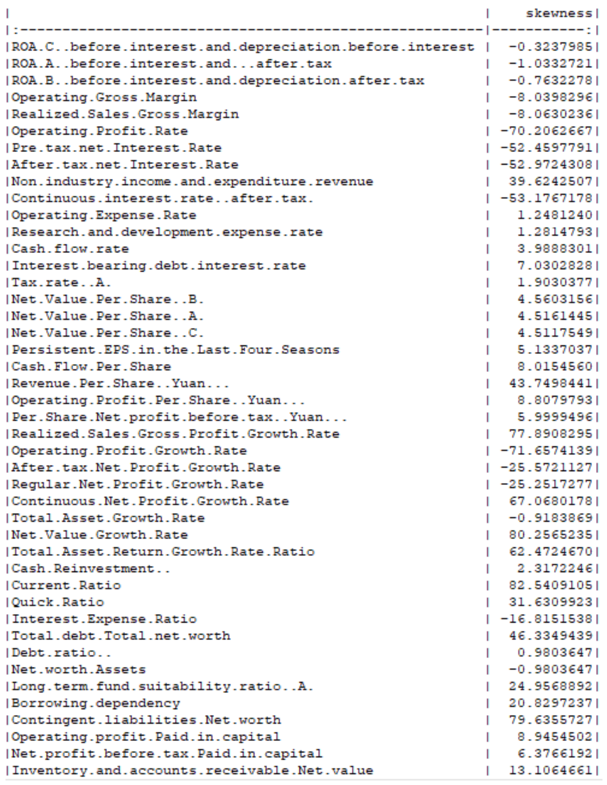
```
61 Current.Liability.to.Assets                   0    1 9.07e-2 5.03e-2  0 5.33e-2 8.27e-2 1.20e-1           1
62 Operating.Funds.to.Liability                  0    1 3.54e-1 3.51e-2  0 3.41e-1 3.49e-1 3.61e-1           1
63 Inventory.Working.Capital                     0    1 2.77e-1 1.05e-2  0 2.77e-1 2.77e-1 2.77e-1           1
64 Inventory.Current.Liability                   0    1 5.58e+7 5.82e+8  0 3.16e-3 6.50e-3 1.11e-2  9910000000
65 Current.Liabilities.Liability                 0    1 7.62e-1 2.07e-1  0 6.27e-1 8.07e-1 9.42e-1           1
66 Working.Capital.Equity                        0    1 7.36e-1 1.17e-2  0 7.34e-1 7.36e-1 7.39e-1           1
67 Current.Liabilities.Equity                    0    1 3.31e-1 1.35e-2  0 3.28e-1 3.30e-1 3.32e-1           1
68 Long.term.Liability.to.Current.Assets         0    1 5.42e+7 5.70e+8  0 0       1.97e-3 9.01e-3  9540000000
69 Retained.Earnings.to.Total.Assets             0    1 9.35e-1 2.56e-2  0 9.31e-1 9.38e-1 9.45e-1           1
70 Total.income.Total.expense                    0    1 2.55e-3 1.21e-2  0 2.24e-3 2.34e-3 2.49e-3           1
71 Total.expense.Assets                          0    1 2.92e-2 2.71e-2  0 1.46e-2 2.27e-2 3.59e-2           1
72 Current.Asset.Turnover.Rate                   0    1 1.20e+9 2.82e+9  0 1.46e-4 1.99e-4 4.53e-4 10000000000
73 Quick.Asset.Turnover.Rate                     0    1 2.16e+9 3.37e+9  0 1.42e-4 2.25e-4 4.9  e+9 10000000000
74 Working.capitcal.Turnover.Rate                0    1 5.94e-1 8.96e-3  0 5.94e-1 5.94e-1 5.94e-1           1
75 Cash.Turnover.Rate                            0    1 2.47e+9 2.94e+9  0 2.74e-4 1.08e+9 4.51e+9 10000000000
76 Cash.Flow.to.Sales                            0    1 6.72e-1 9.34e-3  0 6.72e-1 6.72e-1 6.72e-1           1
77 Fixed.Assets.to.Assets                        0    1 1.22e+6 1.01e+8  0 8.54e-2 1.97e-1 3.72e-1  8320000000
78 Current.Liability.to.Liability                0    1 7.62e-1 2.07e-1  0 6.27e-1 8.07e-1 9.42e-1           1
79 Current.Liability.to.Equity                   0    1 3.31e-1 1.35e-2  0 3.28e-1 3.30e-1 3.32e-1           1
80 Equity.to.Long.term.Liability                 0    1 1.16e-1 1.95e-2  0 1.11e-1 1.12e-1 1.17e-1           1
81 Cash.Flow.to.Total.Assets                     0    1 6.50e-1 4.74e-2  0 6.33e-1 6.45e-1 6.63e-1           1
82 Cash.Flow.to.Liability                        0    1 4.62e-1 2.99e-2  0 4.57e-1 4.60e-1 4.64e-1           1
83 CFO.to.Assets                                 0    1 5.93e-1 5.86e-2  0 5.66e-1 5.93e-1 6.25e-1           1
84 Cash.Flow.to.Equity                           0    1 3.16e-1 1.30e-2  0 3.13e-1 3.15e-1 3.18e-1           1
85 Current.Liability.to.Current.Assets           0    1 3.15e-2 3.08e-2  0 1.80e-2 2.76e-2 3.84e-2           1
86 Liability.Assets.Flag                         0    1 1.17e-3 3.42e-2  0 0       0       0                 1
87 Net.Income.to.Total.Assets                    0    1 8.08e-1 4.03e-2  0 7.97e-1 8.11e-1 8.26e-1           1
88 Total.assets.to.GNP.price                     0    1 1.86e+7 3.76e+8  0 9.04e-4 2.09e-3 5.27e-3  9820000000
89 No.credit.Interval                            0    1 6.24e-1 1.23e-2  0 6.24e-1 6.24e-1 6.24e-1           1
90 Gross.Profit.to.Sales                         0    1 6.08e-1 1.69e-2  0 6.00e-1 6.06e-1 6.14e-1           1
91 Net.Income.to.Stockholder.s.Equity            0    1 8.40e-1 1.45e-2  0 8.40e-1 8.41e-1 8.42e-1           1
92 Liability.to.Equity                           0    1 2.80e-1 1.45e-2  0 2.77e-1 2.79e-1 2.81e-1           1
93 Degree.of.Financial.Leverage..DFL.            0    1 2.75e-2 1.57e-2  0 2.68e-2 2.68e-2 2.69e-2           1
94 Interest.Coverage.Ratio..Interest.expense.to.EBIT. 0 1 5.65e-1 1.32e-2  0 5.65e-1 5.65e-1 5.66e-1           1
95 Net.Income.Flag                               0    1 1    e+0 0        1 1    e+0 1    e+0 1    e+0           1
96 Equity.to.Liability                           0    1 4.76e-2 5.00e-2  0 2.45e-2 3.38e-2 5.28e-2           1
```

Fig 1: Descriptive Statistics contd.

Skewness:

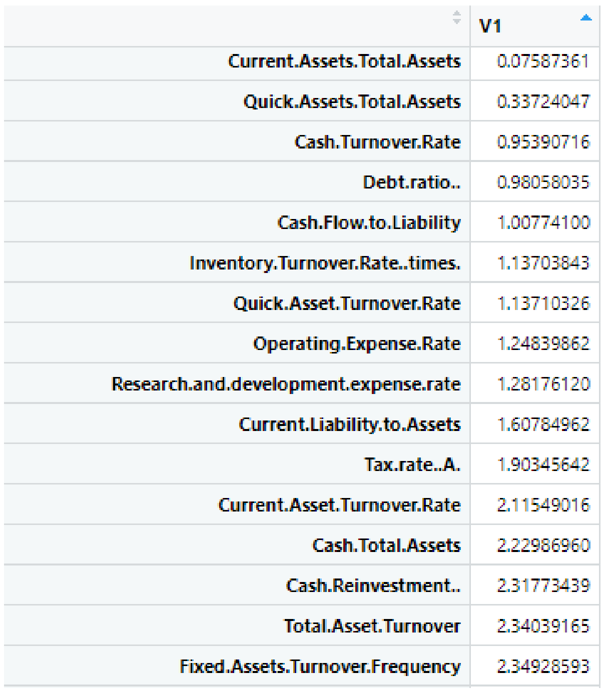
In our analysis, we have identified the presence of both left-skewed and right-skewed continuous predictor variables. This implies that certain factors exhibit a tendency to cluster towards either end of their respective distributions. Additionally, our observations have revealed some distributions closely approximate a normal distribution. The diverse nature of these distributions underscores the importance of considering various types of variables and their inherent patterns in our analytical endeavors. So we wanted to handle the skewness for which we had first performed BOX-COX transformation and then we applied spatial sign transformation to handle outliers and reduce the outliers.

Before Transformation



```
|                                                        |   skewness|
|:-------------------------------------------------------|----------:|
|ROA.C..before.interest.and.depreciation.before.interest |  -0.3237985|
|ROA.A..before.interest.and...after.tax                  |  -1.0332721|
|ROA.B..before.interest.and.depreciation.after.tax       |  -0.7632278|
|Operating.Gross.Margin                                  |  -8.0398296|
|Realized.Sales.Gross.Margin                             |  -8.0630236|
|Operating.Profit.Rate                                   | -70.2062667|
|Pre.tax.net.Interest.Rate                               | -52.4597791|
|After.tax.net.Interest.Rate                             | -52.9724308|
|Non.industry.income.and.expenditure.revenue             |  39.6242507|
|Continuous.interest.rate..after.tax.                    | -53.1767178|
|Operating.Expense.Rate                                  |   1.2481240|
|Research.and.development.expense.rate                   |   1.2814793|
|Cash.flow.rate                                          |   3.9888301|
|Interest.bearing.debt.interest.rate                     |   7.0302828|
|Tax.rate..A.                                            |   1.9030377|
|Net.Value.Per.Share..B.                                 |   4.5603156|
|Net.Value.Per.Share..A.                                 |   4.5161445|
|Net.Value.Per.Share..C.                                 |   4.5117549|
|Persistent.EPS.in.the.Last.Four.Seasons                 |   5.1337037|
|Cash.Flow.Per.Share                                     |   8.0154560|
|Revenue.Per.Share..Yuan...                              |  43.7498441|
|Operating.Profit.Per.Share..Yuan...                     |   8.8079793|
|Per.Share.Net.profit.before.tax..Yuan...                |   5.9999496|
|Realized.Sales.Gross.Profit.Growth.Rate                 |  77.8908295|
|Operating.Profit.Growth.Rate                            | -71.6574139|
|After.tax.Net.Profit.Growth.Rate                        | -25.5721127|
|Regular.Net.Profit.Growth.Rate                          | -25.2517277|
|Continuous.Net.Profit.Growth.Rate                       |  67.0680178|
|Total.Asset.Growth.Rate                                 |  -0.9183869|
|Net.Value.Growth.Rate                                   |  80.2565235|
|Total.Asset.Return.Growth.Rate.Ratio                    |  62.4724670|
|Cash.Reinvestment..                                     |   2.3172246|
|Current.Ratio                                           |  82.5409105|
|Quick.Ratio                                             |  31.6309923|
|Interest.Expense.Ratio                                  | -16.8151538|
|Total.debt.Total.net.worth                              |  46.3349439|
|Debt.ratio..                                            |   0.9803647|
|Net.worth.Assets                                        |  -0.9803647|
|Long.term.fund.suitability.ratio..A.                    |  24.9568892|
|Borrowing.dependency                                    |  20.8297237|
|Contingent.liabilities.Net.worth                        |  79.6355727|
|Operating.profit.Paid.in.capital                        |   8.9454502|
|Net.profit.before.tax.Paid.in.capital                   |   6.3766192|
|Inventory.and.accounts.receivable.Net.value             |  13.1064661|
```

