Prediction of Company Bankruptcy with Machine Learning

Mahdi Karami

2023-02-02

1- Introduction

The present report aims to perform prediction of companies bankruptcy through machine learning. Multiple features of several companies have been collected from the Taiwan Economic Journal for the years 1999 to 2009. The class label includes the binary status of 'Survived'/'Bankrupted' for each company, where the classification task will be applied on. According to relatively high number of features in the database (95 features), the visualization of the data is challenging. Some statistical methods, in conjunction with correlation analysis, will be engaged to inspect and visualize the data. In addition, the proper orthogonal decomposition (POD) will be implemented to decompose the data into independent components (i.e., modes) which maximize the variance explained by data. Two well-known machine learning techniques will be implemented for classifications task namely, Decision Tree (DT) and Support Vector Machine (SVM). The mentioned algorithm will be applied for both original database and the decomposed database resulted by POD analysis. For DT models, the most important features will be introduced and visualized as well. All the analysis and algorithms are implemented in R programming language.

2- Data Preparation

The companies bankruptcy database have been collected from the Taiwan Economic Journal for the years 1999 to 2009. The original database is available on Kaggle website. This journal paper has also referenced the dataset. Alternatively, there is a copy of the bankruptcy dataset in my github.

Important Note!

The provided R script would download the zip file from the source. In the case of any error, please download the zip file source manually and put it in the root of the working directory (beside the R script).

3- Data Analysis

The given database is inspected in this section.

3-1- Overview of the Company Bankruptcy Database

There are totally 6819 rows and 96 columns in the dataset, where the first column indicates the class label and the remaining 95 columns are database features. The description of the features are listed below.

- X1 - ROA(C) before interest and depreciation before interest: Return On Total Assets(C)

- X2 - ROA(A) before interest and % after tax: Return On Total Assets(A)
- X3 - ROA(B) before interest and depreciation after tax: Return On Total Assets(B)
- X4 - Operating Gross Margin: Gross Profit/Net Sales
- X5 - Realized Sales Gross Margin: Realized Gross Profit/Net Sales
- X6 - Operating Profit Rate: Operating Income/Net Sales
- X7 - Pre-tax net Interest Rate: Pre-Tax Income/Net Sales
- X8 - After-tax net Interest Rate: Net Income/Net Sales
- X9 - Non-industry income and expenditure/revenue: Net Non-operating Income Ratio
- X10 - Continuous interest rate (after tax): Net Income-Exclude Disposal Gain or Loss/Net Sales
- X11 - Operating Expense Rate: Operating Expenses/Net Sales
- X12 - Research and development expense rate: (Research and Development Expenses)/Net Sales
- X13 - Cash flow rate: Cash Flow from Operating/Current Liabilities
- X14 - Interest-bearing debt interest rate: Interest-bearing Debt/Equity
- X15 - Tax rate (A): Effective Tax Rate
- X16 - Net Value Per Share (B): Book Value Per Share(B)
- X17 - Net Value Per Share (A): Book Value Per Share(A)
- X18 - Net Value Per Share (C): Book Value Per Share(C)
- X19 - Persistent EPS in the Last Four Seasons: EPS-Net Income
- X20 - Cash Flow Per Share
- X21 - Revenue Per Share (Yuan ¥): Sales Per Share
- X22 - Operating Profit Per Share (Yuan ¥): Operating Income Per Share
- X23 - Per Share Net profit before tax (Yuan ¥): Pretax Income Per Share
- X24 - Realized Sales Gross Profit Growth Rate
- X25 - Operating Profit Growth Rate: Operating Income Growth
- X26 - After-tax Net Profit Growth Rate: Net Income Growth
- X27 - Regular Net Profit Growth Rate: Continuing Operating Income after Tax Growth
- X28 - Continuous Net Profit Growth Rate: Net Income-Excluding Disposal Gain or Loss Growth
- X29 - Total Asset Growth Rate: Total Asset Growth
- X30 - Net Value Growth Rate: Total Equity Growth
- X31 - Total Asset Return Growth Rate Ratio: Return on Total Asset Growth
- X32 - Cash Reinvestment %: Cash Reinvestment Ratio
- X33 - Current Ratio
- X34 - Quick Ratio: Acid Test
- X35 - Interest Expense Ratio: Interest Expenses/Total Revenue
- X36 - Total debt/Total net worth: Total Liability/Equity Ratio
- X37 - Debt ratio %: Liability/Total Assets
- X38 - Net worth/Assets: Equity/Total Assets
- X39 - Long-term fund suitability ratio (A): (Long-term Liability+Equity)/Fixed Assets
- X40 - Borrowing dependency: Cost of Interest-bearing Debt
- X41 - Contingent liabilities/Net worth: Contingent Liability/Equity
- X42 - Operating profit/Paid-in capital: Operating Income/Capital
- X43 - Net profit before tax/Paid-in capital: Pretax Income/Capital
- X44 - Inventory and accounts receivable/Net value: (Inventory+Accounts Receivables)/Equity
- X45 - Total Asset Turnover
- X46 - Accounts Receivable Turnover
- X47 - Average Collection Days: Days Receivable Outstanding
- X48 - Inventory Turnover Rate (times)
- X49 - Fixed Assets Turnover Frequency
- X50 - Net Worth Turnover Rate (times): Equity Turnover
- X51 - Revenue per person: Sales Per Employee
- X52 - Operating profit per person: Operation Income Per Employee
- X53 - Allocation rate per person: Fixed Assets Per Employee
- X54 - Working Capital to Total Assets
- X55 - Quick Assets/Total Assets

- X56 - Current Assets/Total Assets
- X57 - Cash/Total Assets
- X58 - Quick Assets/Current Liability
- X59 - Cash/Current Liability
- X60 - Current Liability to Assets
- X61 - Operating Funds to Liability
- X62 - Inventory/Working Capital
- X63 - Inventory/Current Liability
- X64 - Current Liabilities/Liability
- X65 - Working Capital/Equity
- X66 - Current Liabilities/Equity
- X67 - Long-term Liability to Current Assets
- X68 - Retained Earnings to Total Assets
- X69 - Total income/Total expense
- X70 - Total expense/Assets
- X71 - Current Asset Turnover Rate: Current Assets to Sales
- X72 - Quick Asset Turnover Rate: Quick Assets to Sales
- X73 - Working capital Turnover Rate: Working Capital to Sales
- X74 - Cash Turnover Rate: Cash to Sales
- X75 - Cash Flow to Sales
- X76 - Fixed Assets to Assets
- X77 - Current Liability to Liability
- X78 - Current Liability to Equity
- X79 - Equity to Long-term Liability
- X80 - Cash Flow to Total Assets
- X81 - Cash Flow to Liability
- X82 - CFO to Assets
- X83 - Cash Flow to Equity
- X84 - Current Liability to Current Assets
- X85 - Liability-Assets Flag: 1 if Total Liability exceeds Total Assets, 0 otherwise
- X86 - Net Income to Total Assets
- X87 - Total assets to GNP price
- X88 - No-credit Interval
- X89 - Gross Profit to Sales
- X90 - Net Income to Stockholder's Equity
- X91 - Liability to Equity
- X92 - Degree of Financial Leverage (DFL)
- X93 - Interest Coverage Ratio (Interest expense to EBIT)
- X94 - Net Income Flag: 1 if Net Income is Negative for the last two years, 0 otherwise
- X95 - Equity to Liability

By inspecting the database, it can be seen that the 95th column (i.e., the 94th feature: "Net.Income.Flag") does not change over all rows. Hence, one can suggest removing this feature since it does not bring any information/difference about the class labels. Accordingly, our database possesses 94 features now. Due to the large number of features with some similar names, the feature number is selected as the name of the feature for understanding and presentation. For convenience, the name of class label column is changed into "label" as well.

3-2- Variation in Data

The portion and number of data labels are indicated in Table 1 which shows a very large level of imbalance with only about 3% of positive label (bankrupted companies).

Table 1: Overview of Data Labels

label	Number	Portion(%)
0	6599	96.77372
1	220	3.22628

Figures 1 and 2 depict the histogram of the first 18 features of the bankruptcy database. One can see that most of the features vary in the range of (0,1) while there are some features with a wide range of variations.

According to different range of the features, it is difficult to compare them in a unique measure. Hence, One can suggest normalizing all the features by engaging a linear transformation to limit all the values in the range (0,1).

Figures 3 and 4 depict the variations of all features for normalized database, which are obtained by boxplots. One can see a considerable difference in the variation of the features. Some feature has a relatively wide range of variation which enables them to provide more explained variance for classification analysis. Some other features, on the other hand, vary in a very narrow range, that make it difficult to use them for classification.

The standard deviation is a very appropriate measure for a normalized database for comparing the variation of the features. Figure 5 illustrates the standard deviation values for the features with the most standard deviations, descendingly sorted.

As depicted in Figure 5, the first four features with the most standard deviation are respectively #72, #48, #11, and #74. Figure 6 illustrates the 2D plots of class labels for different combinations of the four mentioned features. The red points indicate the bankrupted cases while the blue circles show the survived companies. One can see that the class labels are completely mixed in the given plots and cannot be simply separated by these features.

3-3- Calculation of Correlations

Correlations play an important role in data classification, especially for those with too many features. If two features are highly correlated ($|corr| \approx 1$), it means that their change are always in the same direction (or opposite for $ corr \approx -1$). Thus, one can conclude that one of the two features is redundant since it does not add any information to the data. In addition, a strong correlation of a feature with the class label demonstrates the importance of that feature which is able to predict the value of the label.

The following equation gives the correlation of two given vectors, u_1 and u_2.

$$corr(u_1, u_2) = \frac{\sum_{i=1}^{N} (u_{1,i} - \bar{u}_1) * (u_{2,i} - \bar{u}_2)}{\sqrt{(\sum_{i=1}^{N} (u_{1,i} - \bar{u}_1)^2)} * \sqrt{(\sum_{i=1}^{N} (u_{2,i} - \bar{u}_2)^2)}}$$

where $\bar{X}$ indicates the average of vector X, and N is the length of given vectors. The correlation value always varies between -1 and +1.

