Project Summary

Batch details	PGPDSE-FT Pune Jan21.
Team members	Akshay Ukarande Nandan Loya Jyoti Patil Vaishnavi Salunke Harsha Sapkale
Domain of Project	Finance and Risk Management
Proposed project title	Corporate bankruptcy prediction
Group Number	Group 4
Team Leader	Akshay Ukarande
Mentor Name	Mr. Vikash Chandra

Date:

Signature of the Mentor

Signature of the Team Leader

Table of Contents

Project Details

OVERVIEW

Bankruptcy is a legal process through which people or other entities who cannot repay debts to creditors may seek relief from some or all of their debts. Prediction of an enterprise bankruptcy is of great importance in economic decision making. Financial risk is a huge domain and corporate bankruptcy forms a significant part of it. In recent years artificial intelligence and machine learning methods have achieved promising results in corporate bankruptcy prediction settings. The dataset is about bankruptcy prediction of Polish companies.

The bankrupt companies were analyzed in the period 2000-2012, while the still operating companies were evaluated from 2007 to 2013. There is ample amount of data available in public domain about company financials and thus this large amount of data makes the area well suited for sophisticated data intensive computation methods.

Business problem statement (GOALS)

1. What would you achieve by this project?

   From a business point of view, it can be very helpful if one has an idea which corporate entities shall become Bankrupt. This classification will help in segregating corporate entities based on their fundamentals, which is considered essential for existence of the company. This will be helpful for both local as well as global economy.

2. How would this help the business or clients?

   This business objective, the Machine Learning model developed shall help to determine if the corporate entity falls in the category of Bankrupt or Non-Bankrupt by analyzing the various financial attributes and ratios of the entity. These attributes include assets, liabilities, capital etc. This shall help the creditors to take necessary action in timely manner. If an entity is classified as Bankrupt the creditors will be more cautious in financing the company and thus will be averted.

3. What is the further scope of the project?

   Taking a clue from this project automated web-based products can be deployed. This can also be used by National Governments or national banks in assessing the health of national economy. If many entities are being classified as Bankrupt then this indicates that there is need of serious and prompt action to be taken. Also considering Indian Economy this becomes all the way more important with passing of Insolvency and Bankruptcy Code 2016 in parliament.

4. Limitation of the project

   Bankruptcy of a company is not completely dependent on fundamentals or financials of company because sometimes bankruptcy may be result of external factors like sudden change in government policy or sudden shock to economy for example Lehman Brothers crisis in 2008. Also, sometimes financial fugitives become the reason of company going bankrupt. So, we can say that our model is not a generalized model rather a specific model based on specific attributes.

TOPIC SURVEY IN BRIEF (200-250 words)

1. Problem understanding

   The high social and economic costs because of corporate bankruptcies have attracted attention of researchers for better understanding of bankruptcy causes and eventually prediction of business distress. The aim of predicting financial distress is to develop a predictive model that combines various econometric parameters which allow foreseeing the financial condition of a firm.

2. Current solution to the problem

   Application of Statistical models Multidimensional analysis model and most recently linear model was used to predict bankruptcy.

3. Proposed solution to the problem

   Algorithm and ML driven classification of corporate entity.

4. Reference to the problem

   UCI, Kaggle, blogs on bankrupt

CRITICAL ASSESSMENT OF TOPIC SURVEY (50-100 words)

1. **Find the key area, gaps identified in the topic survey where the project can add value to the customers and business**.

   Present way of predicting bankruptcy is becoming comparatively slow and also involves lot of statistical analysis. This statistical analysis requires lot of domain expertise which makes it difficult to be used by layman person with some knowledge of finance. Also, as the volume of data is increasing and new ratios and attributes are being formulated it becomes a cumbersome task to constantly update the mathematical and statistical formulae.

2. **What key gaps are you trying to solve?**

   This project aims to solve the tedious task that a person has to go through every time a new financial measure is added or the volume and scope of data is increased. With the use of machine learning algorithm, the whole process of prediction becomes computationally efficient and effective at the same time. Machine learning algorithm are fast, accurate and less prone to human bias

METHODOLOGY TO BE FOLLOWED (Follow 1-2-3-4-5)

1) Business Understanding

Sometimes Bankruptcy of a company can have local as well as global impact. For instance, Lehman Brothers crisis of 2008 led to recession in the USA and later spreading to other countries. Also considering this Government of India has come up with Insolvency and Bankruptcy Code of India (2016) which signifies the intensity of problem created by Bankruptcy of corporate firms. The operational cost of resolving the debt-ridden bankrupt company becomes a tedious task and it becomes difficult for the Bad Bank or any other intermediary resolving the dispute, to make all the creditors come on the same page.

Thus, if creditors start making decisions based on results obtained through machine learning algorithms, they will be cautious while investing in a 'risky' corporate entity. Thus, predicting the bankruptcy of a company can have a positive impact on economy as the loses can be averted in advance and also create a stable and positive sentiment in the economy.

Variable identification:

There are many factors because of which a company can go bankrupt. One large problem may be at the root of a failing business. Some of the most common factors are:

1) **Outside business conditions** like an increase in competition, general costs of running a business, troubles inflicted by local hooligans etc.

2**) Inside business conditions** like a weak management, inappropriate location, client loss, trade credit problems etc.

3**) Financial problems** like loss of capital, inability to secure new capital when needed, high debt or difficulties with cash flow.

4**) Tax-related problems:** Often small business owners do not keep a keen eye on the tax structure and when they finally notice, the hefty amount crushes their resources.

5) **Accidents:** Even though insurance supposedly covers this, bureaucratic red tape can prevent the owner from getting his or her money.

Among the above-mentioned reasons this project takes into consideration only financial component like capital, assets, liabilities etc.

Some of the important attributes that we will be taking into consideration are:

Attribute or econometric evaluators information

ID	Description
X1	net profit / total assets
X2	total liabilities / total assets
X3	working capital / total assets
X4	current assets / short-term liabilities
X5	[(cash + short-term securities + receivables - short-term liabilities) / (operating expenses - depreciation)] * 365
X6	retained earnings / total assets
X7	EBIT / total assets
X8	book value of equity / total liabilities
X9	sales / total assets
X10	equity / total assets
X11	(Gross profit + extraordinary items + financial expenses) / total assets
X12	gross profit / short-term liabilities
X13	(Gross profit + depreciation) / sales
X14	(Gross profit + interest) / total assets
X15	(Total liabilities * 365) / (gross profit + depreciation)
X16	(Gross profit + depreciation) / total liabilities
X17	total assets / total liabilities
X18	gross profit / total assets
X19	gross profit / sales
X20	(inventory * 365) / sales
X21	sales (n) / sales (n-1)
X22	profit on operating activities / total assets
X23	net profit / sales
X24	gross profit (in 3 years) / total assets
X25	(Equity - share capital) / total assets
X26	(Net profit + depreciation) / total liabilities
X27	profit on operating activities / financial expenses
X28	working capital / fixed assets
X29	logarithm of total assets
X30	(Total liabilities - cash) / sales
X31	(Gross profit + interest) / sales
X32	(Current liabilities * 365) / cost of products sold

ID	Description
X33	operating expenses / short-term liabilities
X34	operating expenses / total liabilities
X35	profit on sales / total assets
X36	total sales / total assets
X37	(Current assets - inventories) / long-term liabilities
X38	constant capital / total assets
X39	profit on sales / sales
X40	(Current assets - inventory - receivables) / short-term liabilities
X41	total liabilities / ((profit on operating activities + depreciation) * (12/365))
X42	profit on operating activities / sales
X43	rotation receivables + inventory turnover in days
X44	(receivables * 365) / sales
X45	net profit / inventory
X46	(Current assets - inventory) / short-term liabilities
X47	(inventory * 365) / cost of products sold
X48	EBITDA (profit on operating activities - depreciation) / total assets
X49	EBITDA (profit on operating activities - depreciation) / sales
X50	current assets / total liabilities
X51	short-term liabilities / total assets
X52	(short-term liabilities * 365) / cost of products sold)
X53	equity / fixed assets
X54	constant capital / fixed assets
X55	working capital
X56	(Sales - cost of products sold) / sales
X57	(Current assets - inventory - short-term liabilities) / (sales - gross profit - depreciation)
X58	total costs /total sales
X59	long-term liabilities / equity
X60	sales / inventory
X61	sales / receivables
X62	(short-term liabilities *365) / sales
X63	sales / short-term liabilities
X64	sales / fixed assets

X1 net profit / total assets

Return on assets (ROA) is an indicator of how profitable a company is relative to its total assets. Return on assets, or ROA, measures how much money a company earns by putting its assets to use. In other words, ROA is an indicator of how efficient or profitable a company is relative to its assets or the resources it owns or controls.