After Transformation



	V1
Current.Assets.Total.Assets	0.07587361
Quick.Assets.Total.Assets	0.33724047
Cash.Turnover.Rate	0.95390716
Debt.ratio..	0.98058035
Cash.Flow.to.Liability	1.00774100
Inventory.Turnover.Rate..times.	1.13703843
Quick.Asset.Turnover.Rate	1.13710326
Operating.Expense.Rate	1.24839862
Research.and.development.expense.rate	1.28176120
Current.Liability.to.Assets	1.60784962
Tax.rate..A.	1.90345642
Current.Asset.Turnover.Rate	2.11549016
Cash.Total.Assets	2.22986960
Cash.Reinvestment..	2.31773439
Total.Asset.Turnover	2.34039165
Fixed.Assets.Turnover.Frequency	2.34928593

Fig 2: Skewness Before and After Transformation

Upon examining the distribution of categorical variables, we conducted a thorough assessment for near-zero variance and subsequently verified that these specific features indeed exhibited zero variance. Recognizing the limited variability in these categorical attributes, we deemed them non-contributory to our analysis and, consequently, opted to exclude them. This strategic decision not only streamlines our dataset but also ensures that our analytical focus is directed toward more informative and discerning variables, thereby enhancing the precision and relevance of our findings.

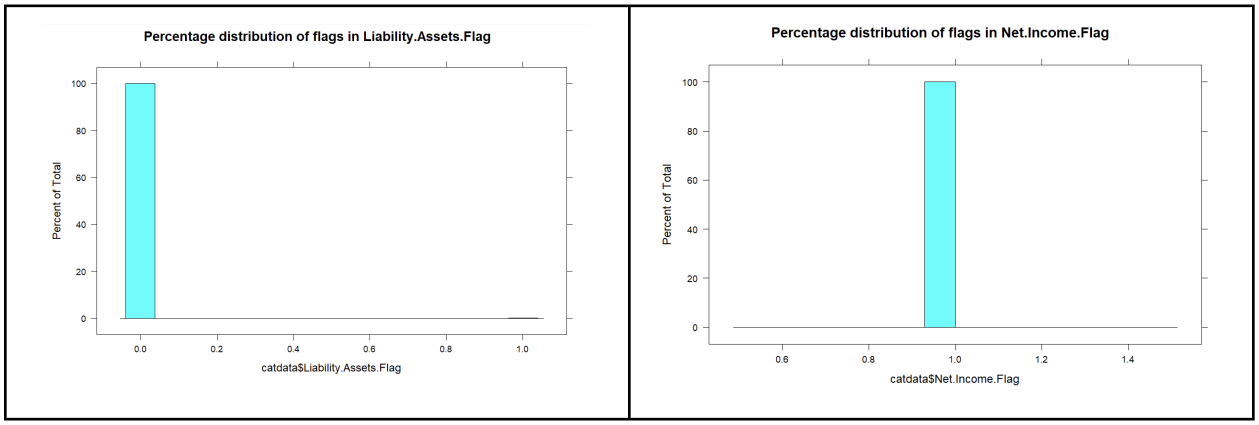


Fig 3: Categorical Variables

We checked for correlation among all the predictors and we found a couple of clusters where we could see that the predictor variables were correlated we wanted to reduce the dimensionality so we used the dimensionality reduction technique Principal Component Analysis(PCA).

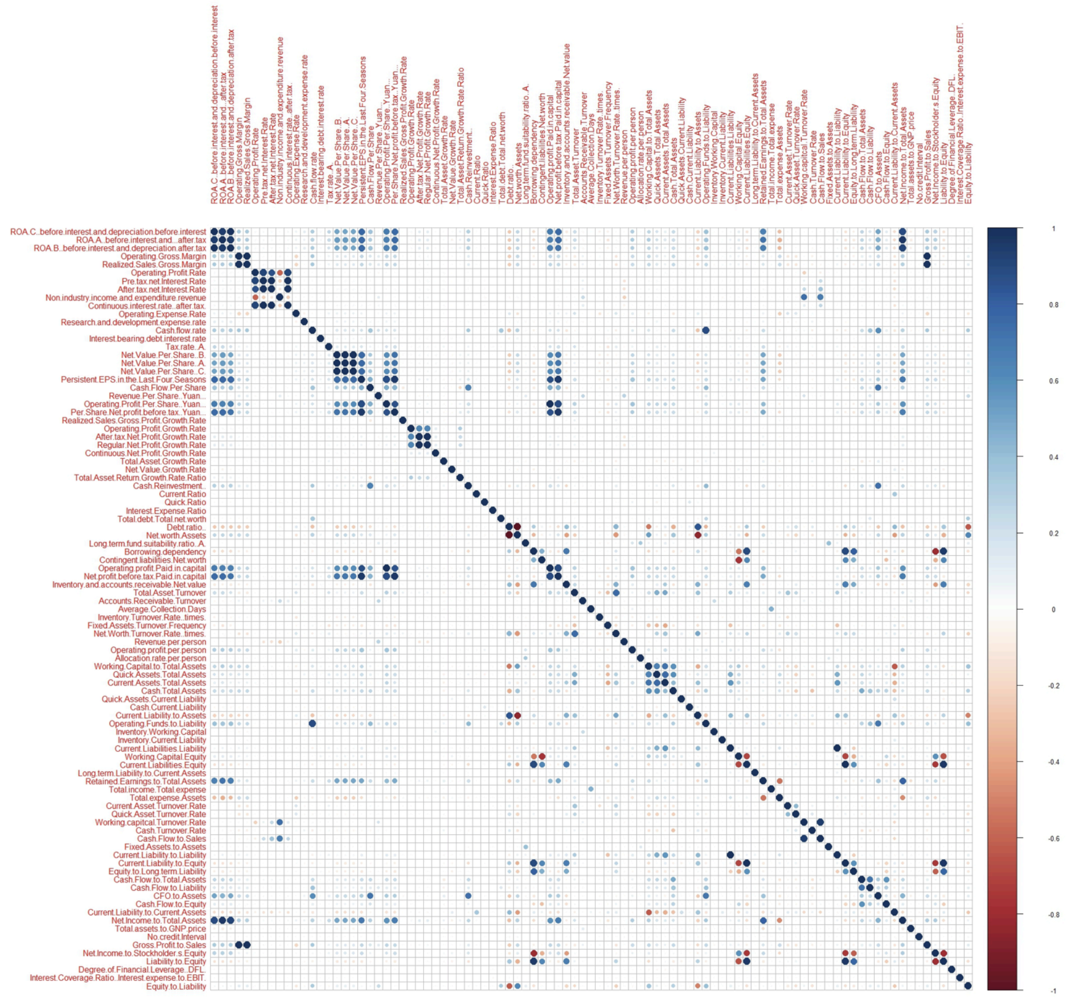
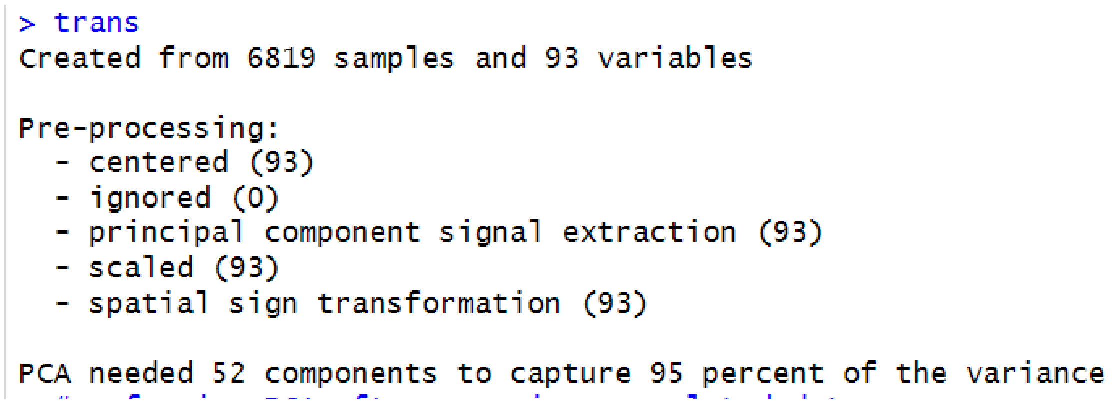


Fig 4: Correlation Plot before PCA.

In our data preprocessing pipeline using the preProcess function in R, we implemented several key transformations to enhance the quality and efficiency of our dataset. Firstly, we applied the Box-Cox transformation to address any non-normality in our data, ensuring a more robust statistical foundation. Subsequently, the Spacialsign transformation was employed to effectively handle outliers and enhance the overall resilience of our dataset. The Center and Scale transformations were then implemented to achieve a mean-centered and standardized dataset, respectively.



```
> trans
Created from 6819 samples and 93 variables

Pre-processing:
  - centered (93)
  - ignored (0)
  - principal component signal extraction (93)
  - scaled (93)
  - spatial sign transformation (93)

PCA needed 52 components to capture 95 percent of the variance
```

Fig 5: PCA transformations.