The correlations of all combinations of the features are indicated in Figure 7. The correlation of the features with the class label are depicted in the plot as well.

As shown in Figure 7, the strong correlations among the features are relatively rare. Most of the correlations fall in values near zero. Also, it can be seen that few features have strong correlation with the class label. The distribution of the correlations are shown in Figure 8 in a histogram plot. There totally 181 ($= 175 + 6$) strong correlations ($|corr| > 0.8$) in the database, where 95 of them are the correlations of each column to itself.

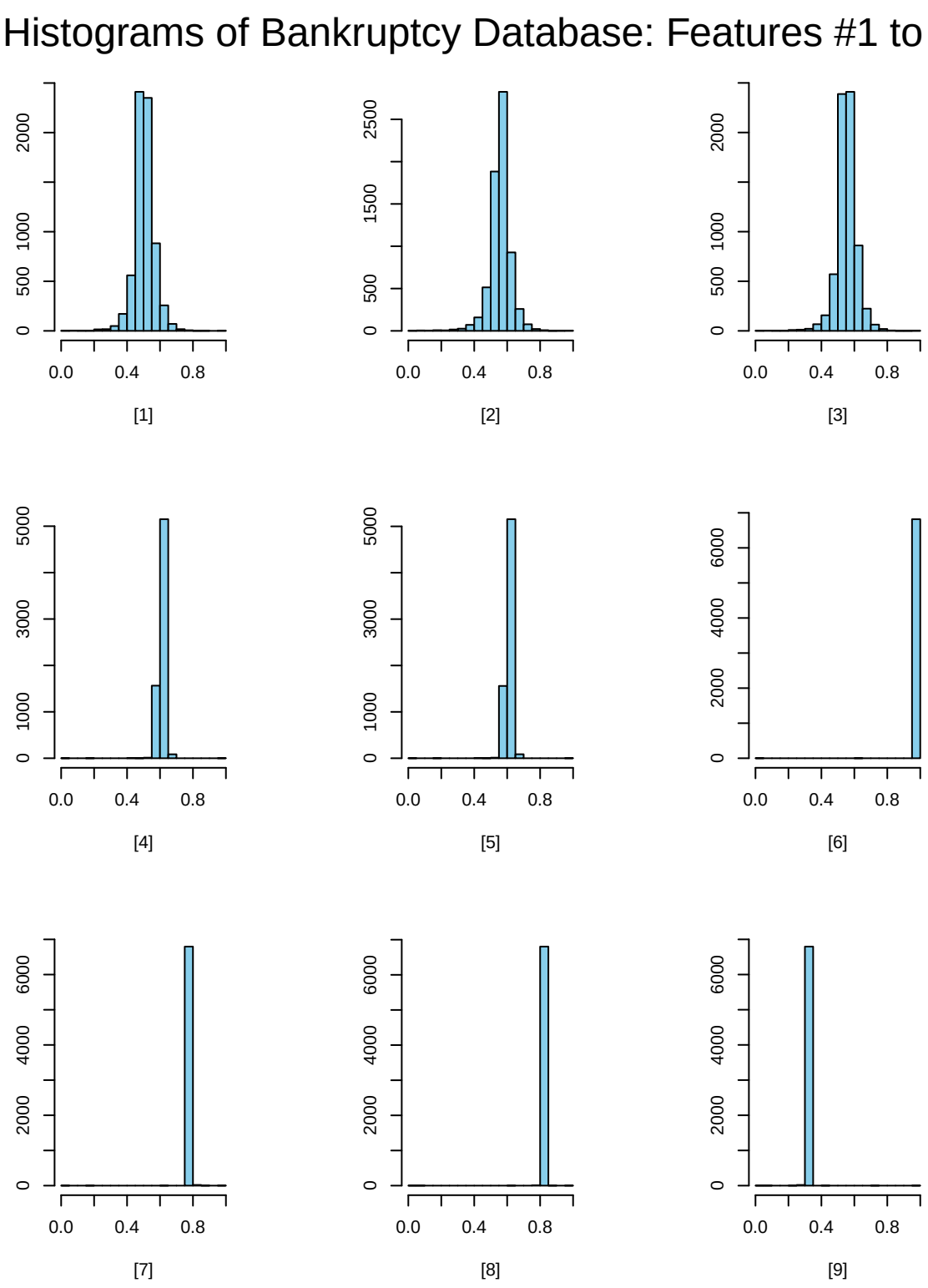

Figure 1: Histograms of Bankruptcy Database: Features #1 to #9

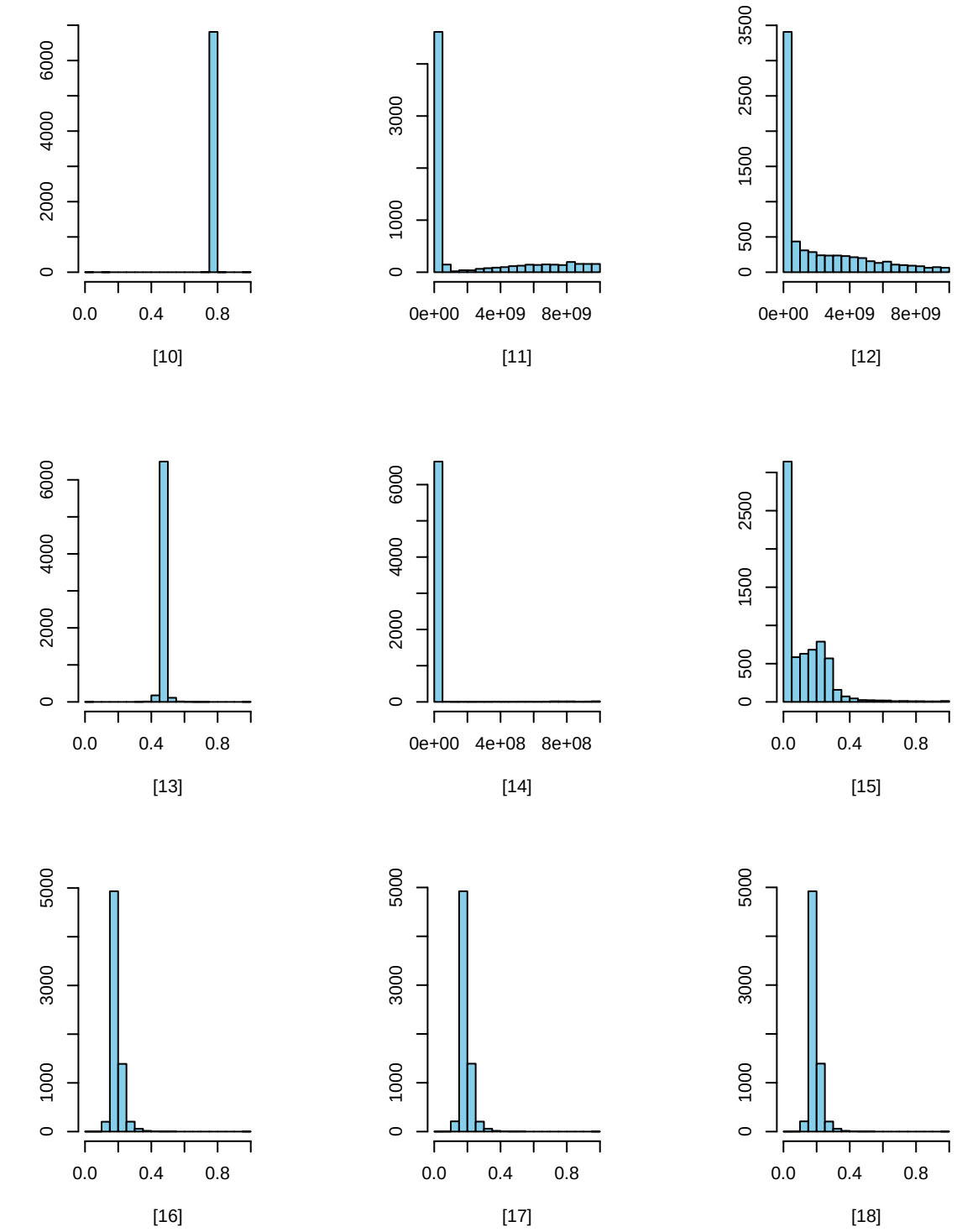

Figure 2: Histograms of Bankruptcy Database: Features #10 to #18

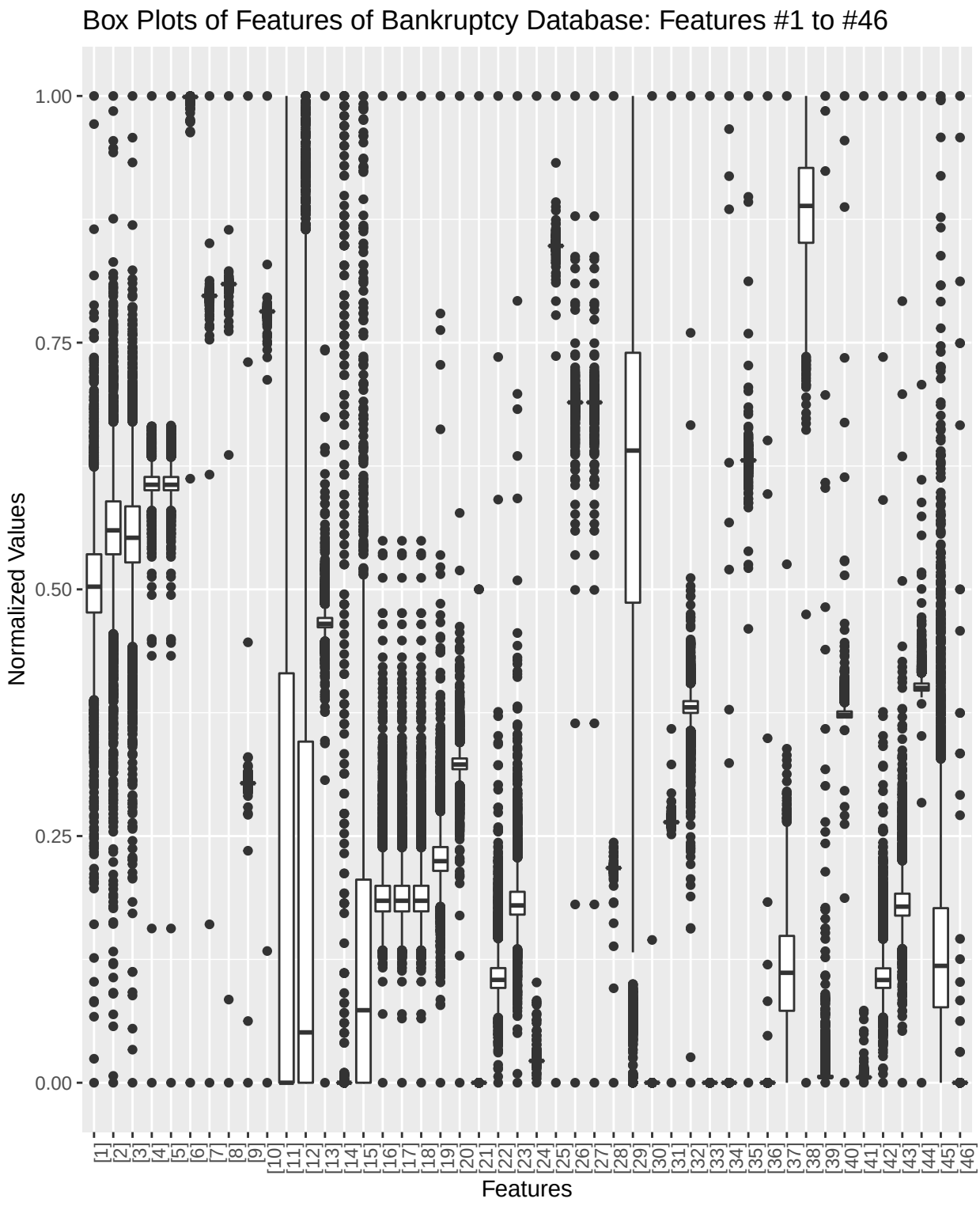

Figure 3: Box Plots of Normalized Features of Bankruptcy Database: Features #1 to #46

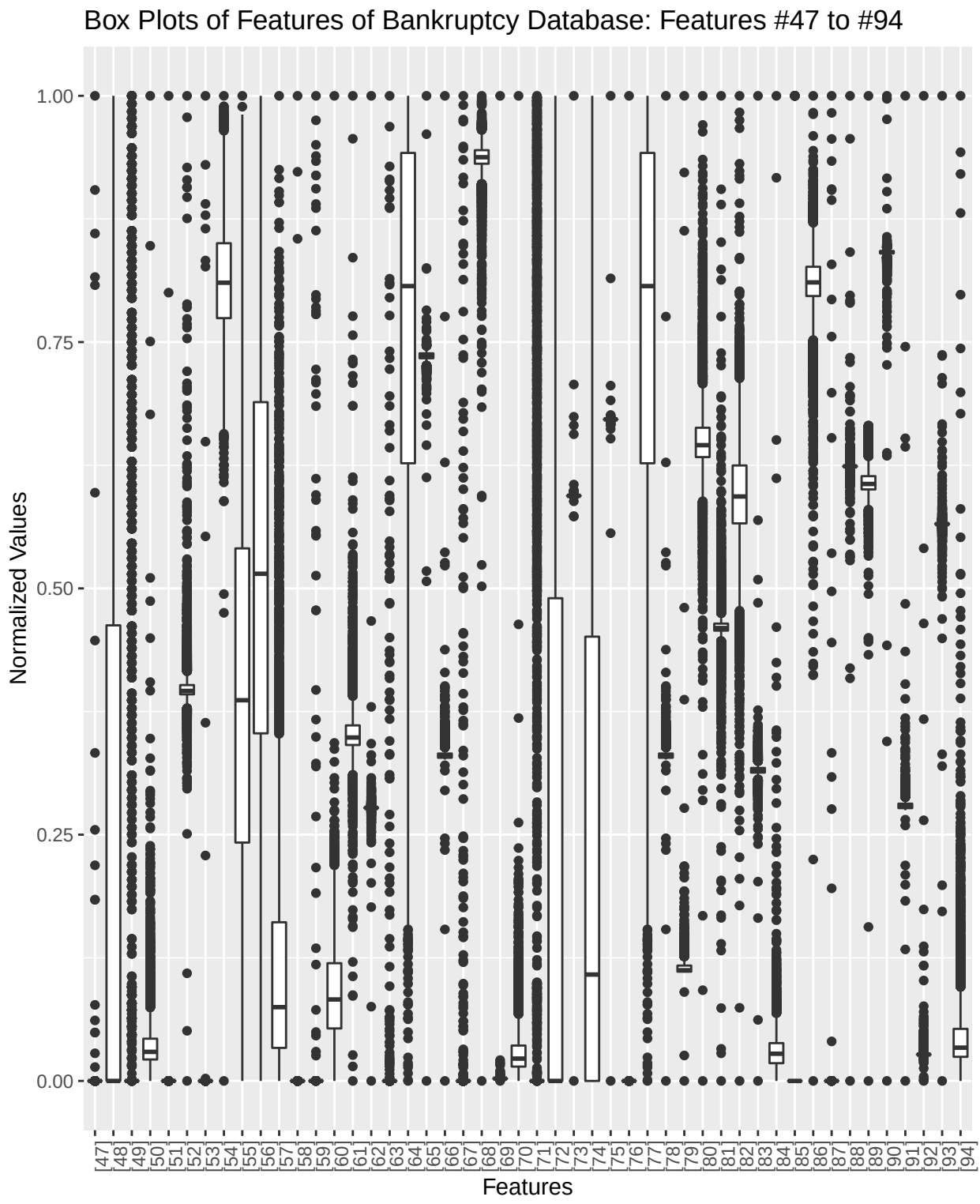

Figure 4: Box Plots of Normalized Features of Bankruptcy Database: Features #47 to #94