Investors can use ROA to find stock opportunities because the ROA shows how efficient a company is at using its assets to generate profits.

A ROA that rises over time indicates the company is doing a good job of increasing its profits with each investment dollar it spends. A falling ROA indicates the company might have over-invested in assets that have failed to produce revenue growth, a sign the company may be in some trouble. ROA can also be used to make apples-to-apples comparisons across companies in the same sector or industry.

X2 total liabilities / total assets

TD/TA= (Short-Term Debt + Long-Term Debt) / Total Assets

A ratio greater than 1 show that a considerable portion of the assets is funded by debt. In other words, the company has more liabilities than assets. A high ratio also indicates that a company may be putting itself at risk of defaulting on its loans if interest rates were to rise suddenly.
A ratio below 1, meanwhile, indicates that a greater portion of a company's assets is funded by equity. Reciprocal – attr17

X6 Retained Earnings/Total Assets (RE/TA)

This ratio measures the amount of reinvested earnings or losses, which reflects the extent of the company's leverage. Companies with low RE/TA are financing capital expenditure through borrowings rather than through retained earnings. Companies with high RE/TA suggest a history of profitability and the ability to stand up to a bad year of losses.

X7 Earnings Before Interest and Tax/Total Assets (EBIT/TA)

The ratio Earnings Before Interest and Tax/Total Assets (EBIT/TA) is a version of return on assets (ROA), an effective way of assessing a firm's ability to squeeze profits from its assets before deducting factors like interest and tax.

X19 gross profit / sales

Gross Profit=Revenue−Cost of Goods Sold
Gross profit does not include fixed costs (that is, costs that must be paid regardless of the level of output). Fixed costs include rent, advertising, insurance, salaries for employees not directly involved in the production and office supplies.

X22 profit on operating activities / total assets

Return on Operating Assets = Net Income / Operating Assets

Since the ROOA equation uses net income, there are several factors that could contribute to a change in this ratio. Everything from cost of goods sold to employee salaries and utilities expenses affect net income, making ROOA a fairly sensitive measurement.

X23 net profit / sales

Net profit is the amount of money that a company has after all its expenses are paid. You can think of net profit like your paycheck: It's the money left after all taxes and benefits are subtracted. Net Profit = Total Revenue – Total Expenses

X29 logarithm of total assets

Firm size is measured using the logarithm of total assets.

X55 working capital

Working Capital = Current Assets – Current Liabilities
The working capital formula tells us the short-term liquid assets available after short-term liabilities have been paid off.

X48 EBITDA

EBITDA, which stands for Earnings Before Interest, Taxes, Depreciation, and Amortization, is a financial calculation that measures a company's profitability before deductions that are often considered irrelevant in the decision-making process. In other words, it's the net income of a company with certain expenses like amortization, depreciation, taxes, and interest added back into the total.

Why is EBITDA used in valuation? It's a profitability calculation that measures how profitable a company is before paying interest to creditors, taxes to the government, and taking paper expenses like depreciation and amortization. This is not a financial ratio. Instead, it's a calculation of profitability that is measured in dollars rather than percentages.
Like all profitability measurements, higher numbers are always preferred over lower numbers because higher numbers indicate the company is more profitable.

X60 sales / inventory

Inventory or stock refers to the goods and materials that a business holds for the ultimate goal of resale, production or utilization. Inventory management is a discipline primarily about specifying the shape and placement of stocked goods

X63 sales / short-term liabilities
Short-term debt, also called current liabilities, is a firm's financial obligations that are expected to be paid off within a year. Common examples of short-term debt include accounts payable, current taxes due for payment, short-term loans, salaries, and wages due to employees, and lease payments.

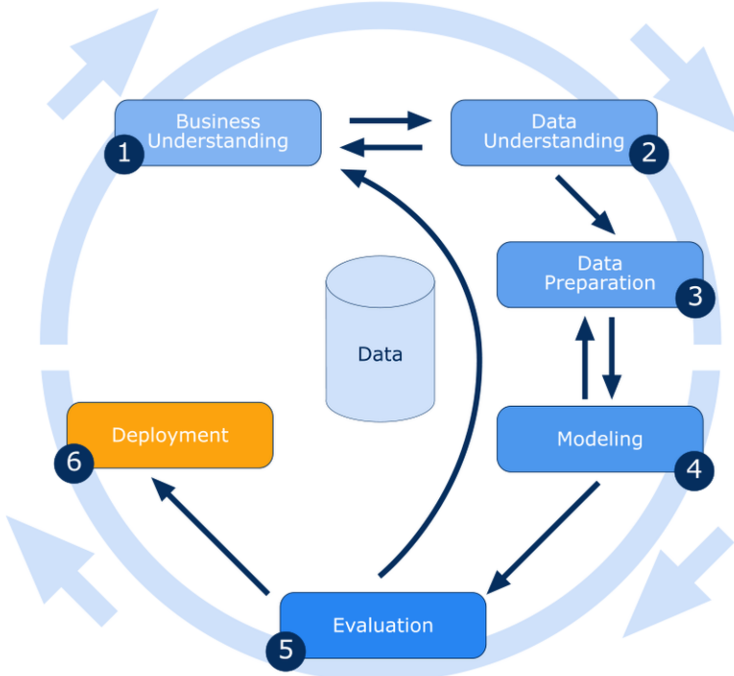
X59 long-term liabilities / equity

The ratio is calculated by taking the company's long-term debt and dividing it by the book value of common equity. The greater a company's leverage, the higher the ratio.

Generally, companies with higher Debt to Equity ratios are thought to be riskier. This is because a higher proportion of assets must go towards servicing interest payments on debt, which are fixed. If income falls it can quickly fall below the minimum level required to service these interest payments leaving equity investors with nothing.

The long-term portion of a bond payable is reported as a long-term liability. Because a bond typically covers many years, the majority of a bond payable is long term. The present value of a lease payment that extends past one year is a long-term liability. Deferred tax liabilities typically extend to future tax years, in which case they are considered a long-term liability. Mortgages, car payments, or other loans for machinery, equipment, or land are long term, except for the payments to be made in the coming 12 months.

X46 (current assets - inventory) / short-term liabilities: - Current assets are all the assets of a company that are expected to be sold or used as a result of standard business operations over the next year. Current assets include cash, cash equivalents, accounts receivable, stock inventory, marketable securities, pre-paid liabilities, and other liquid assets.



2) Data Understanding

The dataset that we have considered for addressing the bankruptcy prediction problem is the Polish bankruptcy data, hosted by the University of California Irvine (UCI) Machine Learning Repository—a huge repository of freely accessible datasets for research and learning purposes intended for the Machine Learning/Data Science community. The dataset is about bankruptcy prediction of Polish companies. The data was collected from Bankruptcy Prediction: Mining the Polish Bankruptcy Data 6 Emerging Markets Information Service (EMIS), which is a database containing information on emerging markets around the world.

Five data five classification cases were distinguished, that depends on the forecasting period:

1. **1st year:** The data contains financial rates from 1st year of the forecasting period and corresponding class label that indicates bankruptcy status after 5 years.
2. **2nd year:** The data contains financial rates from 2nd year of the forecasting period and corresponding class label that indicates bankruptcy status after 4 years.
3. **3rd year:** The data contains financial rates from 3rd year of the forecasting period and corresponding class label that indicates bankruptcy status after 3 years.
4. **4th year:** The data contains financial rates from 4th year of the forecasting period and corresponding class label that indicates bankruptcy status after 2 years.
5. **5th year:** The data contains financial rates from 5th year of the forecasting period and corresponding class label that indicates bankruptcy status after 1 years.