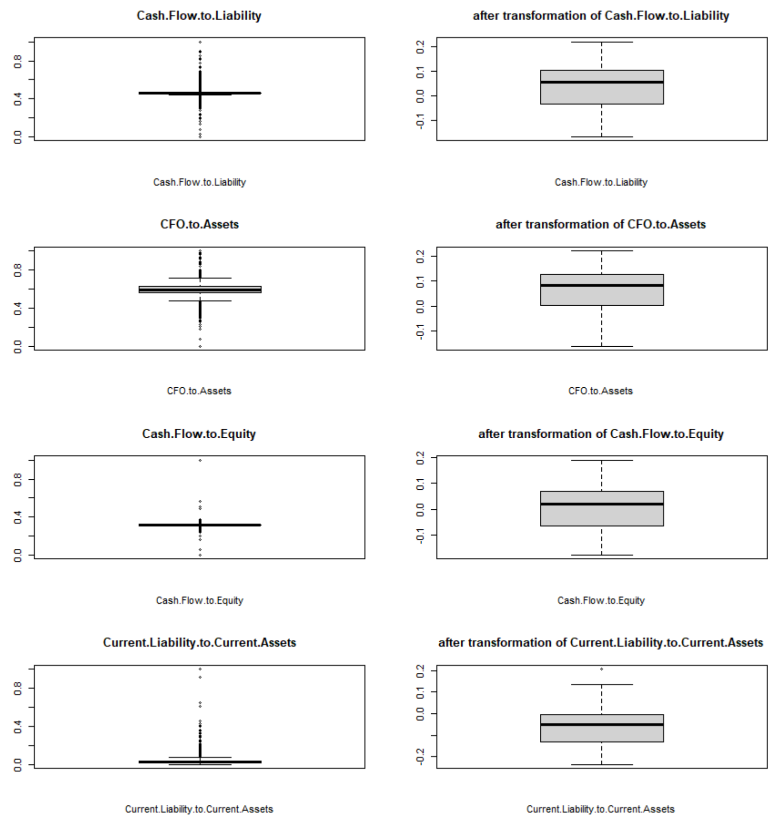


Fig 6: Before and After transformation Boxplots.

Following these preparatory steps, we conducted Principal Components Analysis (PCA) to reduce the dimensionality of our data. The PCA yielded 52 principal components (PCs), which collectively encapsulate the entirety of the dataset's variability. Notably, 90% of the variance was explained by the first 40 PCs, while 95% was captured by the initial 45 PCs. For a more comprehensive representation, we extended our consideration to 51 PCs, covering 99% of the dataset's variability. Ultimately, we retained all 52 PCs, as they collectively account for 100% of the variance in our data. This meticulous approach to dimensionality reduction ensures that our dataset maintains a balance between preserving crucial information and achieving a more manageable and insightful form for subsequent analyses.

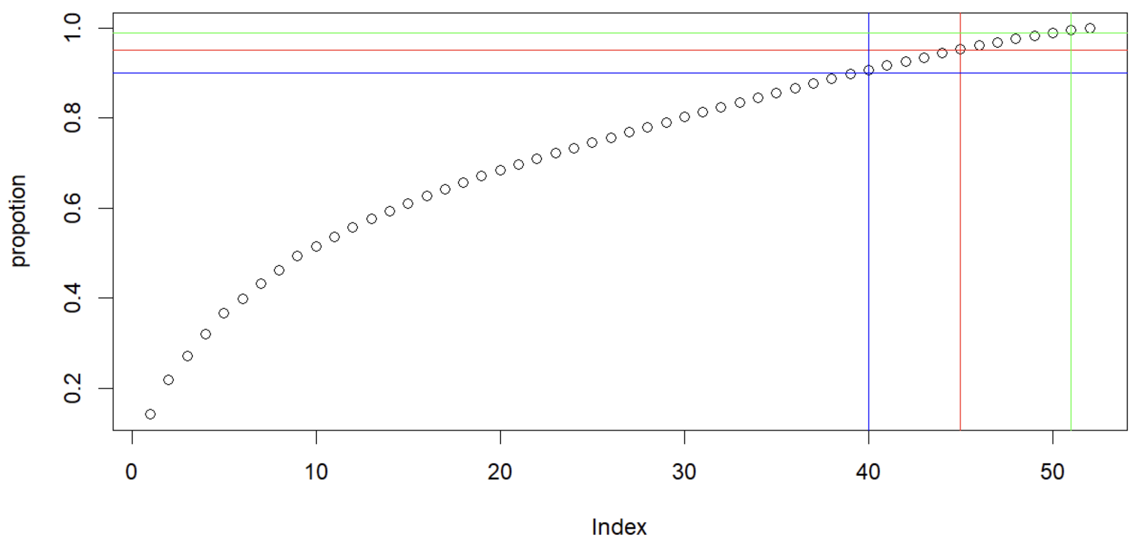
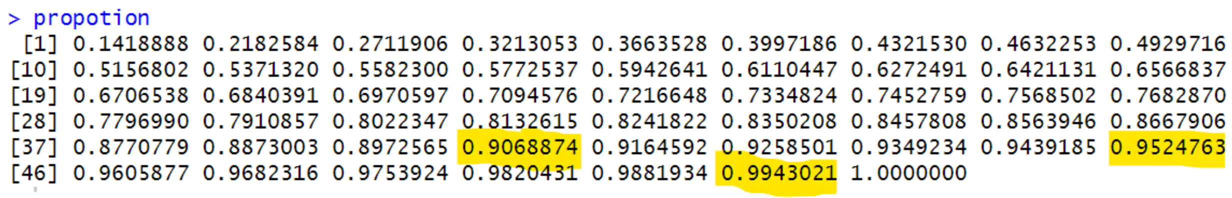


Fig 7: Scree Plot.

We can summarize the above graph using the below data.



```
> propotion
 [1] 0.1418888 0.2182584 0.2711906 0.3213053 0.3663528 0.3997186 0.4321530 0.4632253 0.4929716
[10] 0.5156802 0.5371320 0.5582300 0.5772537 0.5942641 0.6110447 0.6272491 0.6421131 0.6566837
[19] 0.6706538 0.6840391 0.6970597 0.7094576 0.7216648 0.7334824 0.7452759 0.7568502 0.7682870
[28] 0.7796990 0.7910857 0.8022347 0.8132615 0.8241822 0.8350208 0.8457808 0.8563946 0.8667906
[37] 0.8770779 0.8873003 0.8972565 0.9068874 0.9164592 0.9258501 0.9349234 0.9439185 0.9524763
[46] 0.9605877 0.9682316 0.9753924 0.9820431 0.9881934 0.9943021 1.0000000
```

Fig 8: Combined Variance of PC's.

After the removal of categorical features and the application of Principal Components Analysis (PCA), our dataset now comprises 52 continuous variables. To ensure the reliability of our data, we conducted an additional step by assessing the correlation among the principal components obtained from PCA. Fortunately, our thorough examination did not reveal any significant collinearity issues in the dataset. This absence of correlation among the principal components is vital, as it indicates that the reduced set of variables derived from the PCA retains valuable information without introducing multicollinearity concerns.

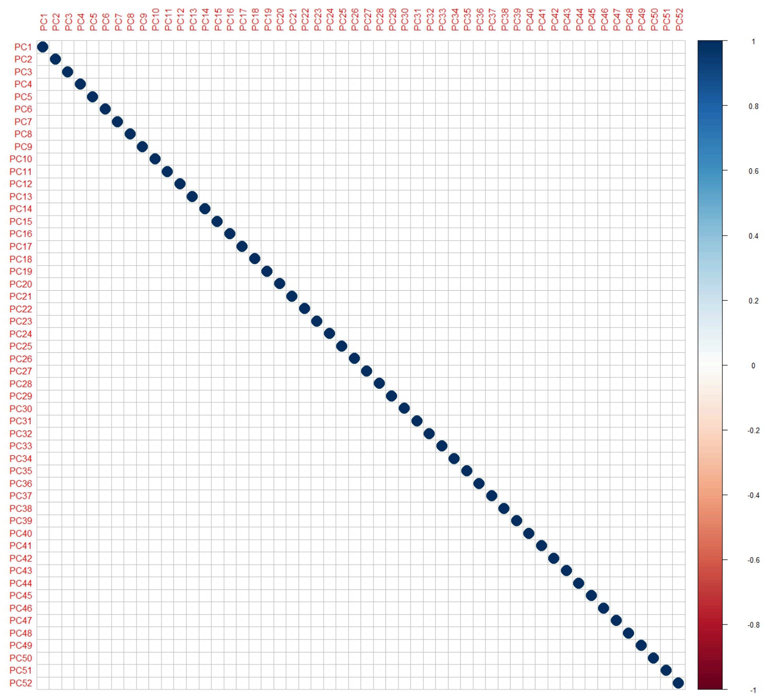


Fig 9: Correlation Plot after PCA.

DATA SPLITTING

In our dataset, which consists of 6819 samples, 52 principal components (PCs), and one response variable, we encounter a notable class imbalance, particularly with only 220 instances of bankrupt companies, representing approximately 3.22% of the total sample. Recognizing the significance of this imbalance, we have implemented a strategic solution by employing stratified sampling during the data-splitting process. This approach ensures that the class distribution is maintained in both the training and testing subsets. Specifically, we have divided the dataset into a 70% training subset and a 30% testing subset, preserving the proportional representation of each class in both partitions. By doing so, we aim to enhance the robustness of our model training and evaluation processes, particularly in addressing the challenges posed by the imbalanced nature of the dataset.

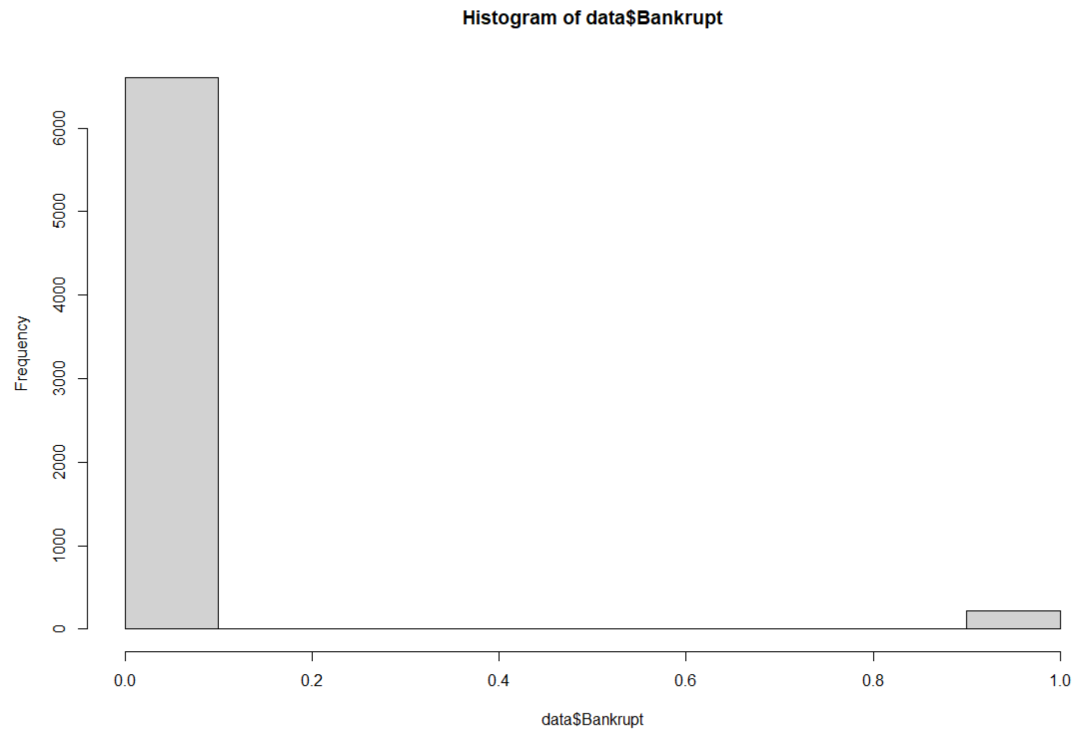


Fig 10: Bar Plot of Response Variable.