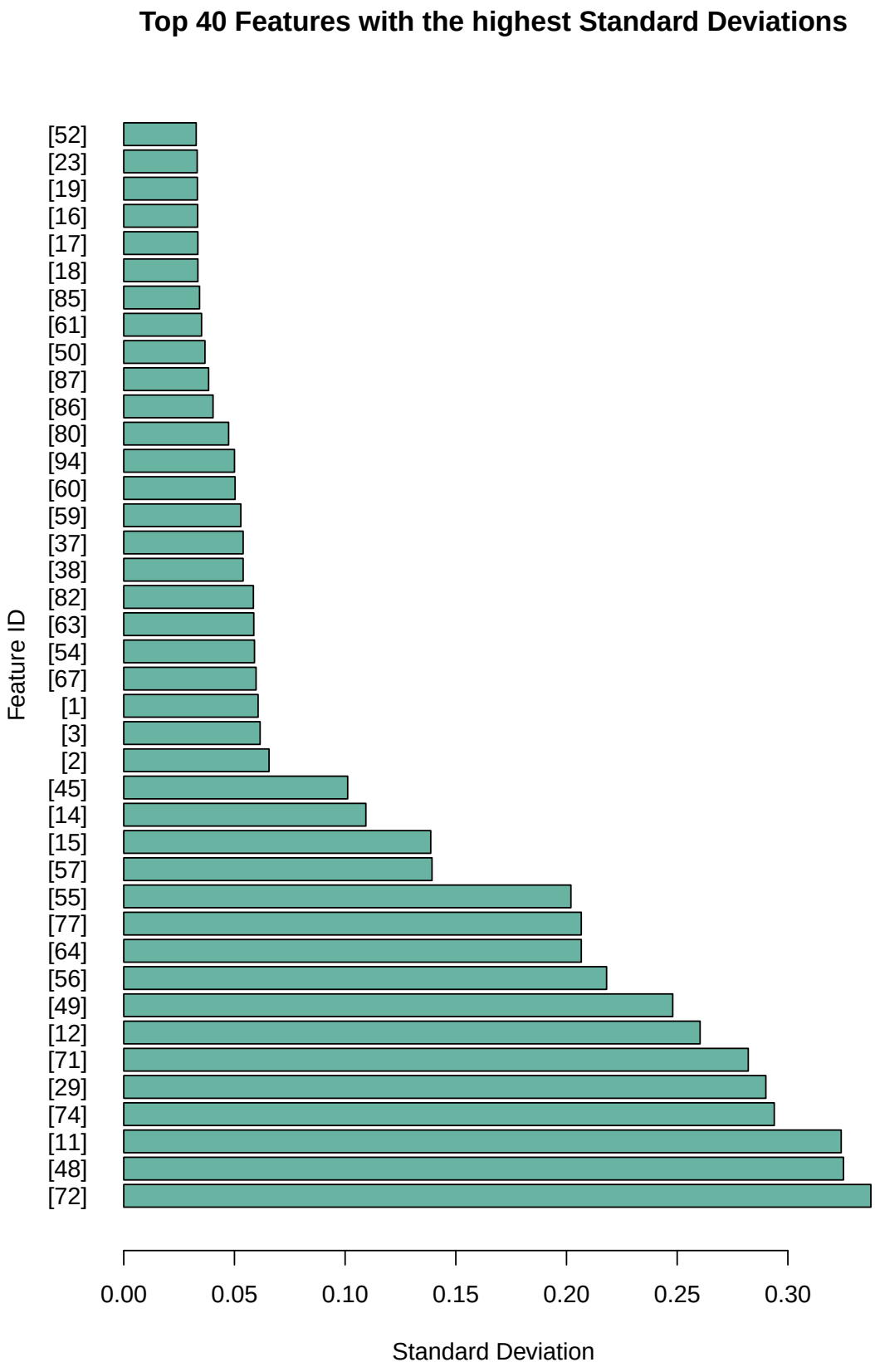

Figure 5: Top 40 Features with the highest Standard Deviations

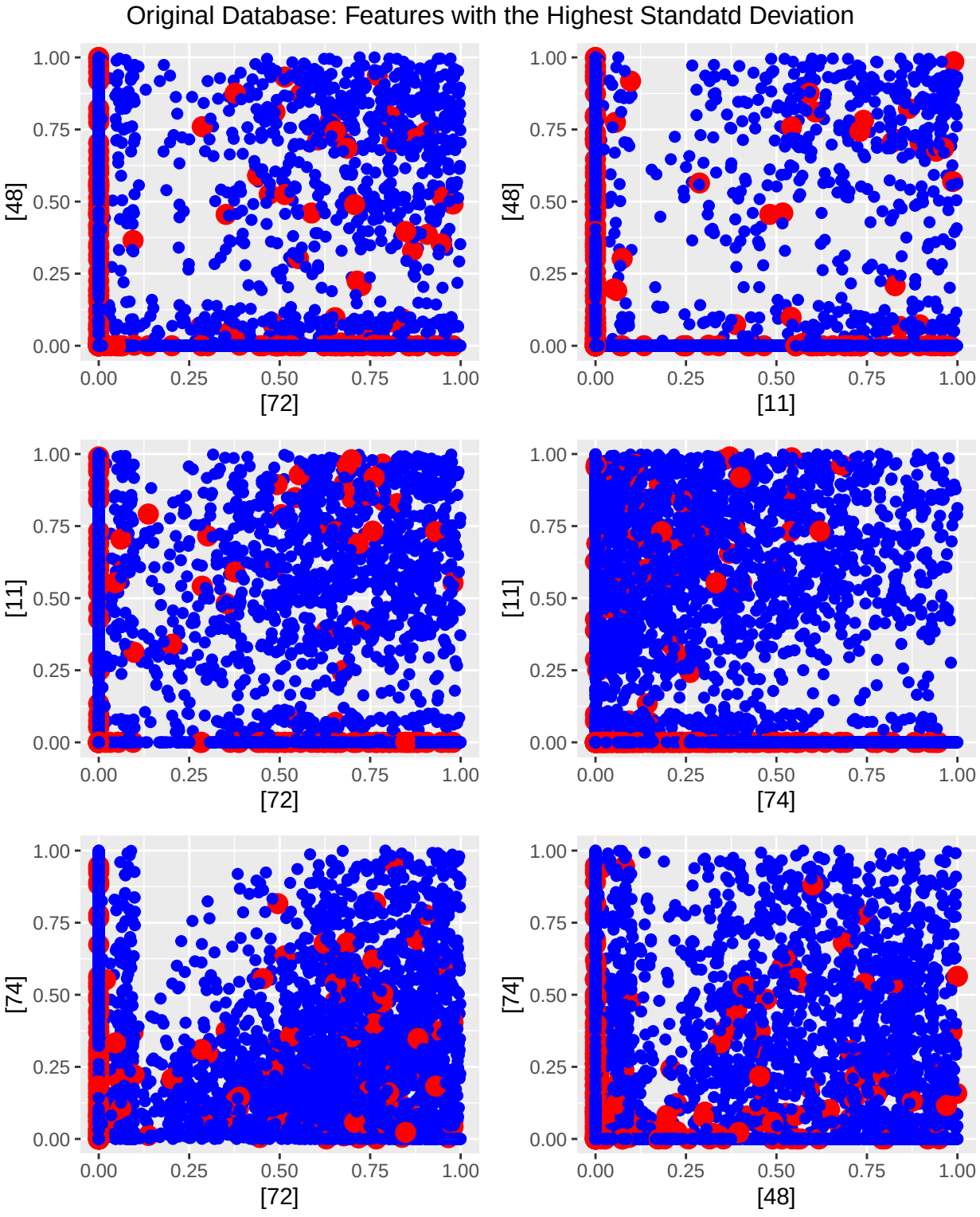

Figure 6: Original Database: Features with the Highest Standatd Deviation

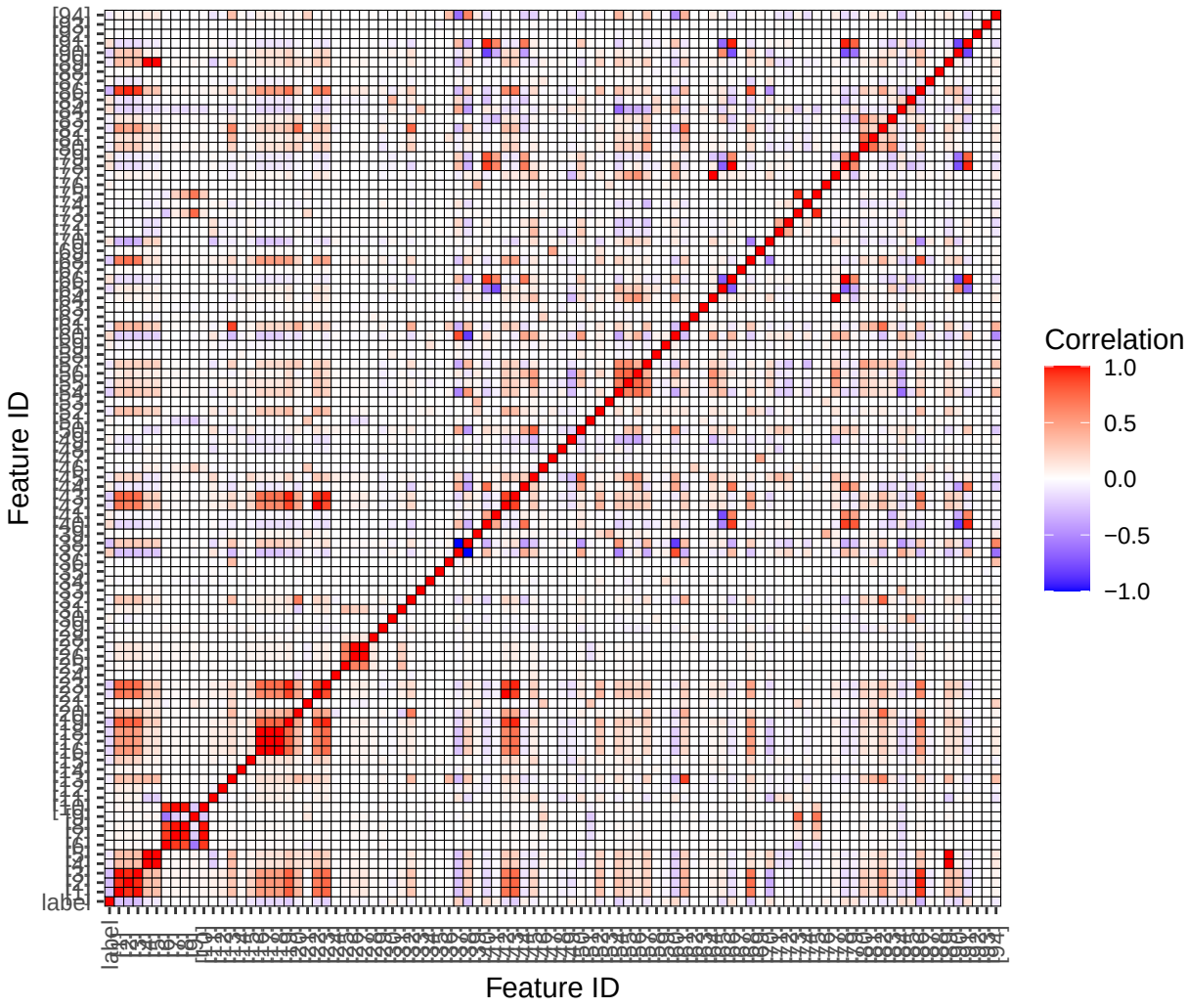

Figure 7: Contour of Cross-Correlations of Features

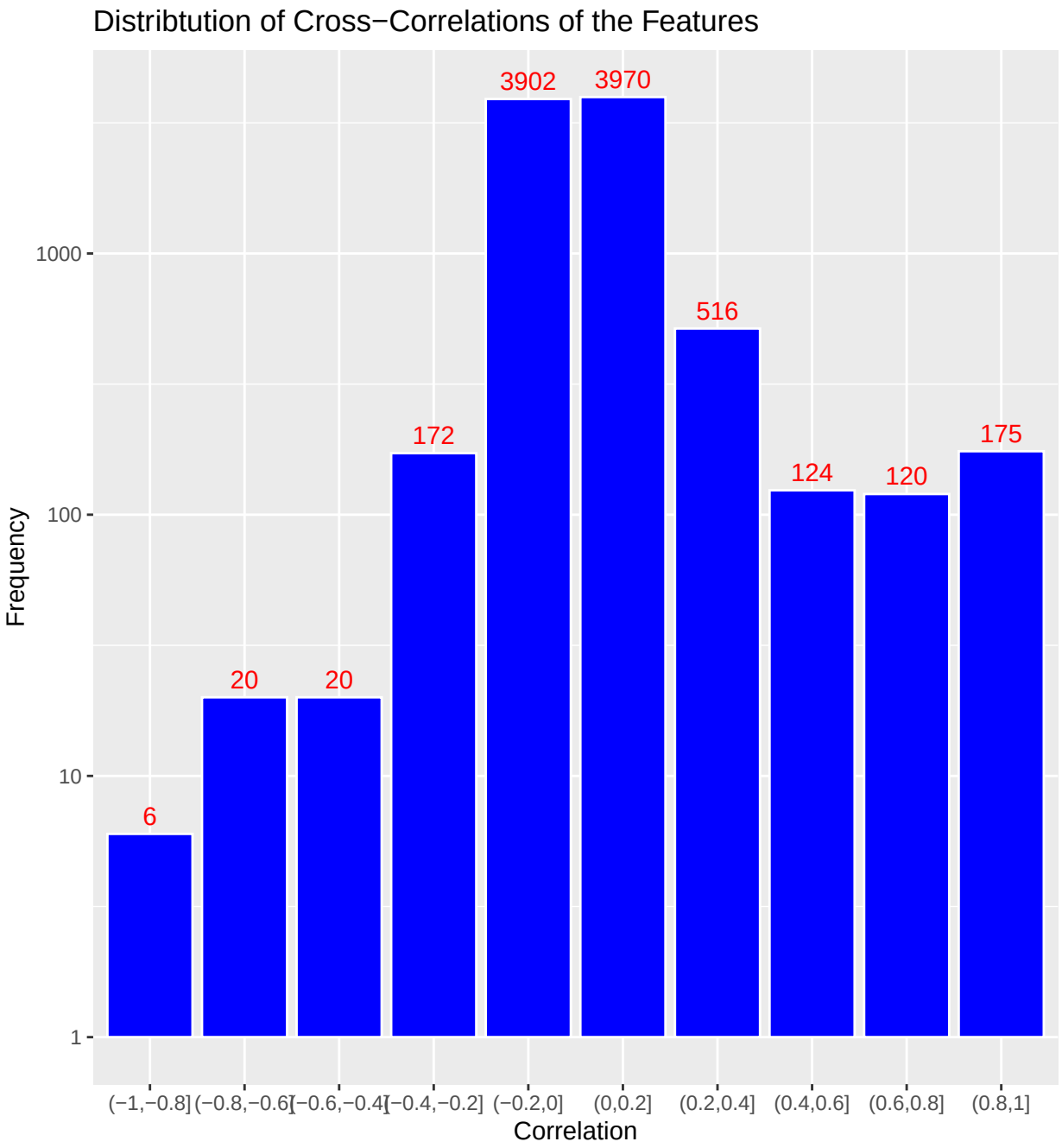

Figure 8: Histogram of Cross-Correlations of the Features

3-4- Decomposition of Database

Matrix decomposition is a strong data reduction techniques commonly used for databases with too many features. A very useful method is the proper orthogonal decomposition (POD), which decomposes a given matrix into orthogonal modes. For a given matrix A, the POD performs matrix decomposition by the following equation.

$$A^{(m*n)} = U^{m*m} * \Sigma^{m*n} * V^{T(n*n)}$$

where U and V are respectively the mode shape and temporal mode matrices. The diagonal elements of Σ indicate the contribution of each mode in the total matrix, usually being sorted from the largest mode to the smallest one. For actual databases in machine learning applications, the number of rows are greater than that of columns (i.e. $m > n$). Hence, most of the elements of Σ would be zero. The POD process is somehow similar to PCA (Principal Components Analysis), which finds the independent (i.e., orthogonal) directions with the maximum variations of data. The orthogonality of the directions of the POD axes leads to the minimum dependency of between the resulted modes. The number of modes is theoretically equal to the number of features, but for most of databases, only a few first mode can reconstruct the original database with a good estimation.

The contribution of each mode as well as the accumulated variance of modes are illustrated in Figures 9 and 10, respectively. One can easily see that the first mode alone contains about 50% of the contribution. In addition, it is seen that the first 25 modes can reconstruct more than 90% of the total variance.

Figure 11 presents the 2D distribution of class labels in the domains composed with the four largest principal components (i.e., modes). Comparing to the similar plot of the original database (i.e., Figure 6), one can see that the level of mixing of the labels is lower in POD components although they are still inseparable.