3) Data Preparation

Files were in **". arff"** format, those files were converted to **".csv"** format using WEKA tool. Five datasets were obtained for 5 years of data separately.

- Csv Files were upload as df_1, df_2, df_3, df_4, df_5.

a) Null Value Assessment

Total 66 columns are present in each dataset. Missing Values were found in the data. Some in the form of **"?"**, which were replaced as **nan values** (in order to perform imputation later).

Datatype of attributes were also seen; it was found that some attributes had 'object' datatype which were converted to 'float64' using function NumPy library.

Parameters	**1st year**	**2nd year**	**3rd year**	**4th year**	**5th year**
Number of Rows	7027	10173	10503	9792	5910
Bankrupt Rows	271	400	495	515	410
Non-Bankrupt instances	6758	9773	10008	9227	5500
Missing Data Percentage	54.54 %	59.81 %	53.48 %	51.29 %	48.71 %
Percentage of minority class samples	3.85 %	3.93 %	4.71 %	5.25 %	6.93 %

Table1: Table showing Percentage null values and Data Imbalance

b) Visualization of Null values

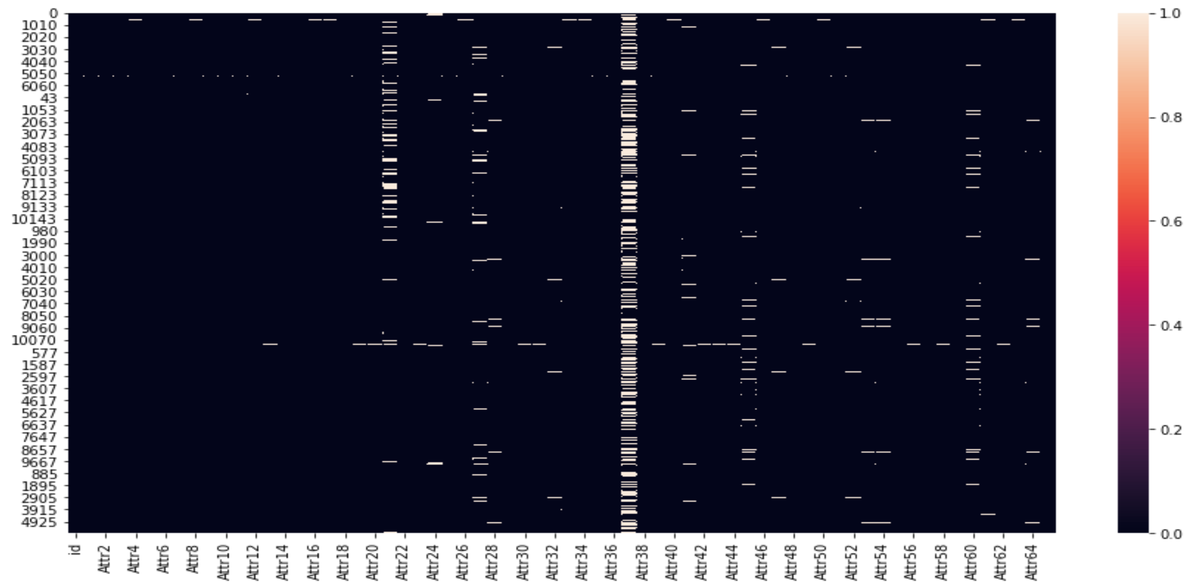


Figure 1: Visualization of Null values

As the data loss in most of the datasets is over 50%, it is now clear that we cannot simply drop the rows with missing values, as it leads to severe loss in the representativeness of data.We observe that most of the null values are present in **Attr37 (43.73%) and Attr21 (13.48%)** which was above 1000 null values. These **columns** were dropped as percentage of null values was high.

c) Imputation of Null values

Imputation is the process of replacing missing data with substituted values and it preserves all the cases by replacing missing data with an estimated value, based on other available information. We tried following ways of Imputing the Null values: -

1) Mean imputation using Simple Imputer.
2) k-Nearest Neighbors(kNN) Regressor imputation using Iterative Imputer.
3) Linear Regressor imputation using Iterative Imputer.
4) Multivariate Imputation by Chained Equations (MICE) Imputer using Impyute library.
5) Expectation Maximization Imputer (EM) using Impyute library.

1) **Mean Imputation:**

Mean imputation technique is the process of replacing any missing value in the data with the mean of that variable in context. In our dataset, we replaced a missing value of a feature, with the mean of the other non-missing values of that feature.

2) **kNN Regressor Imputation:**

   The k-nearest neighbors' algorithm or k-NN, is a **non-parametric** method used for classification and regression. It can also be used as a data imputation technique k-NN imputation replaces NaNs in Data with the corresponding value from the nearest-neighbor row or column depending upon the requirement. The nearest-neighbor row or column is the closest row or column by Euclidean distance. If the corresponding value from the nearest-neighbor is also NaN, the next nearest neighbor is used.

3) **Linear Regressor Imputation:**

   Linear Regressor Imputation works on the basis of linear regression algorithm, which in turn uses extrapolation and best fit line as its base to predict the values. In this way the NaN values are filled with predicted values from the linear regressor model.

4) **Multivariate Imputation by Chained Equations (MICE) Imputer:**

   Multiple imputation using chained equations or MICE is an imputation technique that uses multiple imputations as opposed to single imputation. MICE is regarded as a fully conditional specification or sequential regression multiple imputations. It has become one of the principal methods of addressing missing data. Creating multiple imputations, as opposed to single imputations, accounts for the statistical uncertainty in the imputations. In addition, the chained equations approach is very flexible and can handle variables of varying types (for example., continuous or binary**). MICE is beneficial when the missing data is large.** Because multiple imputation involves creating multiple predictions for each missing value, the analyses of multiply imputed data take into account the uncertainty in the imputations and yield accurate standard errors. each variable can be modeled according to its distribution, with, for example, binary variables modeled using logistic regression and continuous variables modeled using linear regression.

5) **Expectation Maximization Algorithm**:
EM Imputation is the process of imputing missing values using Expectation-Maximization. Missing values of quantitative variables are replaced by their expected value computed using the Expectation-Maximization (EM) algorithm. In practice, a Multivariate Gaussian distribution is assumed.

Statistical Test were performed on each Data frame before and after imputation to check the central tendency of data. This was done using non-parametric 2 sample test i.e., **Mannwhithneyu** test.

- P-value<= alpha: reject H0, Different Sample Distributions.
- P-value> alpha: Fail to reject H0, Sample Distributions are equal.
  Here we considered **alpha as 5%** and **confidence interval 95%**

Imputation	Central tendency Changed Attributes
Simple Imputer -Mean	Dataset1 – Attr45. Dataset2 – Attr24. Dataset3 – Attr24 Dataset4 – Attr24 Dataset5 – Attr24
k-Neighbors Regressor	Dataset1 – Attr27. Dataset2 – Attr24, Attr27, Attr28, Attr41, Attr45, Attr53, Attr54, Attr60. Dataset3 – Attr24, Attr27, Attr28, Attr41, Attr45, Attr53, Attr54, Attr60. Dataset4 - Attr24, Attr27, Attr28, Attr41, Attr45, Attr53, Attr54, Attr60. Dataset5 - Attr24, Attr27, Attr28, Attr45, Attr54, Attr60.
Linear Regression	Dataset1 - Attr27, Attr45. Dataset2 - Attr24, Attr27, Attr45, Attr60. Dataset3 – Attr24, Attr27, Attr45, Attr53, Attr60, Attr64. Dataset4 – Attr24, Attr27, Attr41, Attr45, Attr60. Dataset5 – Attr24, Attr27.
MICE (Multivariate Imputation by Chained Equations) Imputation	Dataset1 - Attr27, Attr45. Dataset2 - Attr24, Attr27, Attr45, Attr60. Dataset3 – Attr24, Attr27, Attr45, Attr60. Dataset4 – Attr24, Attr27, Attr41, Attr45, Attr60. Dataset5 – Attr24, Attr27.
EM (Expectation Maximization) Imputation	Dataset1 – Attr24, Attr27, Attr45, Attr60. Dataset2 – Attr24, Attr27, Attr28, Attr41, Attr45, Attr53, Attr54, Attr60, Attr64. Dataset3 – Attr24, Attr27, Attr28, Attr41, Attr45, Attr53, Attr54, Attr60, Attr64. Dataset4 – Attr24, Attr27, Attr28, Attr41, Attr45, Attr53, Attr54, Attr60, Attr64. Dataset5 – Attr24, Attr27, Attr28, Attr45, Attr53, Attr54, Attr60.