With a total of 4774 samples in our training dataset, comprising 70% of the overall 6819 sample size, we intend to enhance our model assessment through a rigorous 10-fold cross-validation strategy. By partitioning the data into 10 subsets and iteratively training and evaluating the model on different combinations of these subsets, we aim to obtain a robust understanding of the model's performance. This approach allows us to leverage the entire training dataset for both learning and validation, providing a comprehensive evaluation that can help ensure the generalizability of our model across diverse data configurations. The use of 10 folds adds an extra layer of reliability to our assessment, contributing to a more thorough and insightful analysis of our predictive model.

Model Training:

In our pursuit of model training, we systematically explored both linear and non-linear approaches to uncover the most suitable predictive algorithm for our dataset. Among the linear models, we considered:

1. Logistic Regression
2. Linear Discriminant Analysis
3. Partial Least Squares (PLS)
4. Penalized models
5. Nearest Shrunken Centroid method

On the non-linear front, our repertoire included:

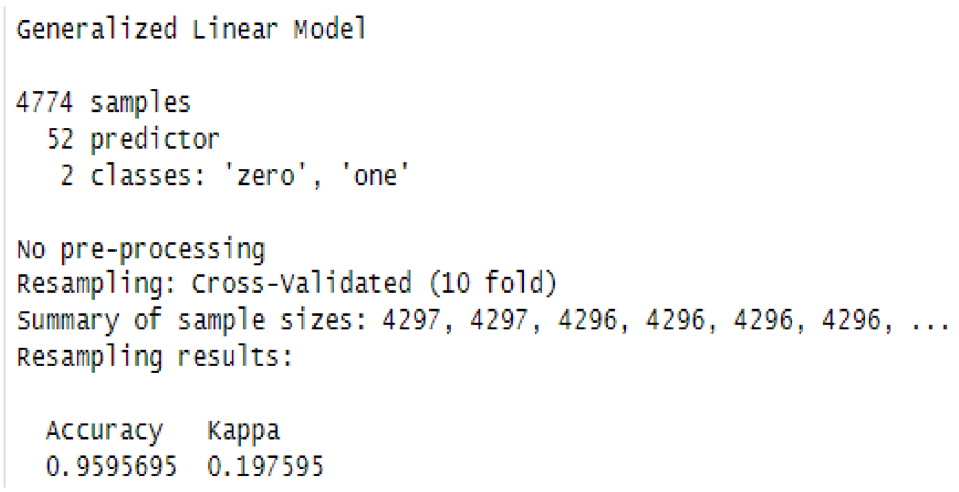
6. Non-Linear Discriminant Analysis
7. Neural Networks
8. Flexible Discriminant Analysis
9. Support Vector Machines (SVM)
10. k-nearest Neighbors (KNN)
11. Naive Bayes

This comprehensive selection aims to capture the diverse underlying patterns in our data, allowing us to evaluate the strengths and limitations of each model type. The outcome of this rigorous exploration will guide us in determining the optimal model that aligns with the intricacies of our dataset and fulfills the specific requirements of our predictive task.

For all the models we have used 10-fold cross-validation and also used the defaultSummary function.

<u>1. LOGISTIC Regression:</u>

The logistic regression model is trained on the training data. The performance metric used for evaluation is the Area Under the Receiver Operating Characteristic curve (ROC). The results of this training process are examined to understand the model's performance and characteristics. This approach allows for systematic and controlled training of a logistic regression model with the specified settings.

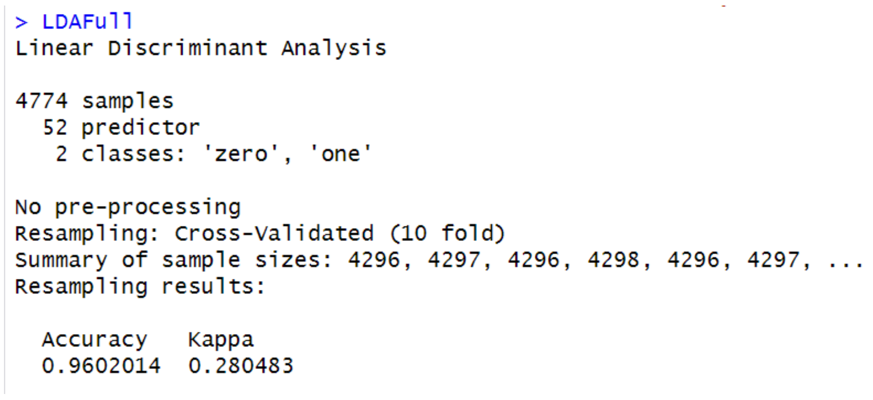


```
Generalized Linear Model

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4297, 4297, 4296, 4296, 4296, 4296, ...
Resampling results:

  Accuracy   Kappa
  0.9595695  0.197595
```

There are no tuning parameters for the Logistic Regression Model.

2. Linear Discriminant Analysis (LDA)

The model is trained on the predictors and the response variable. The evaluation metric for assessing the model's performance is the Area Under the Receiver Operating Characteristic curve (ROC). The output of this summary provides insights into how well the LDA model performs on the given dataset, particularly in terms of its ability to discriminate between different classes based on the ROC metric.

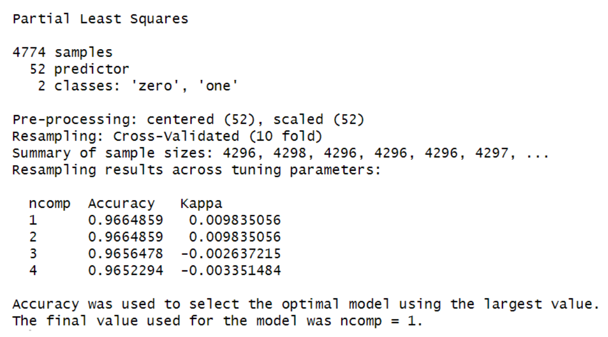


```
> LDAFull
Linear Discriminant Analysis

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4296, 4297, 4296, 4298, 4296, 4297, ...
Resampling results:

  Accuracy   Kappa
  0.9602014  0.280483
```

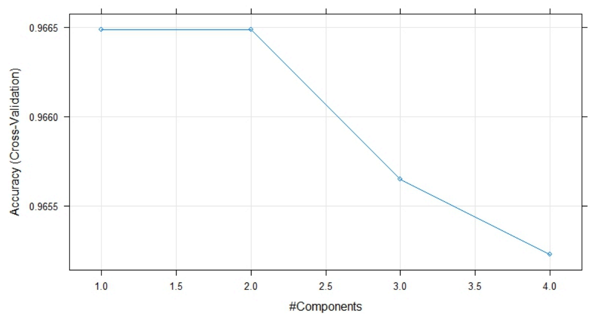
There are no tuning parameters for the Linear Discriminant Analysis Model.

3. Partial Least Squares (PLS):

The tuning of the model involves exploring different numbers of components (ncomp) within the range of 1 to 4. The model's performance is assessed using the Area Under the Receiver Operating Characteristic Curve (ROC) metric. This model aims to find the optimal number of components in the PLS regression that maximizes its ability to discriminate between classes, with ROC serving as a key metric for this evaluation.



```
Partial Least Squares

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

Pre-processing: centered (52), scaled (52)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4296, 4298, 4296, 4296, 4296, 4297, ...
Resampling results across tuning parameters:

  ncomp  Accuracy   Kappa
  1      0.9664859   0.009835056
  2      0.9664859   0.009835056
  3      0.9656478  -0.002637215
  4      0.9652294  -0.003351484

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was ncomp = 1.
```

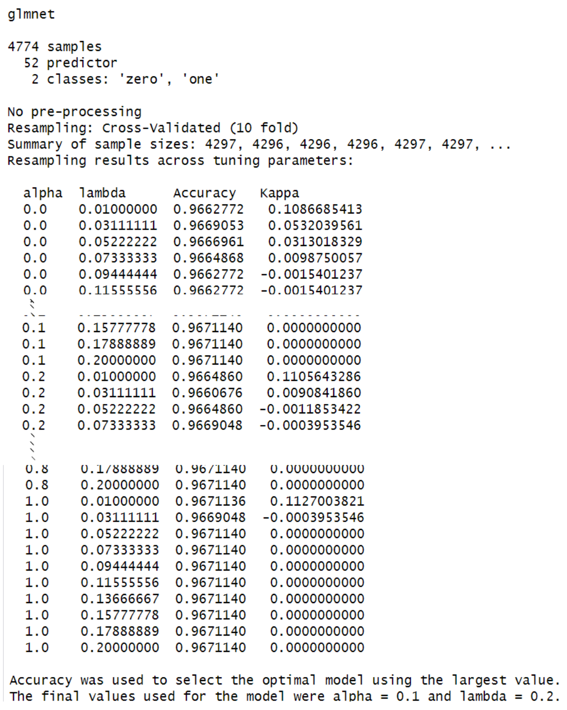


Ncomp = 1 was selected as the tuning parameter for the PLS model.

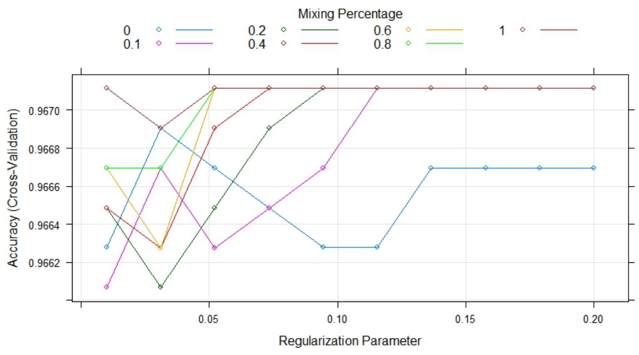
4. Penalized Model - Generalized linear models with elastic net regularization:

The method used is "glmnet," which stands for generalized linear models with elastic net regularization. The tuning of the model involves searching for optimal values of hyperparameters, specifically alpha (the mixing parameter between L1 and L2 regularization) and lambda (the regularization parameter). The grid of values for alpha includes 0, 0.1, 0.2, 0.4, 0.6, 0.8, and 1, while lambda varies from 0.01 to 0.2 with ten equidistant values.

The train function has defaulted to accuracy for tuning the parameters even after giving ROC as a metric.