4- Data Classification

The classification of the bankruptcy database aims to distinguish the bankrupted companies from the survived ones by considering all the features. Two machine learning approaches will be examined namely, the decision tree (DT) and the support vector machine (SVM). In addition to the original database, the classification algorithms will be implemented for the decomposed database extracted from POD analysis.

4-1- Methodology

The decision tree is a classification model in machine learning which provides a sequence of queries (or tests) to distinguish a class label among another classes. The queries commonly consist of yes/no questions that guide the input row of data (with multiple columns or features) through a tree-like path to finally guess its class. The decision tree model is able to provide a good solution for many daily classification tasks in a reasonable time. A very important ability of DT model is that the resulted predicting model is completely interpretable since it is based on a series of simple conditions on the original features. On the other hand, it is not an isotropic model that makes it susceptible to the rotation of the axes of features domain. while DT has basically been designed for classification, it can be used for regression tasks as well. There are several hyper parameters in DT model to change the maximum height/depth of the tree, as well as the minimum number of data points for new breakdowns, that can be used to control the complexity of the model and prevent overfitting from happening. One of the most robust models for classification tasks is SVM which is among the mostly used algorithms in the current machine learning projects. The SVM tries to find lines (or supporting vectors) that classify the data with the maximum margins. For simple problems, it can result in a simple line (in 2D space), or a simple hyperplane (in multi-dimensional space), while several vectors might be engaged for complicated classification tasks. Some limitations of the SVM model include the difficulty of training hyper parameters to achieve an appropriate results, as well as its demanding computational cost.