It was observed that the mean imputed data showed minimum variation in central tendency from actual data. Linear Regressor imputed data failed on many instances. Also, kNN, MICE and EM failed on some instances, but showed better results than Linear Regressor Imputation.

Further for model building we used **Mean Imputed Data.**

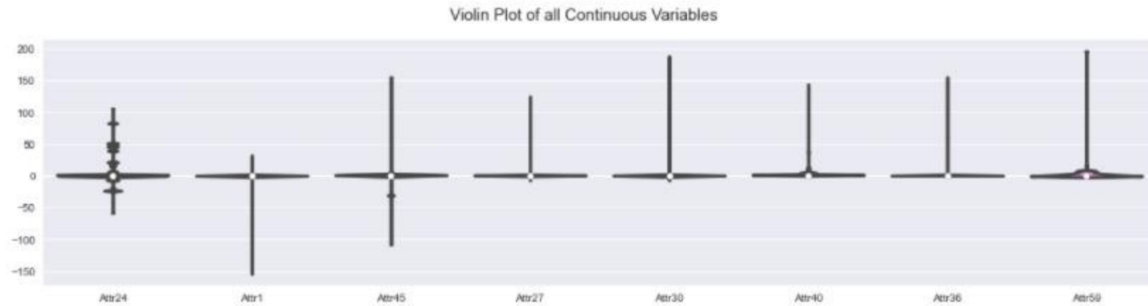
d) Data Imbalance

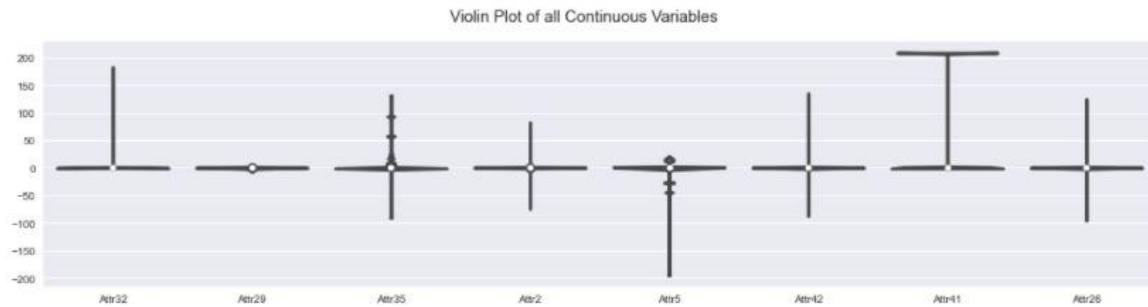
From Table 1 we observe that there is high Data imbalance in each dataset. For a better classification model, we have to handle this imbalance. This is because the machine learning models for classification are designed in such a way that, there is minimum error and maximum accuracy. Thus, it falsely classifies the lesser represented class as higher represented class. This increase the accuracy but this accuracy can be termed as misleading accuracy.

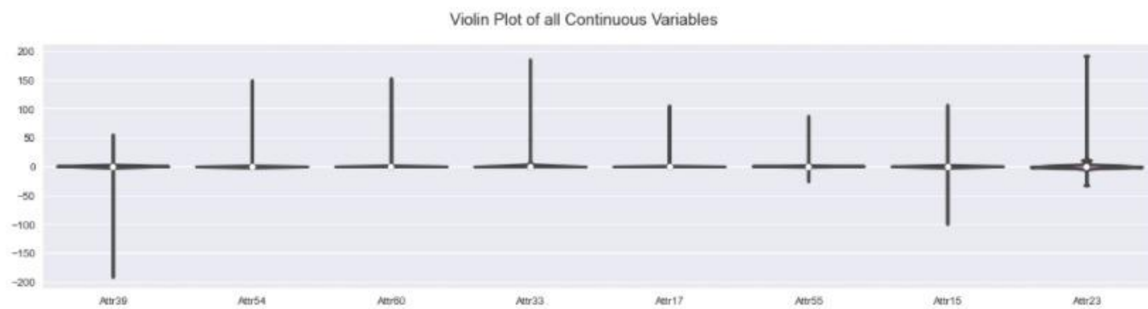
Data Imbalance can be treated with **Oversampling** and/or **Under sampling.** In data analysis, Oversampling and Undersampling are opposite and roughly equivalent techniques of dealing with Data Imbalance, where they adjust the class distribution of a data set. Oversampling is achieved by increasing the class distribution of the minority class label whereas Undersampling is achieved by decreasing the class distribution of the majority class label. Undersampling sometimes leads to data and information loss which affects the model building. So, we have used two methods to handle this imbalance

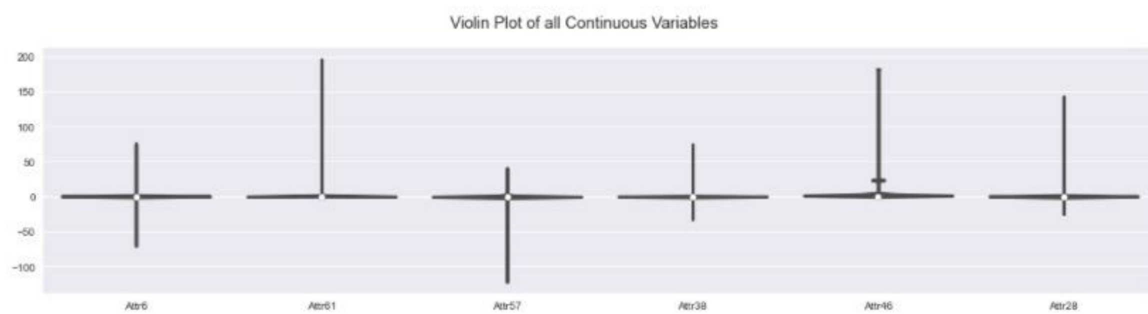
1) **Synthetic Minority Oversampling Technique or SMOTE**
2) **SMOTE TOMEK**