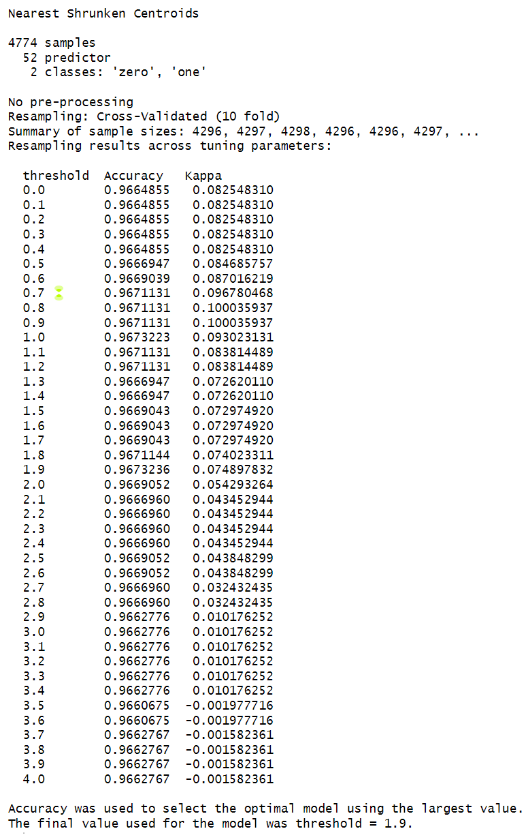
```
glmnet

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4297, 4296, 4296, 4296, 4297, 4297, ...
Resampling results across tuning parameters:

  alpha  lambda      Accuracy   Kappa
  0.0    0.01000000  0.9662772   0.1086685413
  0.0    0.03111111  0.9669053   0.0532039561
  0.0    0.05222222  0.9666961   0.0313018329
  0.0    0.07333333  0.9664868   0.0098750057
  0.0    0.09444444  0.9662772  -0.0015401237
  0.0    0.11555556  0.9662772  -0.0015401237

  0.1    0.15777778  0.9671140   0.0000000000
  0.1    0.17888889  0.9671140   0.0000000000
  0.1    0.20000000  0.9671140   0.0000000000
  0.2    0.01000000  0.9664860   0.1105643286
  0.2    0.03111111  0.9660676   0.0090841860
  0.2    0.05222222  0.9664860  -0.0011853422
  0.2    0.07333333  0.9669048  -0.0003953546

  0.8    0.17888889  0.9671140   0.0000000000
  0.8    0.20000000  0.9671140   0.0000000000
  1.0    0.01000000  0.9671136   0.1127003821
  1.0    0.03111111  0.9669048  -0.0003953546
  1.0    0.05222222  0.9671140   0.0000000000
  1.0    0.07333333  0.9671140   0.0000000000
  1.0    0.09444444  0.9671140   0.0000000000
  1.0    0.11555556  0.9671140   0.0000000000
  1.0    0.13666667  0.9671140   0.0000000000
  1.0    0.15777778  0.9671140   0.0000000000
  1.0    0.17888889  0.9671140   0.0000000000
  1.0    0.20000000  0.9671140   0.0000000000

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were alpha = 0.1 and lambda = 0.2.
```



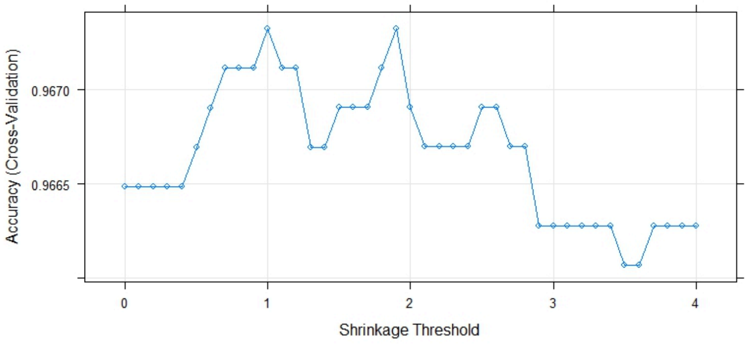
The tuning parameters selected are alpha =0.1 and lambda = 0.2.

5. Nearest Shrunken Centroid:

In this model, the classifier is employed using the train function in R. The method used is "pam," which stands for partitioning around medoids. The tuning of the model involves searching for an optimal threshold value for shrinking centroids. The grid of threshold values ranges from 0 to 4 in increments of 0.1.



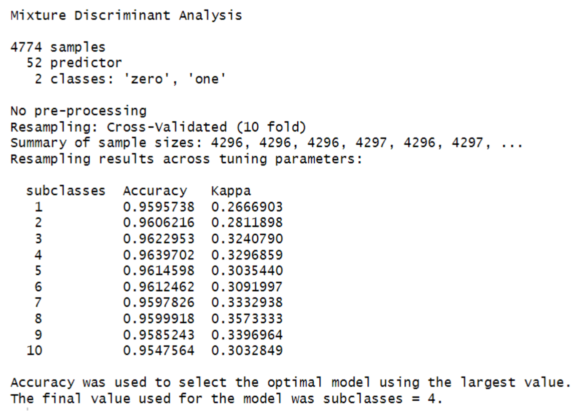
```
Nearest Shrunken Centroids

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4296, 4297, 4298, 4296, 4296, 4297, ...
Resampling results across tuning parameters:

  threshold  Accuracy   Kappa
  0.0        0.9664855   0.082548310
  0.1        0.9664855   0.082548310
  0.2        0.9664855   0.082548310
  0.3        0.9664855   0.082548310
  0.4        0.9664855   0.082548310
  0.5        0.9666947   0.084685757
  0.6        0.9669039   0.087016219
  0.7        0.9671131   0.096780468
  0.8        0.9671131   0.100035937
  0.9        0.9671131   0.100035937
  1.0        0.9673223   0.093023131
  1.1        0.9671131   0.083814489
  1.2        0.9671131   0.083814489
  1.3        0.9666947   0.072620110
  1.4        0.9666947   0.072620110
  1.5        0.9669043   0.072974920
  1.6        0.9669043   0.072974920
  1.7        0.9669043   0.072974920
  1.8        0.9671144   0.074023311
  1.9        0.9673236   0.074897832
  2.0        0.9669052   0.054293264
  2.1        0.9666960   0.043452944
  2.2        0.9666960   0.043452944
  2.3        0.9666960   0.043452944
  2.4        0.9666960   0.043452944
  2.5        0.9669052   0.043848299
  2.6        0.9669052   0.043848299
  2.7        0.9666960   0.032432435
  2.8        0.9666960   0.032432435
  2.9        0.9662776   0.010176252
  3.0        0.9662776   0.010176252
  3.1        0.9662776   0.010176252
  3.2        0.9662776   0.010176252
  3.3        0.9662776   0.010176252
  3.4        0.9662776   0.010176252
  3.5        0.9660675  -0.001977716
  3.6        0.9660675  -0.001977716
  3.7        0.9662767  -0.001582361
  3.8        0.9662767  -0.001582361
  3.9        0.9662767  -0.001582361
  4.0        0.9662767  -0.001582361

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was threshold = 1.9.
```



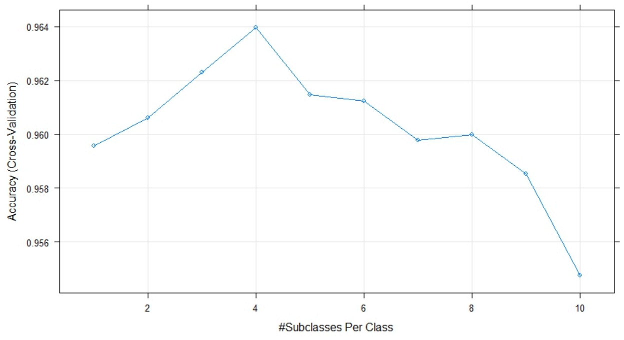
For Nearest Shrunken Centroid the threshold = 1.9 was selected as the tuning threshold.

6. Non-Linear Discriminant Analysis: Multiclass Discriminant Analysis:

In this model, a Multiclass Discriminant Analysis (MDA) classifier the method used is "mda," which stands for Multiclass Discriminant Analysis. The tuning of the model involves searching for an optimal number of subclasses, ranging from 1 to 10.



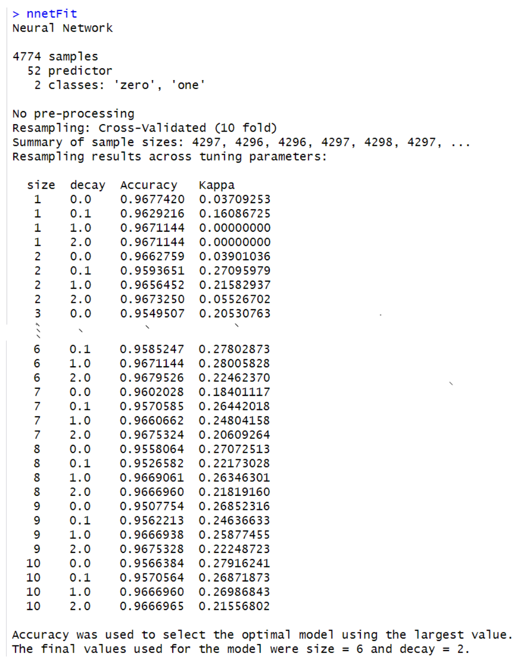
```
Mixture Discriminant Analysis

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4296, 4296, 4296, 4297, 4296, 4297, ...
Resampling results across tuning parameters:

  subclasses  Accuracy   Kappa
   1          0.9595738  0.2666903
   2          0.9606216  0.2811898
   3          0.9622953  0.3240790
   4          0.9639702  0.3296859
   5          0.9614598  0.3035440
   6          0.9612462  0.3091997
   7          0.9597826  0.3332938
   8          0.9599918  0.3573333
   9          0.9585243  0.3396964
  10          0.9547564  0.3032849

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was subclasses = 4.
```



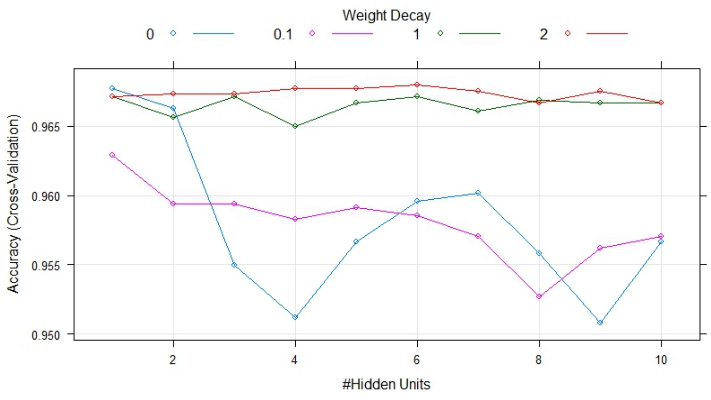
Subclass = 4 was selected as the tuning parameter for Non-Linear Discriminant Analysis.