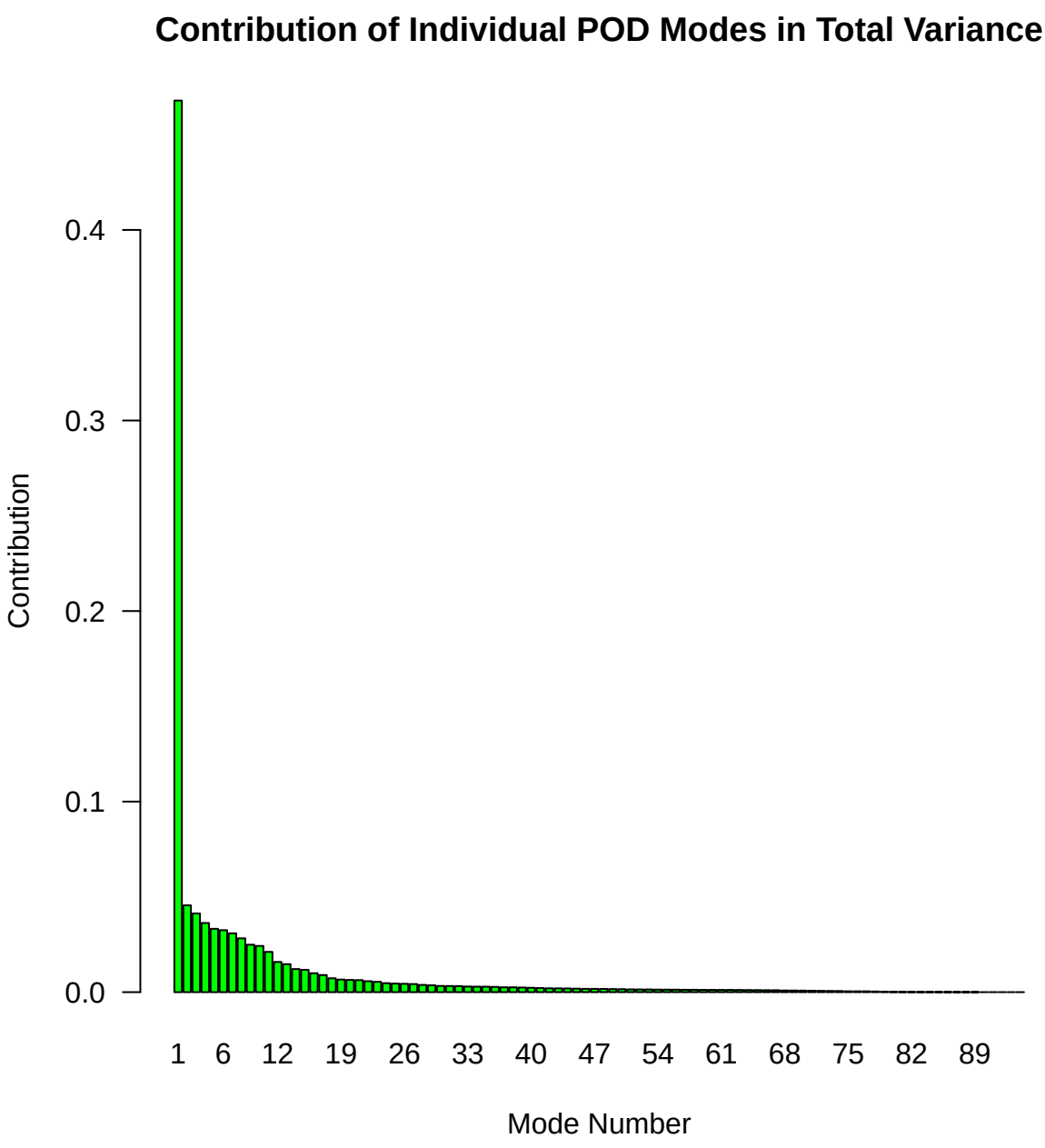

Figure 9: Contribution of Individual POD Modes in Total Variance

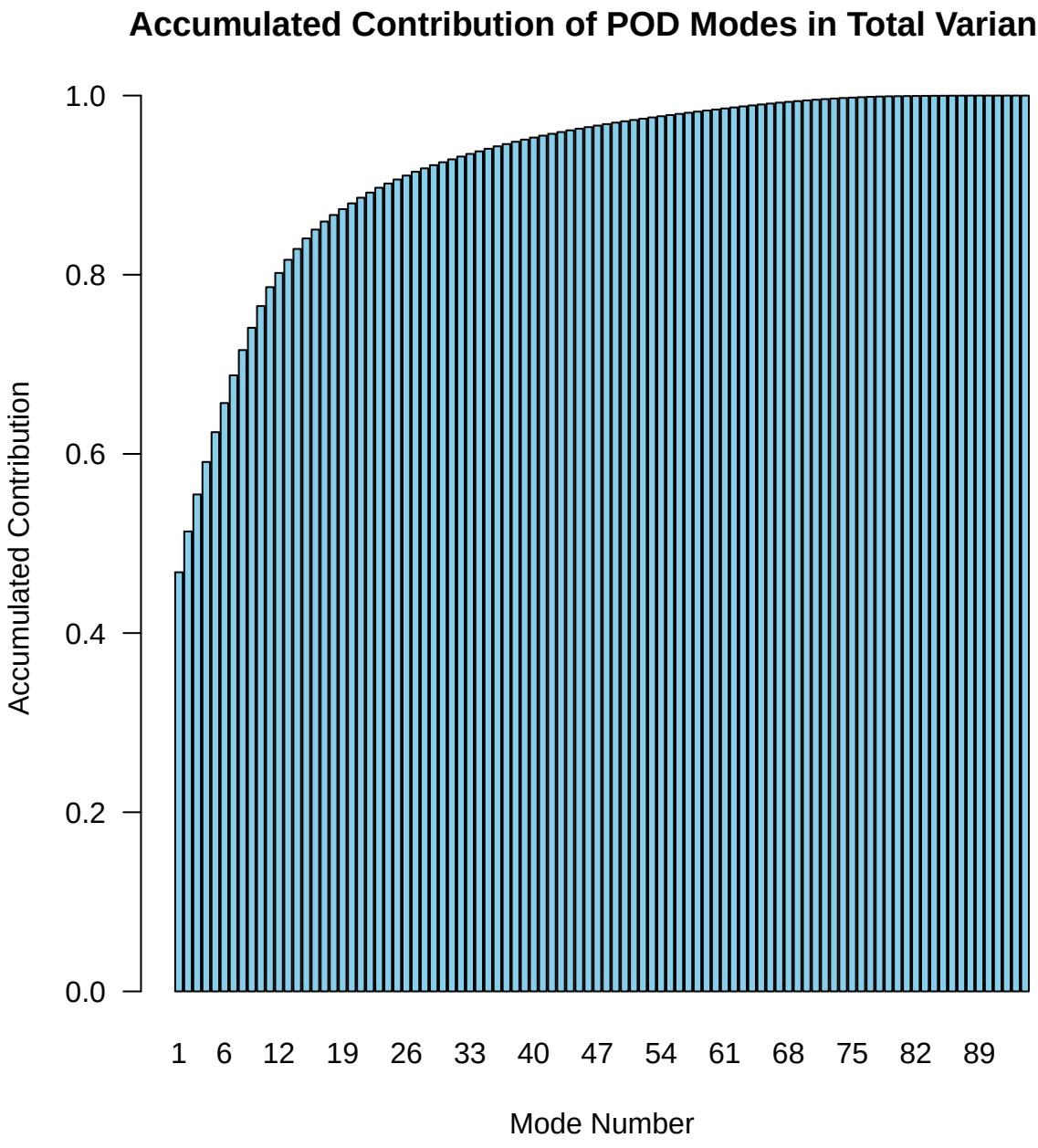

Figure 10: Accumulated Contribution of POD Modes in Total Variance

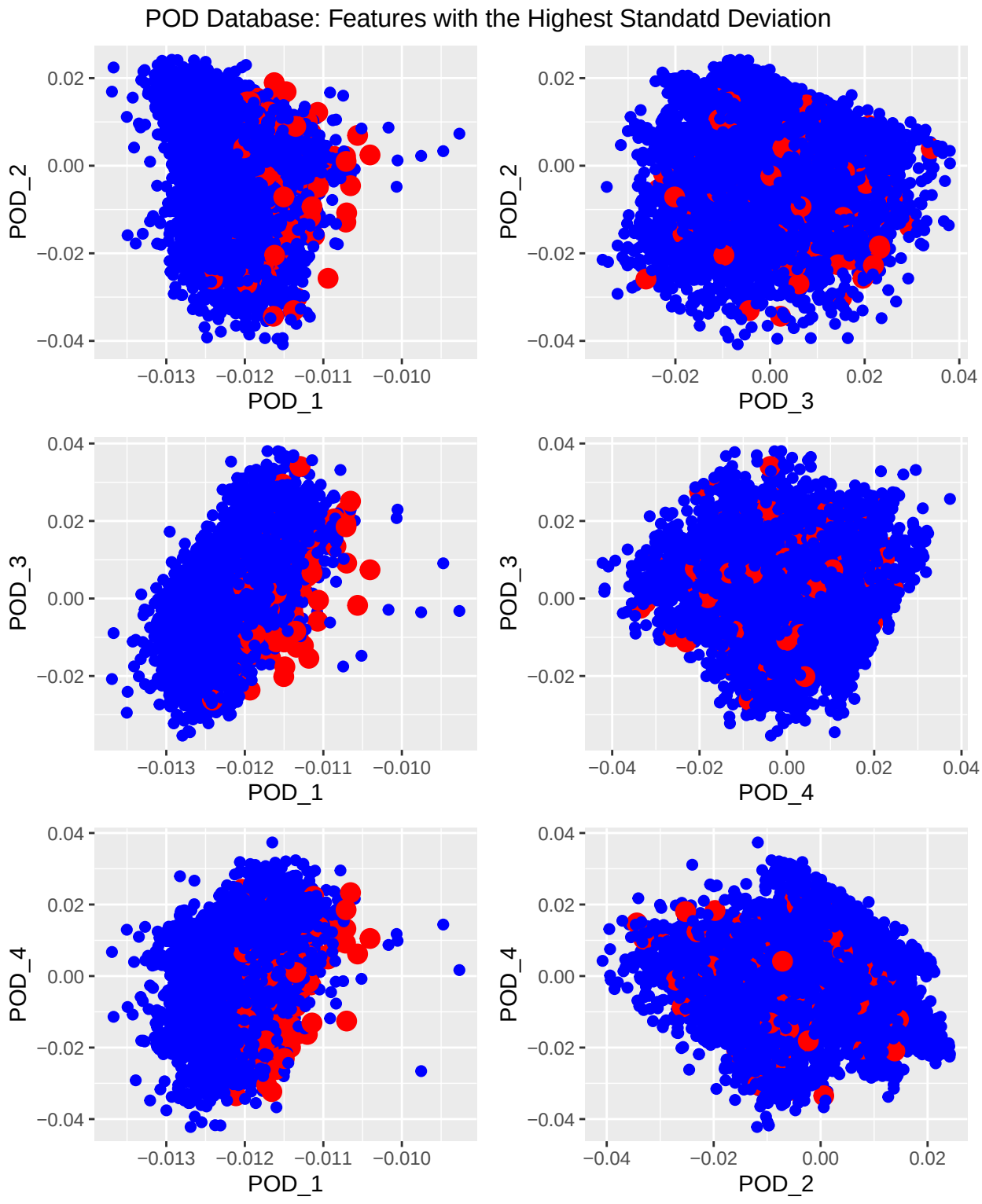

Figure 11: POD Database: Features with the Highest Standatd Deviation

Table 2: Schematic of a Confusion Matrix

	Actual Positive	Actual Negative
Predicted Positive	#TP	#FP
Predicted Negative	#FN	#TN

As described in Section 3-4, the POD decomposition is a strong method for dimensionality reduction in databases with too many features. Some classification problems show a very good performance when the decomposed database is considered as the input instead of the original one. the performance of classification of the mentioned algorithms (DT & SVM) will be examined on the POD results and compared to the performance of the original database. The whole database (for original and POD result) is split into train set (80%) and test set (20%) where the former is engaged to build the predicting model and the latter is used to evaluate the model accuracy. One of the basic tools for evaluating a predicting model is the confusion matrix, which is a square matrix of the of number of class labels that presents the number of predicted labels against the actual ones. For a simple database with binary labels, the confusion matrix is given in Table 2. Some important metrics of the predicting models are described below.

- Accuracy: Indicates the ratio of truly labeled data points to the total number of points. It is a very useful measure of the model performance, but should be considered with caution for databases with imbalanced class labels.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

- Precision: Represents the accuracy of positive labels reported by the predicting model. The equation below displays the calculation of Precision. It somehow means that how much we can trust the reported positive labels of the model.

$$Precision = \frac{TP}{TP + FP}$$

- Recall: Or Sensitivity, indicates the ability of model to predict the actual positive values.

$$Recall = \frac{TP}{TP + FN}$$

- Specificity: Represents the model performance in prediction actual negative labels. A Specificity of 80% for instance means that the predicting model is able to distinguish 8 of 10 negative data points in the dataset. It is similar to the Recall metric, but for the negative label.

$$Specificity = \frac{TN}{TN + FP}$$

- F1 Score: Considers the Recall and Precision metrics simultaneously, according to the formula below. It is called the harmonic average of Recall and Precision. Like other measures, this metric varies in range (0,1), and becomes unity for an ideal predictor. Based on the requirements of the problems, sometimes the weighted average of Recall and Precision is used to calculate F1 Score.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

Table 3: Confusion Matrix:DT-Original

	Actual 0	Actual 1
Predicted 0	1299	34
Predicted 1	17	14

Table 4: Confusion Matrix:DT-POD

	Actual 0	Actual 1
Predicted 0	1313	35
Predicted 1	9	7

It should be noted that choosing the right measure depends on the problem requirement. Also, the mentioned definitions and formula are completely depending on which label we assume positive. For the values indicated in the present report, it is assumed that the label '0', or 'Survived' is positive label.

There are multiple metrics (or measures) used to evaluate the performance of a predicting model, all extracted from the confusion matrix.

4-2- Results and Discussion

Tables 3 to 6 illustrate the confusion matrix of the predicting models (DT & SVM) on the available datasets (Original & POD). One can see that all the results are, more or less, in a similar level of accuracy. Table 7 displays the metrics (or measures) of the classifications task of each model-database combination. For better understanding of the model measures, it should be mentioned that the '0' (i.e., 'survived') class label is considered as the positive one, while '1'(i.e., 'Bankrupted') label is considered as negative. While all the models possess a high Accuracy, they suffer from relatively low Specificity. This occurs due to the imbalance of the class label, where only 3% of the companies in the database are bankrupted. Indeed, most of the models metrics show similar values near unity. The only measure that can distinguish among the models is the Specificity, which is useful when the negative class label is rare. this measure can provide a tool for us to judge the different models, or the same model with various hyper parameters to find the most accurate one. Thus, if we want to perform cross validation analysis to tune the model parameters, the Specificity should be targeted for the present problem.One can see that the DT model with the original database gives the best Specificity while the worst prediction is achieved by SVM with the POD result. The DT-POD and SVM-Original show the same performance in predicting the negative classes correctly.