e) **Outliers:**

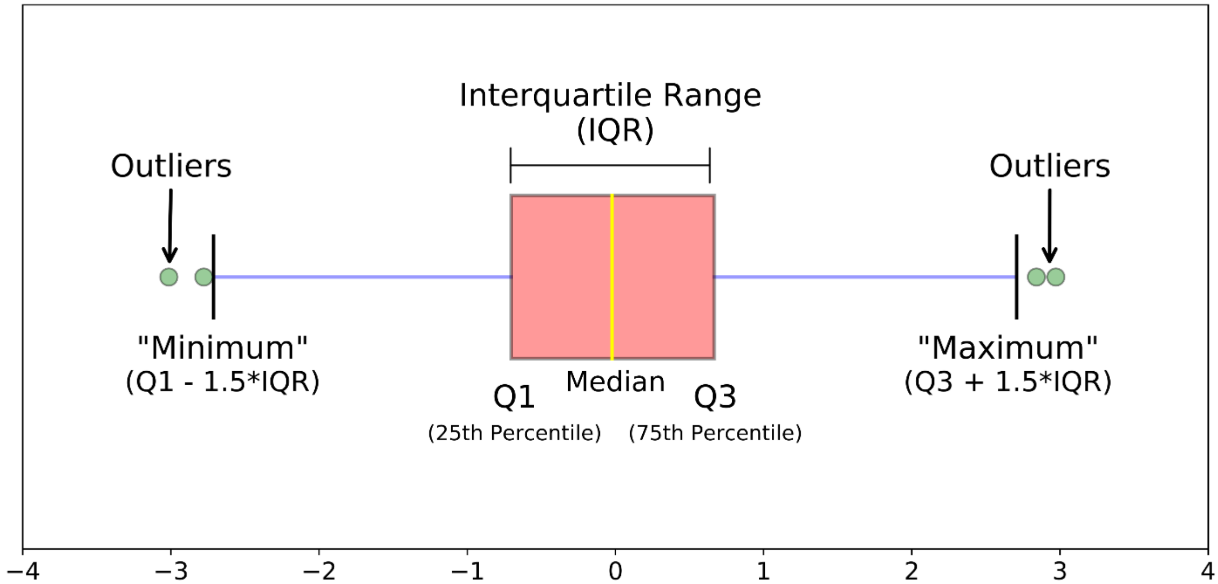








When building a model, it is important to check if a dataset contain extreme values (Outliers) that are outside the range of what is expected and unlike the other data. Outliers were seen visually using violin plot. As we can see there are too many outliers present in the data. As observed, about 97% data points are present above the upper whisker and lower whisker. It was seen, that only 1 -2% data points were present in the inter quartile range, which is 50% of the total data. Attr29 (logarithm of total assets) have less outliers as compared to other features. Model skill in general can be improved by understanding and even removing these outlier values.



These Outliers will be handled further while model building with the use of Power transformation. Due to presence of outliers, we also see that there is high skewness and variance in Attributes.

f) Multicollinearity:

Multicollinearity is a **phenomenon in which one predictor variable in a multiple regression model can be linearly predicted** from the others with a substantial degree of accuracy. As the attributes or econometric features are basically ratios, there is higher chance of multicollinearity thus, hampering the model building.

Multicollinearity causes following problems:

1)The coefficient estimates can swing wildly based on which other independent variables are in the model.

2)The coefficients become very sensitive to small changes in the model.

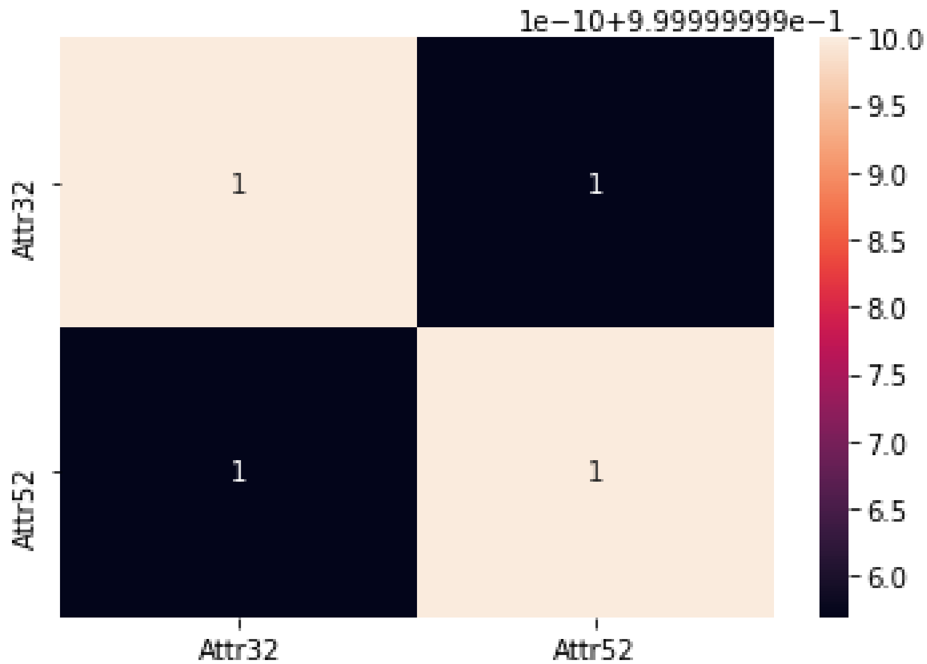


Figure 2: Heatmap for visualization of Multicollinearity between Attr32 and Attr52

Figure 2 shows that,

Attr 32:- (current liabilities * 365) / cost of products sold

Attr 52:- (short-term liabilities * 365) / cost of products sold)

This suggests that short-term liabilities and current liabilities are completely dependent on each other. For model building keeping these both features can hamper the accuracy of the model.

Such multicollinearity is observed in various other attributes too. This is visualized in the heatmap below.

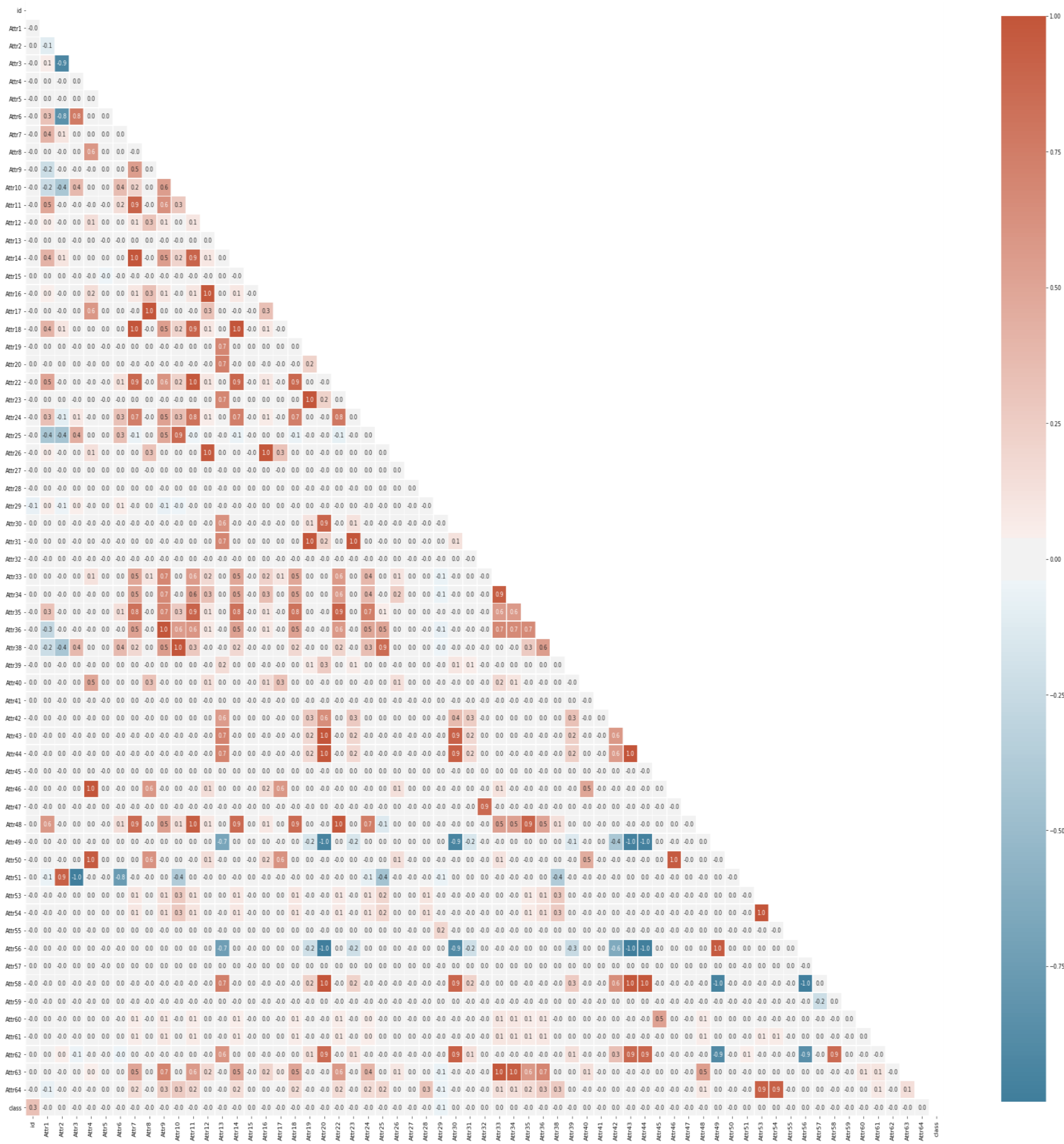


Figure 3: Heatmap for Multicollinearity of all attributes on combined Dataset.