7. Neural Network classifier:

The neural network classifier is constructed using the method used "nnet," indicating a neural network model. The tuning process involves exploring different combinations of network architectures specified by the grid nnetGrid. Specifically, the grid includes different values for the network size (size) and decay parameters.



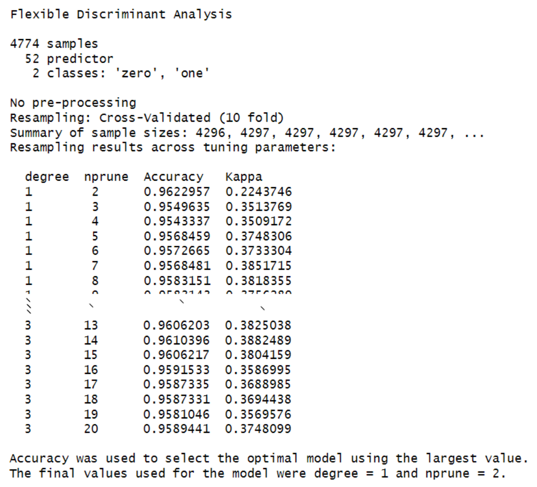
```
> nnetFit
Neural Network

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4297, 4296, 4296, 4297, 4298, 4297, ...
Resampling results across tuning parameters:

  size  decay  Accuracy   Kappa
   1    0.0    0.9677420  0.03709253
   1    0.1    0.9629216  0.16086725
   1    1.0    0.9671144  0.00000000
   1    2.0    0.9671144  0.00000000
   2    0.0    0.9662759  0.03901036
   2    0.1    0.9593651  0.27095979
   2    1.0    0.9656452  0.21582937
   2    2.0    0.9673250  0.05526702
   3    0.0    0.9549507  0.20530763

   6    0.1    0.9585247  0.27802873
   6    1.0    0.9671144  0.28005828
   6    2.0    0.9679526  0.22462370
   7    0.0    0.9602028  0.18401117
   7    0.1    0.9570585  0.26442018
   7    1.0    0.9660662  0.24804158
   7    2.0    0.9675324  0.20609264
   8    0.0    0.9558064  0.27072513
   8    0.1    0.9526582  0.22173028
   8    1.0    0.9669061  0.26346301
   8    2.0    0.9666960  0.21819160
   9    0.0    0.9507754  0.26852316
   9    0.1    0.9562213  0.24636633
   9    1.0    0.9666938  0.25877455
   9    2.0    0.9675328  0.22248723
  10    0.0    0.9566384  0.27916241
  10    0.1    0.9570564  0.26871873
  10    1.0    0.9666960  0.26986843
  10    2.0    0.9666965  0.21556802

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were size = 6 and decay = 2.
```



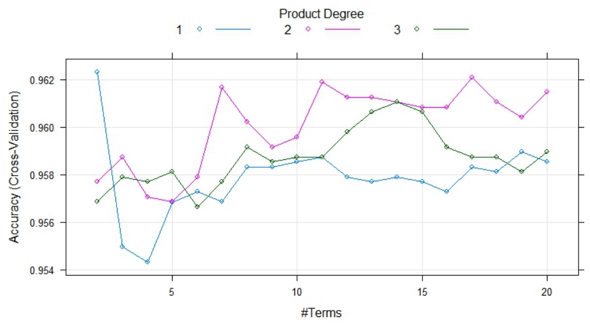
Size = 6 and decay =2 were selected as the tuning parameters for the Neural Network model.

8. Flexible Discriminant Analysis:

Flexible Discriminant Analysis (FDA) classifier is trained using the method used "fda," indicating the application of Flexible Discriminant Analysis. The tuning process involves exploring different combinations of model parameters specified by the grid. Specifically, the grid includes different values for the degree of freedom (degree) and the number of pruning steps (nprune).



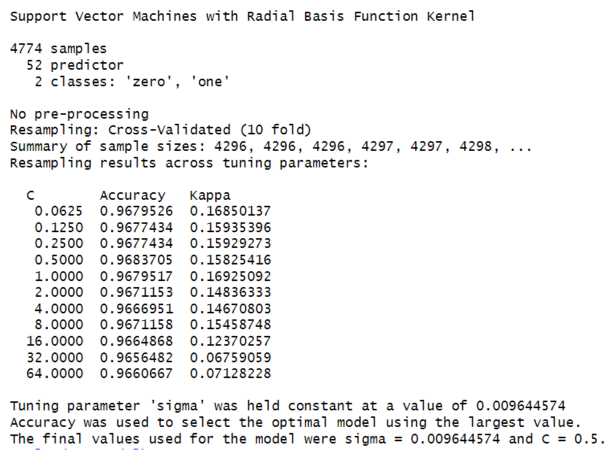
```
Flexible Discriminant Analysis

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4296, 4297, 4297, 4297, 4297, 4297, ...
Resampling results across tuning parameters:

  degree  nprune  Accuracy   Kappa
  1        2      0.9622957  0.2243746
  1        3      0.9549635  0.3513769
  1        4      0.9543337  0.3509172
  1        5      0.9568459  0.3748306
  1        6      0.9572665  0.3733304
  1        7      0.9568481  0.3851715
  1        8      0.9583151  0.3818355
  3       13      0.9606203  0.3825038
  3       14      0.9610396  0.3882489
  3       15      0.9606217  0.3804159
  3       16      0.9591533  0.3586995
  3       17      0.9587335  0.3688985
  3       18      0.9587331  0.3694438
  3       19      0.9581046  0.3569576
  3       20      0.9589441  0.3748099

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were degree = 1 and nprune = 2.
```



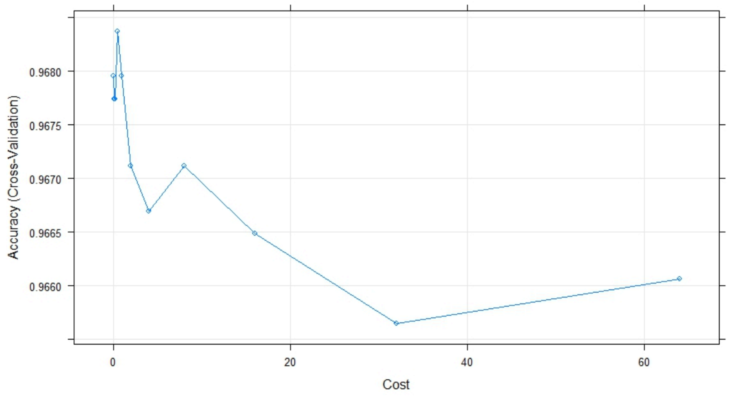
Degree =1 and nprune =2 were selected as the final tuning parameters for Flexible Discriminant Analysis.

9. Support Vector Machine:

The Support Vector Machine (SVM) with a radial kernel is trained using the "svmRadial" method indicating the adoption of a radial basis function kernel for SVM. The tuning process involves exploring different combinations of model parameters specified by the grid svmRGridReduced. This grid includes values for the radial kernel parameter (sigma) obtained from the estimated range of the training data's covariance matrix (sigmaRangeReduced) and the cost parameter (C) with a range of values.



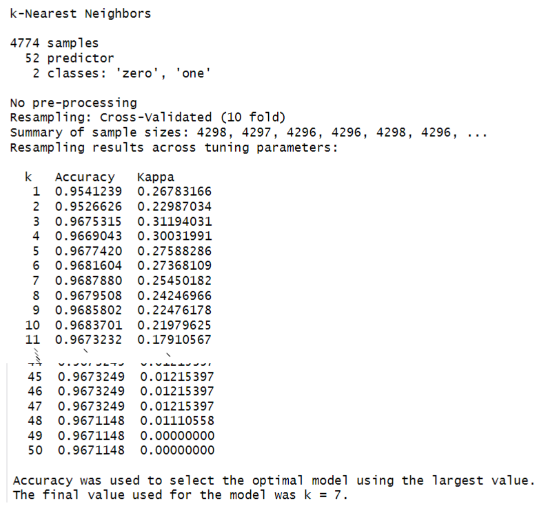
```
Support Vector Machines with Radial Basis Function Kernel

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4296, 4296, 4296, 4297, 4297, 4298, ...
Resampling results across tuning parameters:

  C        Accuracy   Kappa
   0.0625  0.9679526  0.16850137
   0.1250  0.9677434  0.15935396
   0.2500  0.9677434  0.15929273
   0.5000  0.9683705  0.15825416
   1.0000  0.9679517  0.16925092
   2.0000  0.9671153  0.14836333
   4.0000  0.9666951  0.14670803
   8.0000  0.9671158  0.15458748
  16.0000  0.9664868  0.12370257
  32.0000  0.9656482  0.06759059
  64.0000  0.9660667  0.07128228

Tuning parameter 'sigma' was held constant at a value of 0.009644574
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were sigma = 0.009644574 and C = 0.5.
```



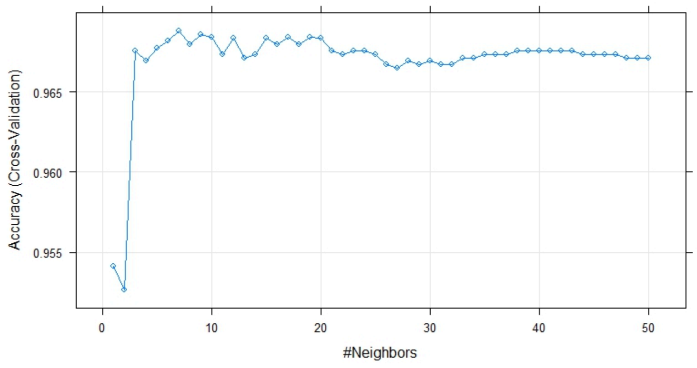
Sigma = 0.009644574 and C = 0.5 were selected as the final parameters for the SVM model.

10. KNN:

The k-Nearest Neighbors (kNN) model is trained using the "knn", method indicating the k-Nearest Neighbors algorithm. The training involves exploring different values of the number of neighbors (k) specified by the grid tuneGrid. In this case, the tuning process is conducted over a range of values from 1 to 50 for the parameter k.