Figures 12 and 13 depict the schematic of the decision tree model for both original database and its POD output, respectively. The maximum depth of the current models is 7, which means that longest path of the tree includes 7 check points. The complexity and depth of the the decision tree can be controlled by its hyper parameters.

The decision tree model can also report the most important features for building the predicting model. The 6 most important features of the decision tree models for the original and POD data frames are displayed in Table 8. the important features of the original dataset are listed below. - X30 - Net Value Growth Rate: Total Equity Growth - X90 - Net Income to Stockholder's Equity - X19 - Persistent EPS in the Last Four

Table 5: Confusion Matrix:SVM-Original

	Actual 0	Actual 1
Predicted 0	1302	40
Predicted 1	14	8

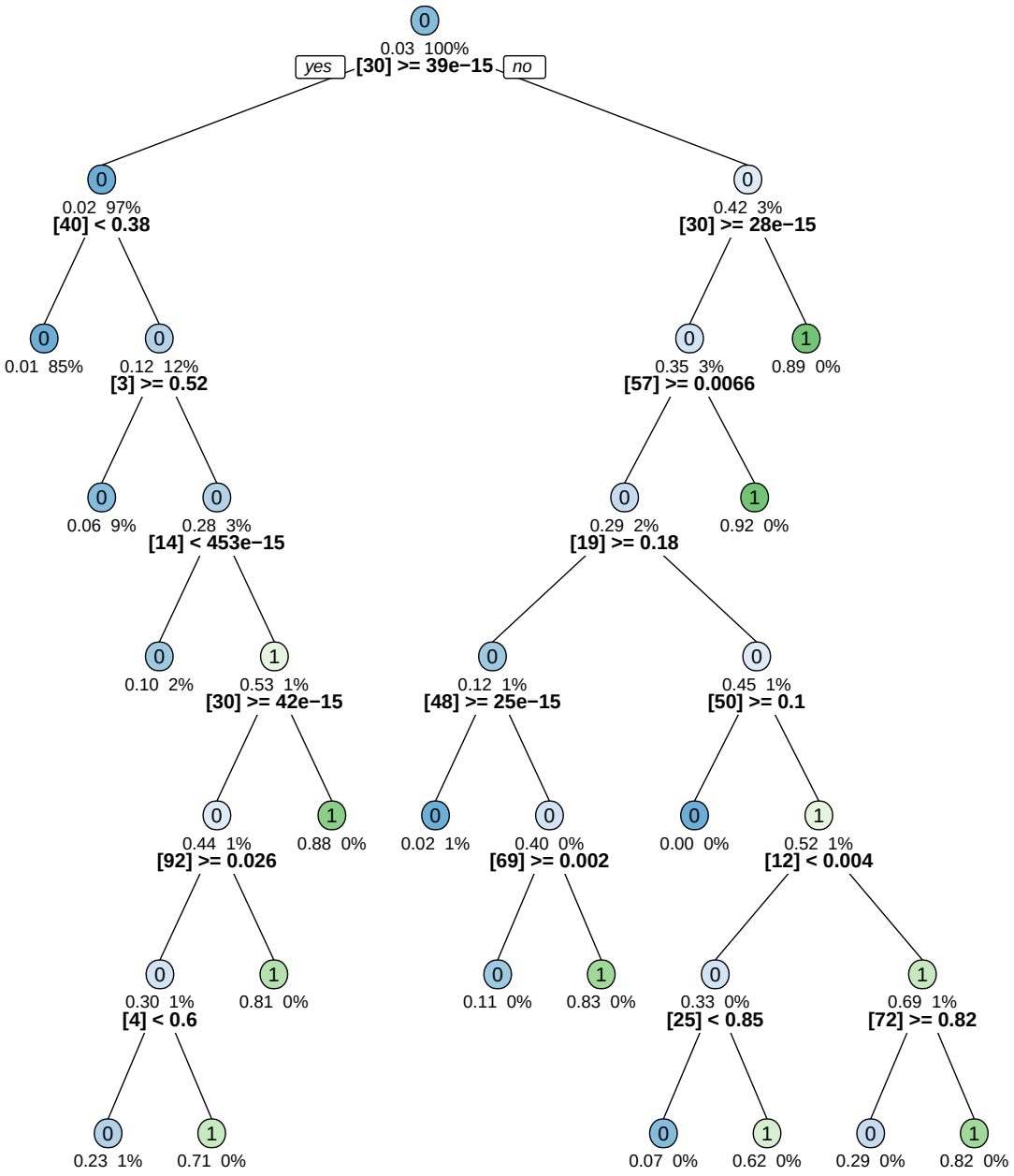

Figure 12: Schematic of Decision Tree for Original Database

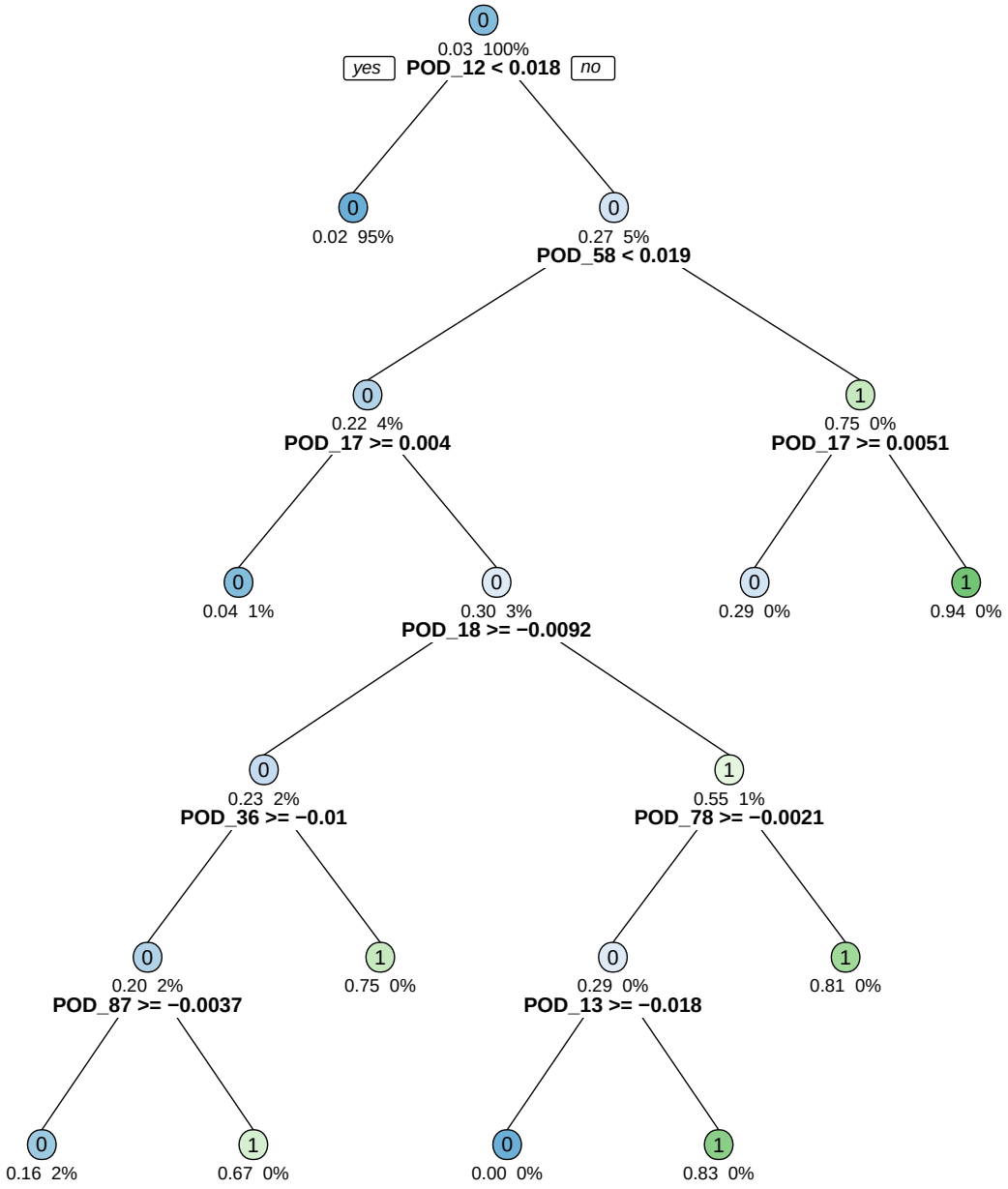

Figure 13: Schematic of Decision Tree for POD Results

Table 6: Confusion Matrix:SVM-POD

	Actual 0	Actual 1
Predicted 0	1312	36
Predicted 1	10	6

Table 7: Metrics of Predicting Models

	Accuracy	Specificity	Precision	Recall	F1
DT-Original	0.9626100	0.2916667	0.9744936	0.9870821	0.9807475
DT-POD	0.9677419	0.1666667	0.9740356	0.9931921	0.9835206
SVM-Original	0.9604106	0.1666667	0.9701937	0.9893617	0.9796840
SVM-POD	0.9662757	0.1428571	0.9732938	0.9924357	0.9827715

Seasons: EPS-Net Income - X43 - Net profit before tax/Paid-in capital: Pretax Income/Capital - X86 - Net Income to Total Assets - X23 - Per Share Net profit before tax (Yuan ¥): Pretax Income Per Share

For POD data frame, there is no interpretation of the most important features.

Figures 14 and 15 depict the 2D plots for combinations of the first four important features for original database as well as the POD results, respectively. One can see that the class labels shown below are more separable for both the original and POD databases, comparing to the corresponding plots displayed in section 3-3 and 3-4.