4)Modelling:

Various Classification Models were Built for

- Without Smote Full Mean Imputed Data
- With Smote Full Mean Imputed Data
- With Smote Tomek Full Mean Imputed Data
- Smote Tomek with Principle Component Analysis with Scaling
- Smote Tomek with Principle Component Analysis without Scaling
- Smote with 32 features
- Smote Tomek with 32 features

Later hyperparameter Tuning was done on models with less overfitting and good accuracy.

5)Evaluation:

To test the Model results after model building we used various evaluation metrics from sklearn library.

- **Accuracy Score:** it takes in the true labels and the predicted labels as arguments and returns the accuracy as a float value.

- **Confusion Matrix:** By definition a confusion matrix C is such that C i, j is equal to the number of observations known to be in group i and predicted to be in group j. Thus in binary classification, the count of true negatives is C 0, 0, false negatives is C 1, 0, true positives is C 1, 1 and false positives is C 0, 1.

- **Classification Report:** A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False.

A) Without Smote Full Mean Imputed Data:

```
#Logistic Regression -Base Model Without SMOTE

Average Accuracy Score Train:  0.9491163922983983
Accuracy Score Test:  0.9510059898633082
Confusion matrix :
 [[12379    28]
 [  610     5]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.95      1.00      0.97     12407
         1.0       0.15      0.01      0.02       615

    accuracy                           0.95     13022
   macro avg       0.55      0.50      0.50     13022
weighted avg       0.92      0.95      0.93     13022
```

```
#random Forest -Base Model Without SMOTE

Average Accuracy Score Train:  0.9646184368846458
Accuracy Score Test:  0.9662878206112733
Confusion matrix :
 [[12373    34]
 [  405   210]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.97      1.00      0.98     12407
         1.0       0.86      0.34      0.49       615

    accuracy                           0.97     13022
   macro avg       0.91      0.67      0.74     13022
weighted avg       0.96      0.97      0.96     13022
```

```
#Decision -Base Model Without SMOTE

Average Accuracy Score Train:  0.9525392366971897
Accuracy Score Test:  0.9513131623406543
Confusion matrix :
 [[12064   343]
 [  291   324]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.98      0.97      0.97     12407
         1.0       0.49      0.53      0.51       615

    accuracy                           0.95     13022
   macro avg       0.73      0.75      0.74     13022
weighted avg       0.95      0.95      0.95     13022
```

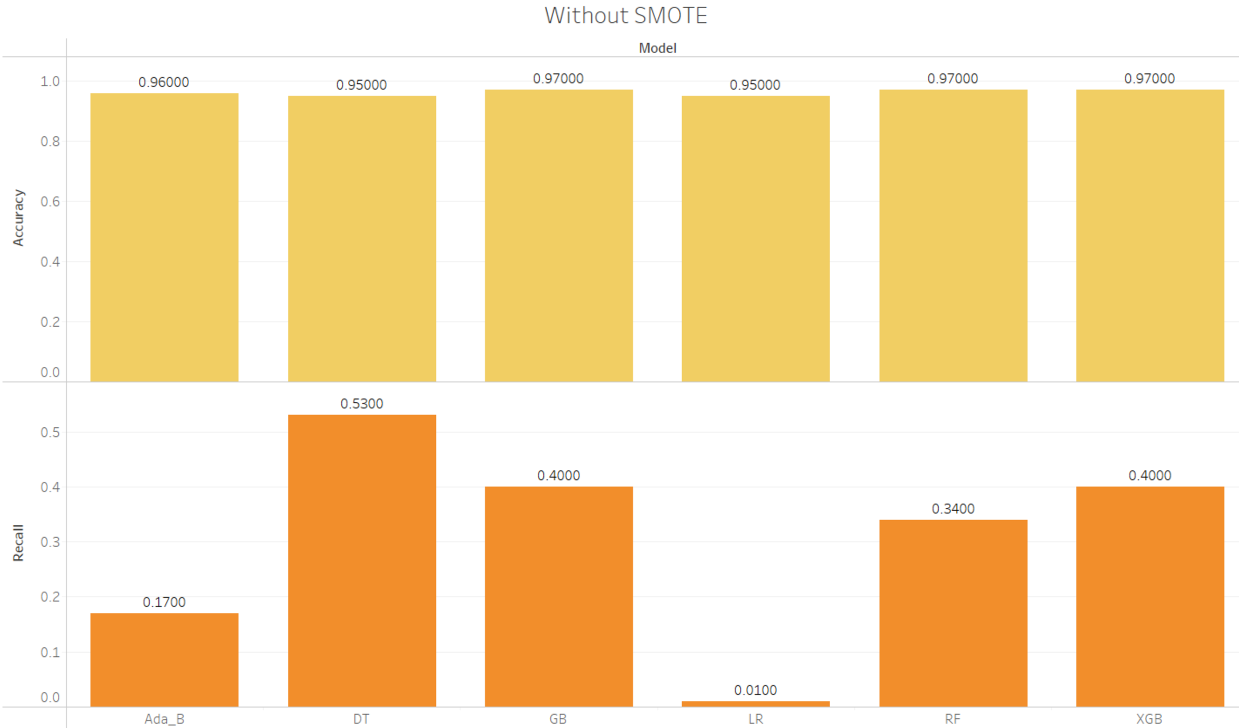
```
#XGBoost -Base Model Without SMOTE

Average Accuracy Score Train:  0.9697858392455911
Accuracy Score Test:  0.9710489940101367
Confusion matrix :
 [[12397    10]
 [  367   248]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.97      1.00      0.99     12407
         1.0       0.96      0.40      0.57       615

    accuracy                           0.97     13022
   macro avg       0.97      0.70      0.78     13022
weighted avg       0.97      0.97      0.97     13022
```

Model	Accuracy	Recall	Precision	f1-score
LR	0.95	0.01	0.12	0.02
DT	0.95	0.53	0.49	0.51
RF	0.97	0.34	0.86	0.49
GB	0.97	0.4	0.87	0.54
Ada_B	0.96	0.17	0.59	0.26
XGB	0.97	0.4	0.96	0.57



Sum of Accuracy and sum of Recall for each Model.

We observe that even if accuracy is high the recall and precision is very low meaning there are high number of false negatives. This is because models most of the data into heavier class.

B) With Smote Full Mean Imputed Data

```
#Logistic regression  -Base Model With SMOTE

Average Accuracy Score Train:  0.6092080525260193
Accuracy Score Test:  0.6058332324821494
Confusion matrix :
 [[9550 2806]
 [6965 5468]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.58      0.77      0.66     12356
         1.0       0.66      0.44      0.53     12433

    accuracy                           0.61     24789
   macro avg       0.62      0.61      0.59     24789
weighted avg       0.62      0.61      0.59     24789
```

```
#Random Forest  -Base Model With SMOTE

Average Accuracy Score Train:  0.9762444261534309
Accuracy Score Test:  0.9787002299406995
Confusion matrix :
 [[11984   372]
 [  156 12277]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.99      0.97      0.98     12356
         1.0       0.97      0.99      0.98     12433

    accuracy                           0.98     24789
   macro avg       0.98      0.98      0.98     24789
weighted avg       0.98      0.98      0.98     24789
```

```
#Decision Tree  -Base Model With SMOTE

Average Accuracy Score Train:  0.9249295283800281
Accuracy Score Test:  0.9291217878897898
Confusion matrix :
 [[11354  1002]
 [  755 11678]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.94      0.92      0.93     12356
         1.0       0.92      0.94      0.93     12433

    accuracy                           0.93     24789
   macro avg       0.93      0.93      0.93     24789
weighted avg       0.93      0.93      0.93     24789
```