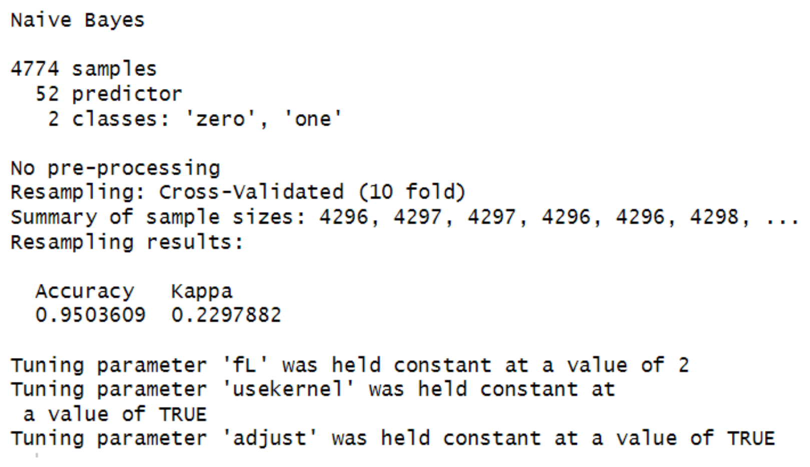
```
k-Nearest Neighbors

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4298, 4297, 4296, 4296, 4298, 4296, ...
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   1  0.9541239  0.26783166
   2  0.9526626  0.22987034
   3  0.9675315  0.31194031
   4  0.9669043  0.30031991
   5  0.9677420  0.27588286
   6  0.9681604  0.27368109
   7  0.9687880  0.25450182
   8  0.9679508  0.24246966
   9  0.9685802  0.22476178
  10  0.9683701  0.21979625
  11  0.9673232  0.17910567
  45  0.9673249  0.01215397
  46  0.9673249  0.01215397
  47  0.9673249  0.01215397
  48  0.9671148  0.01110558
  49  0.9671148  0.00000000
  50  0.9671148  0.00000000

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 7.
```



K =7 was selected as the best tuning parameter for the KNN model.

11. Naive Bayes:

The Naive Bayes model is trained using the "nb" model indicating the Naive Bayes algorithm. The training process involves exploring different combinations of tuning parameters specified by the grid tuneGrid. In this case, the Naive Bayes model is configured to use a fixed number of laplace smoothing levels (fL = 2), enable kernel density estimation (usekernel = TRUE), and adjust the probabilities (adjust = TRUE).



```
Naive Bayes

4774 samples
  52 predictor
   2 classes: 'zero', 'one'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4296, 4297, 4297, 4296, 4296, 4298, ...
Resampling results:

  Accuracy   Kappa
  0.9503609  0.2297882

Tuning parameter 'fL' was held constant at a value of 2
Tuning parameter 'usekernel' was held constant at
 a value of TRUE
Tuning parameter 'adjust' was held constant at a value of TRUE
```

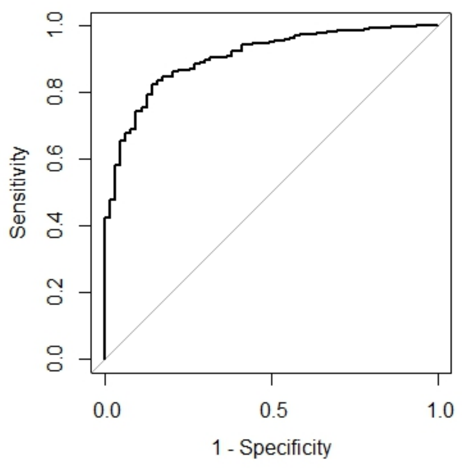
There are no tuning parameters for the Naive Bayes model.

SUMMARY STATISTICS

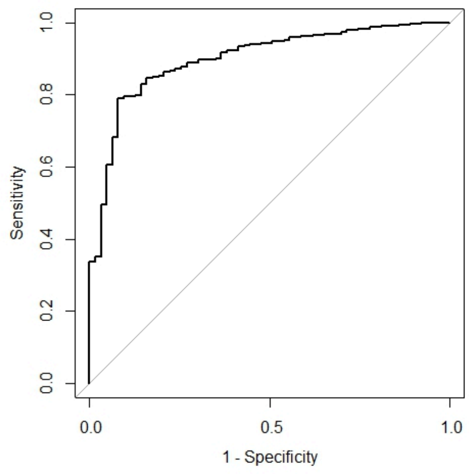
After training the models on train data and evaluating the models on testing data we found the following statistics for all of our models on the train and test datasets,

		Tainning		Test		
Model	**Best Paramaters**	**Accuracy**	**Kappa**	**Accuracy**	**Kappa**	**AUC**
Logistic Regression	-	0.9595695	0.197595	0.9076	0.268	0.7833
LDA	-	0.9602014	0.280483	0.9589	0.2423	0.871
Partial Least Squares	nComp = 1	0.9664859	0.009835056	0.9692	0.0289	0.8811
pinealized Model	alpha =0.1 and lambda = 0.2	0.967114	0	0.9692	0	0.8944
Nearest Shrunken Centroid	**threshold = 1.9**	**0.9673236**	**0.074897832**	**0.9707**	**0.1602**	**0.9042**
Non-Linear Disc. Analysis	Subclass = 4	0.9639702	0.3296859	0.9614	0.2425	0.8518
Neural Net Model	**Size = 6 and decay =2**	**0.9679526**	**0.2246237**	**0.9682**	**0.1864**	**0.8994**
Flexible Disc. Analysis	Degree =1 and nprune =2	0.9622957	0.2243746	0.9702	0.2365	0.8923
Support Vector Machine	Sigma = 0.009644574 and C = 0.5	0.9683705	0.15825416	0.9648	0.0881	0.8835
KNN	K = 7	0.968788	0.25450182	0.9653	0.1698	0.7591
Naive Bayes	-	0.9503609	0.2297882	0.9521	0.1975	0.8545

After evaluating multiple models, it was observed that the Nearest Shrunken Centroid model, the Neural Network model, and Support Vector Machine exhibited the best performance on the train data. Although the Nearest Shrunken Centroid model showed slightly superior results on the test data, with marginally higher accuracy and kappa values, the most noticeable difference was in the Area Under the Curve (AUC) of the ROC curve, where the Nearest Shrunken Centroid model outperformed. Despite similar accuracy and kappa values, the AUC metric highlighted the Nearest Shrunken Centroid model's better discriminatory ability.



Nearest Shrunken Centroid - AUC = 0.904



Neural Network Model - AUC = 0.8994

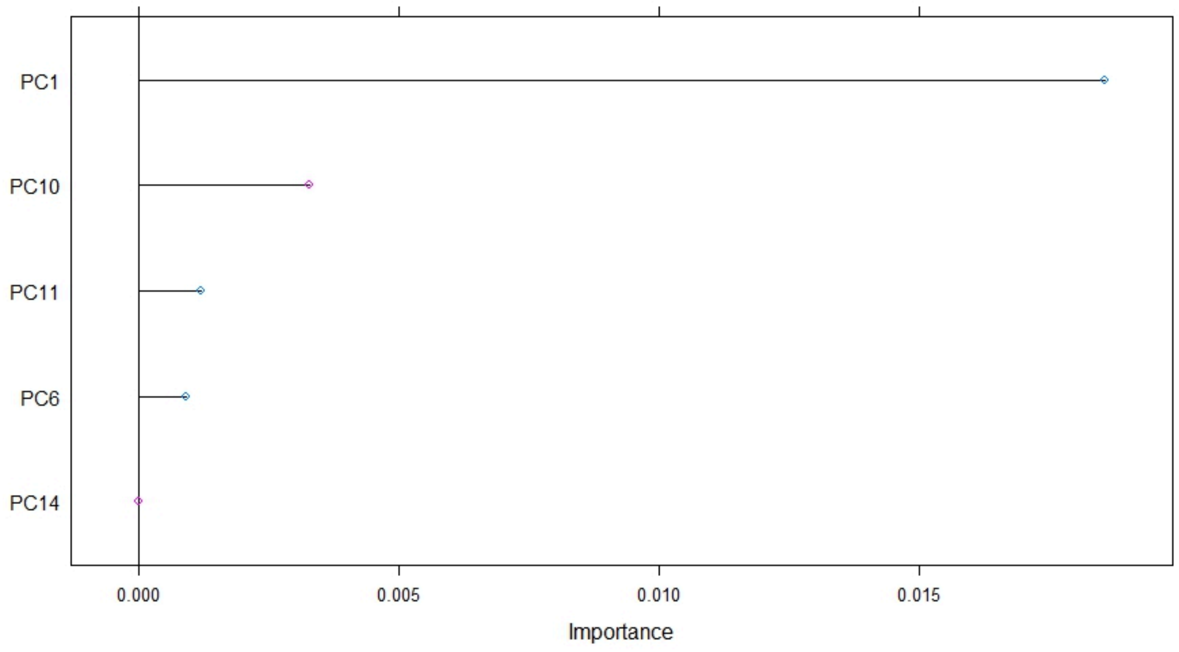
However, the Neural Network model incurred a significantly longer training time compared to the Nearest Shrunken Centroid model. Taking into consideration the trade-off between performance and computational efficiency, the decision was made to select the Nearest Shrunken Centroid model as the final model for the given task. This model is deemed optimal for its balance of high performance and reasonable computational requirements.

For the same model, we found the confusion matrix as shown below, "one's" represents bankruptcy and zero the non-bankrupt data points.

	Reference	
Prediction	**zero**	**one**
zero	1815	22
one	167	41

The sensitivity value for the model was 0.6508 and the specificity value was 0.9157. The Nearest Shrunken Centroid model was outperforming other models.

The Most Important Features of the Nearest Shrunken Centroid Model are.



CONCLUSION

Finally, after completing the necessary pre-processing and splitting the dataset into the right proportions for the training and test sets, we were able to train the model on a couple of linear and non-linear classifications. Out of these, we observed that the Nearest Shrunken Centroid model outperformed other models in accurately predicting companies that were going to go bankrupt.