5- Conclusion

The classification of company bankruptcy database was performed by machine learning. The given database includes 6819 rows and 96 columns, indicating 95 features as well as one binary class label (‘0’ & ‘1’) to identify whether a company will survived or become bankrupted. There was a high level of imbalance among class labels, with only 3% portion of bankrupted companies. One feature with constant value for all rows was excluded before analysis. All the feature values were normalized to range (0,1) to ease the analysis. The cross correlation analysis was performed to find the highly correlated features. POD analysis (proper orthogonal decomposition) was also engaged to determine the independent directions with the maximum variance. Two machine learning algorithm were used for classification task namely, SVM (Support Vector Machine) and DT (Decision Tree), on both original database and its POD representation. Multiple prediction metrics were studied and compared for the performed calculations including Accuracy, Precision, F1 Score, Recall, and Specificity. Since the dataset was highly imbalanced, the Specificity was found to be the best metric to evaluate the performance of the predicting models. The values of all other metrics were very close to unity. The implementation of DT on the original dataset resulted in the best performance, which was about 30% for Specificity metric. The most important features were also identified by the DT model, and the visualization of class labels for the corresponding features was proposed as well. Future studies can be carried out for fine-tuning of the hyper parameters of the models to increase their performance.

Table 8: Most Important Features for Decision Tree Model

	1st	2nd	3rd	4th	5th	6th
Original	[30]	[90]	[19]	[43]	[86]	[23]
POD	POD_12	POD_17	POD_58	POD_56	POD_36	POD_15

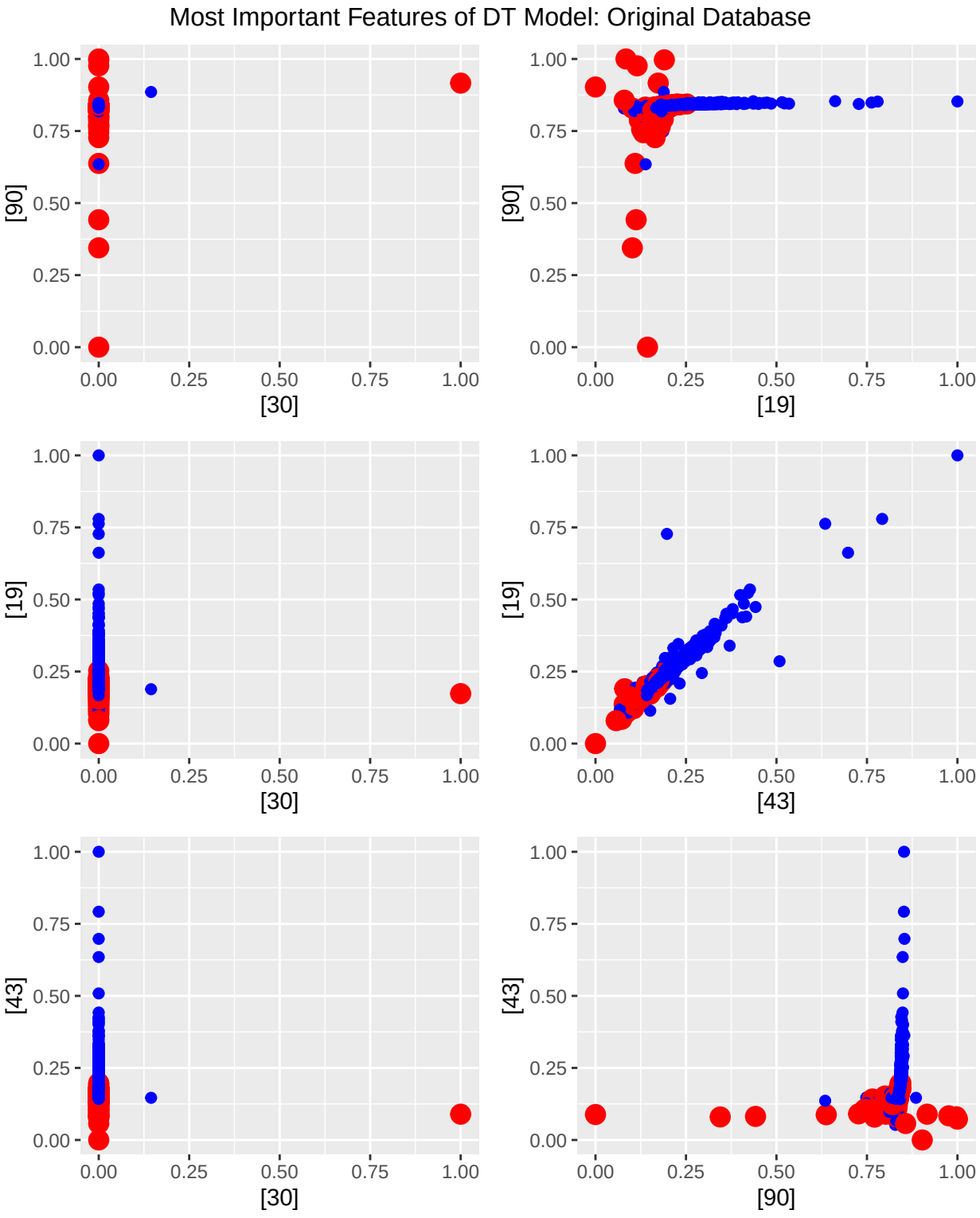

Figure 14: Most Important Features of DT Model: Original Database

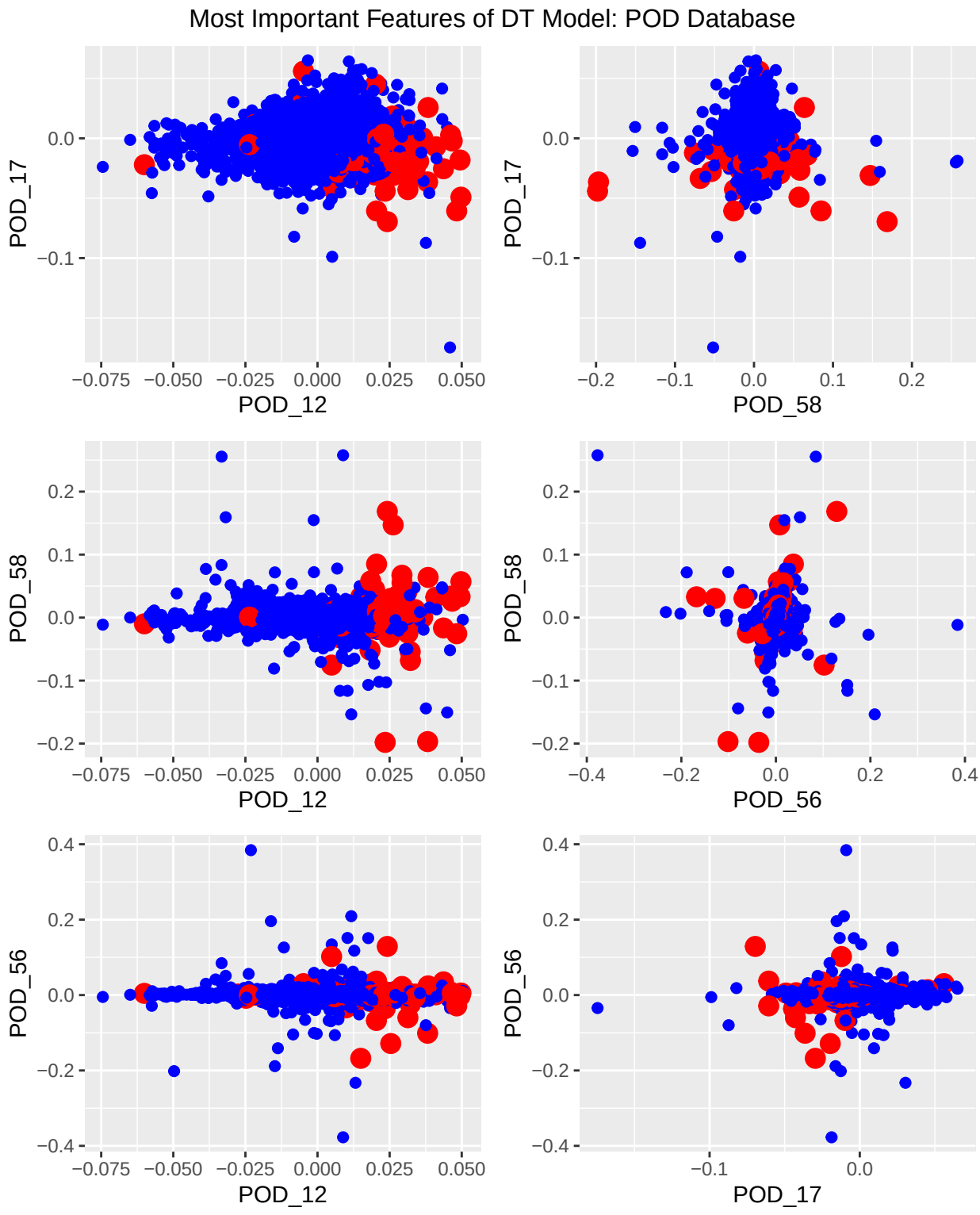

Figure 15: Most Important Features of DT Model: POD Database

6- References

[1] https://github.com/mkaramica/bankruptcy_prediction

[2] https://www.kaggle.com/datasets/fedesoriano/company-bankruptcy-prediction

[3] https://www.sciencedirect.com/science/article/abs/pii/S0377221716000412

[4] https://www.r-bloggers.com/2021/04/decision-trees-in-r/#:~:text=Decision%20Trees%20in%20R%2C%20Decision,mean

[5] https://www.geeksforgeeks.org/classifying-data-using-support-vector-machinessvms-in-r/

[6] https://en.wikipedia.org/wiki/Proper_orthogonal_decomposition

[7] https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/svd

[8] https://rafalab.github.io/dsbook/