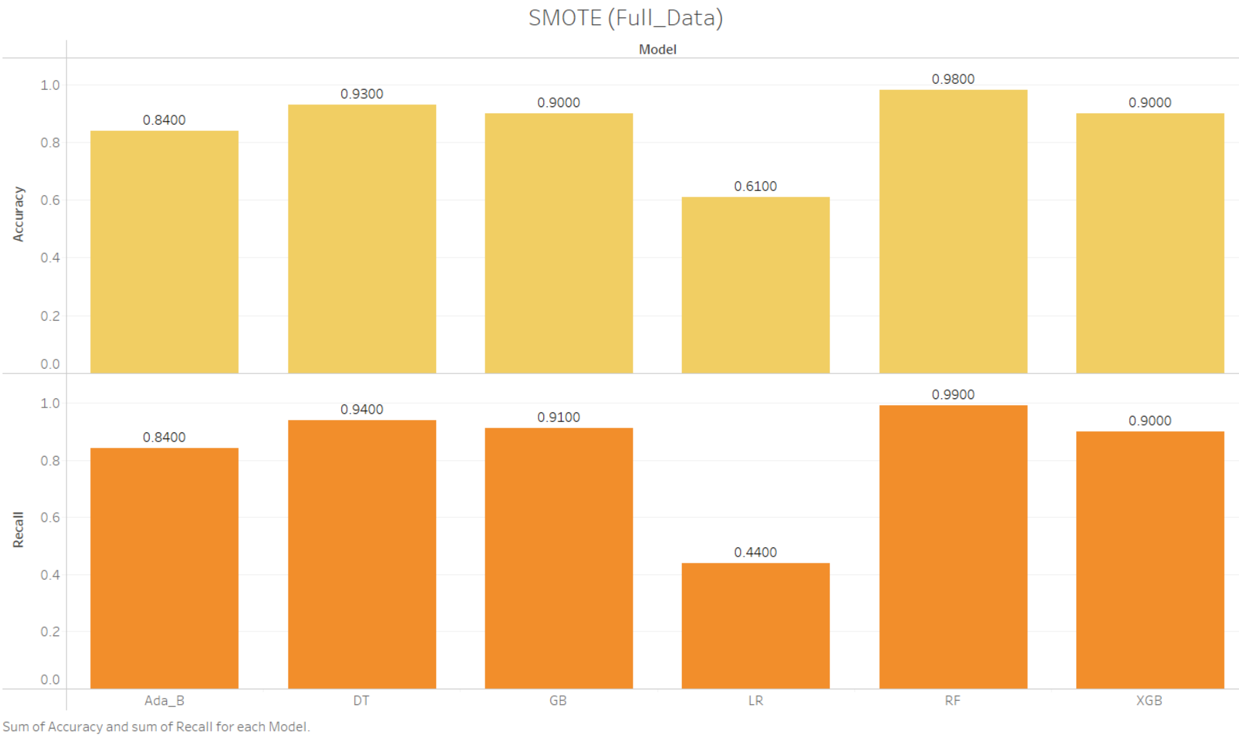
```
#XGBoost  -Base Model With SMOTE

Average Accuracy Score Train:  0.9065163290043537
Accuracy Score Test:  0.903182863366816
Confusion matrix :
 [[11183  1173]
 [ 1227 11206]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.90      0.91      0.90     12356
         1.0       0.91      0.90      0.90     12433

    accuracy                           0.90     24789
   macro avg       0.90      0.90      0.90     24789
weighted avg       0.90      0.90      0.90     24789
```

Model	Accuracy	Recall	Precision	f1-score
LR	0.61	0.44	0.66	0.53
DT	0.93	0.94	0.92	0.93
RF	0.98	0.99	0.97	0.98
GB	0.9	0.91	0.9	0.9
Ada_B	0.84	0.84	0.84	0.84
XGB	0.9	0.9	0.91	0.9



Accuracy , Recall and Precision all have increased with the use of Smote.

C) Smote Tomek with Full Mean data:

```
Average Accuracy Score Train:  0.5918597672875581
Accuracy Score Test:  0.6018158236057068
Confusion matrix :
 [[ 2616  9659]
 [  165 12232]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.94      0.21      0.35     12275
         1.0       0.56      0.99      0.71     12397

    accuracy                           0.60     24672
   macro avg       0.75      0.60      0.53     24672
weighted avg       0.75      0.60      0.53     24672
```

Logistic Regression

```
Average Accuracy Score Train:  0.9724059222627718
Accuracy Score Test:  0.975622595667139
Confusion matrix :
 [[11872   358]
 [  244 12221]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.98      0.97      0.98     12230
         1.0       0.97      0.98      0.98     12465

    accuracy                           0.98     24695
   macro avg       0.98      0.98      0.98     24695
weighted avg       0.98      0.98      0.98     24695
```

Decision Tree

```
Average Accuracy Score Train:  0.9881423611111112
Accuracy Score Test:  0.9901158551405655
Confusion matrix :
 [[12048   207]
 [   37 12394]]
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      0.98      0.99     12255
         1.0       0.98      1.00      0.99     12431

    accuracy                           0.99     24686
   macro avg       0.99      0.99      0.99     24686
weighted avg       0.99      0.99      0.99     24686
```

Random Forest

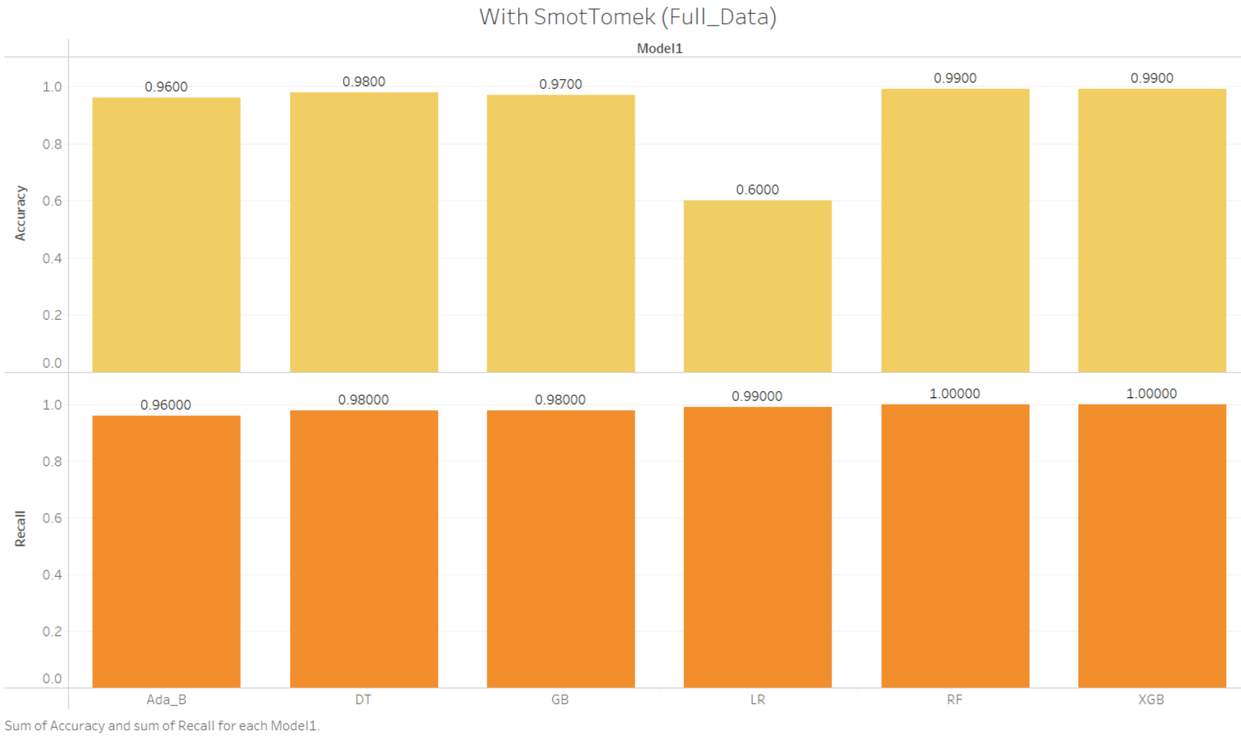
```
Average Accuracy Score Train:  0.9936095690278878
Accuracy Score Test:  0.9938816855753647
Confusion matrix :
 [[12107   136]
 [   15 12422]]
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      0.99      0.99     12243
         1.0       0.99      1.00      0.99     12437

    accuracy                           0.99     24680
   macro avg       0.99      0.99      0.99     24680
weighted avg       0.99      0.99      0.99     24680
```