References

1. *Data Mining for Predicting and Finding Factors of Bankruptcy | IEEE Conference Publication | IEEE Xplore*. (n.d.). Ieeexplore.ieee.org. Retrieved December 15, 2023, from https://ieeexplore.ieee.org/document/9775887

2. Leshno, M., & Spector, Y. (1996). Neural network prediction analysis: The bankruptcy case. Neurocomputing, 10(2), 125–147. https://doi.org/10.1016/0925-2312(94)00060-3

R Code:

```
setwd("C:\\Users\\user\\Desktop\\sem-3\\predictive\\Project")

library(ggplot2)
library(knitr)
library(dplyr)
library(tidyr)
library(caret)
library(corrplot)
library(e1071)
library(nortest)

# loading the dataset
data <- read.csv("taiwanese_bankruptcy_prediction.csv")
summary(data)

#sample size number of predectors and responce
dim(data)

#Histogram plaot if responce vairable Y
hist(data$Bankrupt)

#our data doesnt have any null values
data %>% is.na() %>% colSums()
total_data <- data

#################

#categorical varieble
histogram(total_data$Liability.Assets.Flag, main = 'Percentage distribution of flags in Liability.Assets.Flag')
histogram(total_data$Net.Income.Flag, main = 'Percentage distribution of flags in Net.Income.Flag')
```

```
##################
# remove all the caegorical variables.
zeroVar_cat = nearZeroVar(total_data)
zeroVar_cat

##############
# Box plots on the predictor variabels
y <- total_data$Bankrupt
data <- data[, !colnames(data) %in% c("Bankrupt","Liability.Assets.Flag",
"Net.Income.Flag")]

par(mfrow = c(6,6))
for (i in colnames(data[, 1:36])){
  hist(data[[i]], main = i , xlab = i)
}

par(mfrow = c(6,6))
for (i in colnames(data[, 36:71])){
  hist(data[[i]], main = i , xlab = i)
}

par(mfrow = c(6,6))
for (i in colnames(data[, 72:93])){
  hist(data[[i]], main = i , xlab = i)
}

##############
## check for skewness in data before prep-processing
skewValues <- apply(data, 2, skewness)
skew_matrix <- matrix(skewValues, nrow = 93, ncol = 1, byrow = TRUE)
rownames(skew_matrix) <- names(data)
kable(skew_matrix, format = "pipe", col.names = 'skewness')

################
```

```
#corelation plot before transformation
par(mfrow = c(1,1))
cor_pred = cor(data)
corrplot(cor_pred)# order = 'hclust')

##########################

library(MASS)    # For generating multivariate data
library(ggplot2)  # For plotting

#applying transformation boxcox -venkat
transformed_data <- lapply(names(data), function(x) {
  bxcx = BoxCoxTrans(data[,x])
  predict(bxcx, data[,x])
})

transformed_data <- data.frame(sapply(transformed_data,c))
colnames(transformed_data) <- names(data)

skewValues_trans <- apply(transformed_data, 2, skewness)
skew_matrix_trans <- matrix(skewValues_trans, nrow = 93, ncol = 1, byrow = TRUE)
rownames(skew_matrix_trans) <- names(transformed_data)
kable(skew_matrix_trans, format = "pipe", col.names = 'skewness')


#spatial sign transformation
transformed_data <- as.data.frame.matrix(spatialSign(transformed_data))

skewValues <- apply(transformed_data, 2, skewness)
skew_matrix <- matrix(skewValues, nrow = 93, ncol = 1, byrow = TRUE)
rownames(skew_matrix) <- names(data)
kable(skew_matrix, format = "pipe", col.names = 'skewness')

# Create a data frame for plotting
```

```
original_df <- as.data.frame(data)
original_df_name <- names(data)
transformed_df <- as.data.frame(transformed_data)

par(mfrow = c(4,2))
lapply(1:length(original_df_name), function(i) {
  col_name <- original_df_name[[i]]
  boxplot(original_df[, i], main=col_name, xlab=col_name)
  mlabel <- paste('after transformation of',col_name)
  boxplot(transformed_df[, i], main=mlabel, xlab=col_name)
})

#performing PCA after removing correlated data
trans = preProcess(data, method = c("BoxCox", "center", "scale", "pca"))
trans

# Apply the transformations:
transformed <- predict(trans, data)
dim(data) #before
dim(transformed) #after
str(transformed)


###############
# performign PCA on the dataset
XX <- transformed
XX
sigma=var(XX) ## variance-covariance matrix
sigma
QQ=eigen(sigma) ## calculate eigen values and eigen vectors of sigma
lamda=QQ$values
AA=QQ$vectors

##first colume of AA is the first eigen vector corresponding to the first eigen valus.
PCs=as.matrix(XX)%*%as.matrix(AA)
```

```
PCs

propotion=cumsum(lamda)/sum(lamda)
propotion
par(mfrow = c(1,1))
plot(propotion)
###############

#corelation plot after before
par(mfrow = c(1,1))
cor_pred = cor(PCs)
corrplot(cor_pred)# order = 'hclust')

#########################
# model building
set.seed(5790)
index <- createDataPartition(y, p = 0.7, list = FALSE)


xtrain = XX[index,]
ytrain = factor(y[index], labels = c("zero", "one"))
xtest = XX[-index,]
ytest = factor(y[-index], labels = c("zero", "one"))

##################### Linear Models
#models

## 10-fold cross validation
ctrl <- trainControl(method = "cv",
                 number = 10,
                 classProbs = TRUE,
                 summaryFunction = defaultSummary,
                 savePredictions = TRUE)
```

```
# 1. Logistic Reg.

lrFull <- train(xtrain,
             y = ytrain,
             method = "glm",
             metric = "Kappa",
             trControl = ctrl)
lrFull

#plot(lrFull) #no tuning parameters for this model.
lr_pred <- predict(lrFull, xtest)
postResample(lr_pred, ytest)
confusionMatrix(lr_pred,ytest)

ROC_prob <- predict(lrFull, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
               predictor = ROC_prob[, 2],
               levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)

# 2. Linear Disc. Analysis
LDAFull <- train(xtrain,
             y = ytrain,
             method = "lda",
             metric = "ROC",
             trControl = ctrl)

LDAFull

#plot(LDAFull) #no tuning parameters for this model.
lda_pred <- predict(LDAFull, xtest)
confusionMatrix(data =lda_pred,reference=as.factor(ytest))
```

```
ROC_prob <- predict(LDAFull, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
               predictor = ROC_prob[, 2],
               levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)

# 3. PLS
plsFit2 <- train(xtrain,
              y = ytrain,
              method = "pls",
              tuneGrid = expand.grid(.ncomp = 1:4),
              preProc = c("center","scale"),
              metric = "ROC",
              trControl = ctrl)

plsFit2
plot(plsFit2)

pls_pred <- predict(plsFit2, xtest)
confusionMatrix(data =pls_pred,reference=as.factor(ytest))

ROC_prob <- predict(plsFit2, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
               predictor = ROC_prob[, 2],
               levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)

# 4. pinealized models
glmnGrid <- expand.grid(.alpha = c(0, .1, .2, .4, .6, .8, 1),
                   .lambda = seq(.01, .2, length = 10))
glmnTuned <- train(xtrain,
               y = ytrain,
                method = "glmnet",
                tuneGrid = glmnGrid,
                metric = "ROC",
                trControl = ctrl)
glmnTuned

plot(glmnTuned)
glmn_pred <- predict(glmnTuned, xtest)
confusionMatrix(data =glmn_pred,reference=as.factor(ytest))

ROC_prob <- predict(glmnTuned, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
                predictor = ROC_prob[, 2],
                levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)


# 5. Nearest Shrunken centriod

nscGrid <- data.frame(.threshold = seq(0,4, by=0.1))
nscTuned <- train(xtrain,
               y = ytrain,
               method = "pam",
               tuneGrid = nscGrid,
               metric = "ROC",
               trControl = ctrl)
nscTuned
plot(nscTuned)
nsc_pred <- predict(nscTuned, xtest)
confusionMatrix(data =nsc_pred,reference=as.factor(ytest))

ROC_prob <- predict(nscTuned, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
                predictor = ROC_prob[, 2],
              levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)

##################### Non Linear Models
##1. Non-Linear Disc.
mdaFit <- train(x = xtrain,
            y = ytrain,
            method = "mda",
            metric = "ROC",
            tuneGrid = expand.grid(.subclasses = 1:10),
            trControl = ctrl)
mdaFit
plot(mdaFit)

mdaPred <- predict(mdaFit, xtest)
postResample(mdaPred, ytest)
confusionMatrix(mdaPred, ytest)

ROC_prob <- predict(mdaFit, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
              predictor = ROC_prob[, 2],
              levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)

##2. Neural Network
library(nnet)
library(caret)
## Using the formula interface to fit a simple model:
nnetGrid <- expand.grid(.size = 1:10, .decay = c(0, .1, 1, 2))
maxSize <- max(nnetGrid$.size)
numWts <- (maxSize * (52 + 1) + (maxSize+1)*2) ## 4 is the number of predictors use
when
```

```
nnetFit <- train(x = xtrain,
                 y = ytrain,
                 method = "nnet",
                 metric = "ROC",
                 tuneGrid = nnetGrid,
                 trace = FALSE,
                 maxit = 2000,
                 MaxNWts = numWts,
                 trControl = ctrl)
nnetFit
plot(nnetFit)

nnetPred <- predict(nnetFit, xtest)

postResample(nnetPred, ytest)

confusionMatrix(nnetPred, ytest)

ROC_prob <- predict(nnetFit, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
                   predictor = ROC_prob[, 2],
                   levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)

##3. Flexible Disc
marsGrid <- expand.grid(degree = 1:3, nprune = 2:20)
fdaModel <- train(x = xtrain,
                  y = ytrain,
                  method = "fda",
                  metric = "ROC",
                  tuneGrid = marsGrid,
                  trControl = ctrl)

fdaModel
```

```
plot(fdaModel)

fdaPred <- predict(fdaModel, xtest)
postResample(fdaPred, ytest)
confusionMatrix(data = fdaPred, reference=ytest)

ROC_prob <- predict(fdaModel, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
            predictor = ROC_prob[, 2],
            levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)



##4 SVM
library(kernlab)
library(caret)
sigmaRangeReduced <- sigest(as.matrix(xtrain))
svmRGridReduced <- expand.grid(.sigma = sigmaRangeReduced[1],
                  .C = 2^(seq(-4, 6)))
svmRModel <- train(x = xtrain,
            y = ytrain,
            method = "svmRadial",
            metric = "ROC",
            tuneGrid = svmRGridReduced,
            fit = FALSE,
            trControl = ctrl)
svmRModel
plot(svmRModel)

svmPred <- predict(svmRModel, xtest)
postResample(svmPred, ytest)
confusionMatrix(svmPred, ytest)

ROC_prob <- predict(svmRModel, xtest, type = "prob")
```

```
ROC_FullRoc <- roc(response = ytest,
             predictor = ROC_prob[, 2],
             levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)


##5 KNN
knnFit <- train(x = xtrain,
           y = ytrain,
           method = "knn",
           metric = "ROC",
           tuneGrid = data.frame(.k = 1:50),
           trControl = ctrl)
knnFit
plot(knnFit)

knnPred <- predict(knnFit, xtest)
postResample(knnPred, ytest)
confusionMatrix(knnPred, ytest)

ROC_prob <- predict(knnFit, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
             predictor = ROC_prob[, 2],
             levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)

##6 Naive Bayes
library(klaR)
nbFit <- train( x = xtrain,
           y = ytrain,
           method = "nb",
           metric = "ROC",
           tuneGrid = data.frame(.fL = 2,.usekernel = TRUE,.adjust = TRUE),
          trControl = ctrl)

nbFit

nbPred <- predict(nbFit, xtest)
postResample(nbPred, ytest)
confusionMatrix(nbPred, ytest)

ROC_prob <- predict(nbFit, xtest, type = "prob")
ROC_FullRoc <- roc(response = ytest,
            predictor = ROC_prob[, 2],
            levels = rev(levels(ytest)))
plot(ROC_FullRoc, legacy.axes = TRUE)
auc(ROC_FullRoc)

## feature importance

vimp <- varImp(nscTuned, scale = FALSE)
plot(vimp, top = 5, scales = list(y = list(cex = .95)))
```