XGBoost

Model	Accuracy	Recall	Precision	f1-score
LR	0.6	0.99	0.56	0.71
DT	0.98	0.98	0.97	0.98
RF	0.99	1	0.98	0.99
GB	0.97	0.98	0.96	0.97
Ada_Boost	0.96	0.96	0.95	0.96
XGB	0.99	1	0.99	0.99



With Smote Tomek the accuracy for most of the models is increased. We also see that for XgBoost the accuracy is 0.99 also the recall is high which is a very positive sign. Logistic Regression models is not performing well.

D) Smote Tomek with Principle Component Analysis with Scaling:

N_Components=30

```
accuracy score for train data:  1.0
accuracy score for test data:  0.6759360897670448
confusion_matrix:
[[7253  946]
 [4368 3831]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.62      0.88      0.73      8199
         1.0       0.80      0.47      0.59      8199

    accuracy                           0.68     16398
   macro avg       0.71      0.68      0.66     16398
weighted avg       0.71      0.68      0.66     16398
```

Decision Tree

```
accuracy score for train data:  1.0
accuracy score for test data:  0.6949018172947921
confusion_matrix:
[[7895  304]
 [4699 3500]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.63      0.96      0.76      8199
         1.0       0.92      0.43      0.58      8199

    accuracy                           0.69     16398
   macro avg       0.77      0.69      0.67     16398
weighted avg       0.77      0.69      0.67     16398
```

Random Forest

```
accuracy score for train data:  0.6642717973697029
accuracy score for test data:  0.646176362971094
confusion_matrix:
[[5818 2381]
 [3421 4778]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.63      0.71      0.67      8199
         1.0       0.67      0.58      0.62      8199

    accuracy                           0.65     16398
   macro avg       0.65      0.65      0.64     16398
weighted avg       0.65      0.65      0.64     16398
```

Logistic Regression

```
accuracy score for train data:  0.9741384559181685
accuracy score for test data:  0.7561287961946579
confusion_matrix:
[[7557  642]
 [3357 4842]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.69      0.92      0.79      8199
         1.0       0.88      0.59      0.71      8199

    accuracy                           0.76     16398
   macro avg       0.79      0.76      0.75     16398
weighted avg       0.79      0.76      0.75     16398
```

XGB Model

Model Name	Accuracy	Recall	Precision	F1 Score
DT	0.67	0.45	0.8	0.58
RF	0.69	0.42	0.92	0.58
LR	0.65	0.58	0.67	0.62
GB	0.74	0.72	0.75	0.74
AB	0.72	0.7	0.72	0.71
XG	0.76	0.59	0.88	0.71

E) Smote Tomek with Principle Component Analysis without Scaling:

N_components=10

```
accuracy score for train data:  1.0
accuracy score for test data:  0.7410129709697344
confusion_matrix:
[[7212  883]
 [3310 4785]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.69      0.89      0.77      8095
         1.0       0.84      0.59      0.70      8095

    accuracy                           0.74     16190
   macro avg       0.76      0.74      0.74     16190
weighted avg       0.76      0.74      0.74     16190
```

Decision Tree

```
accuracy score for train data:  0.5585712081468909
accuracy score for test data:  0.5621988882025942
confusion_matrix:
[[1527 6568]
 [ 520 7575]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.75      0.19      0.30      8095
         1.0       0.54      0.94      0.68      8095

    accuracy                           0.56     16190
   macro avg       0.64      0.56      0.49     16190
weighted avg       0.64      0.56      0.49     16190
```

Random Forest

```
accuracy score for train data:  1.0
accuracy score for test data:  0.7781346510191476
confusion_matrix:
[[7662  433]
 [3159 4936]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.71      0.95      0.81      8095
         1.0       0.92      0.61      0.73      8095

    accuracy                           0.78     16190
   macro avg       0.81      0.78      0.77     16190
weighted avg       0.81      0.78      0.77     16190
```

Logistic Regression

```
accuracy score for train data:  0.9598364449930566
accuracy score for test data:  0.8277331686226066
confusion_matrix:
[[7385  710]
 [2079 6016]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.78      0.91      0.84      8095
         1.0       0.89      0.74      0.81      8095

    accuracy                           0.83     16190
   macro avg       0.84      0.83      0.83     16190
weighted avg       0.84      0.83      0.83     16190
```

XGB Boost

Model	Accuracy	Recall	Precision	F1-Score
DT	0.74	0.59	0.85	0.7
RF	0.78	0.61	0.92	0.74
LR	0.56	0.94	0.54	0.68
GB	0.79	0.78	0.79	0.79
AdaBoost	0.77	0.75	0.78	0.76
XGB	0.83	0.74	0.89	0.81

With Principle Component Analysis we are unable to increase accuracy. In most of the models there is overfitting observed. False Negatives are high which is a red flag while predicting Bankruptcy. Model is seen to be performing well on XgBoost classifier. Also, we find that linear models like logistic regression is not performing well.

F) Smote with 32 features:

```
# Logistic Regression Model with SMOTET.
confusion_matrix:
[[9384 3054]
 [6355 5996]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.60      0.75      0.67     12438
         1.0       0.66      0.49      0.56     12351

    accuracy                           0.62     24789
   macro avg       0.63      0.62      0.61     24789
weighted avg       0.63      0.62      0.61     24789

accuracy_score:
0.6204364839243213
```

```
# Random Forest Model with SMOTE with hyperparameter tuning
confusion_matrix:
[[12021   417]
 [  185 12166]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.98      0.97      0.98     12438
         1.0       0.97      0.99      0.98     12351

    accuracy                           0.98     24789
   macro avg       0.98      0.98      0.98     24789
weighted avg       0.98      0.98      0.98     24789

accuracy_score:
0.9757150348945096
```

```
# XGBClassifier Model with SMOTET without hyperparameter tuning
confusion_matrix:
[[12201   237]
 [  199 12152]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.98      0.98      0.98     12438
         1.0       0.98      0.98      0.98     12351

    accuracy                           0.98     24789
   macro avg       0.98      0.98      0.98     24789
weighted avg       0.98      0.98      0.98     24789

accuracy_score:
0.9824115535116382
```

```
# XGBClassifier Model with 50% SMOTE without hyperparameter tuning
confusion_matrix:
[[12261   173]
 [  263  5895]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.98      0.99      0.98     12434
         1.0       0.97      0.96      0.96      6158

    accuracy                           0.98     18592
   macro avg       0.98      0.97      0.97     18592
weighted avg       0.98      0.98      0.98     18592

accuracy_score:
0.9765490533562823
```

Model	Accuracy	Recall	Precision	F1 score
DT	0.92	0.93	0.91	0.92
DT(tuned)	0.92	0.94	0.91	0.93
RF	0.98	0.98	0.97	0.98
RF(tuned)	0.98	0.98	0.97	0.98
LR	0.62	0.48	0.66	0.55
AdaBoost	0.84	0.83	0.84	0.84
GB	0.89	0.9	0.89	0.89
XGB	0.98	0.98	0.98	0.98
XGB(tuned)	0.98	0.98	0.98	0.98

G) Smote Tomek with 32 features:

```
# Logistic Regression Model with SMOTETomek.

confusion_matrix:
[[8908 3063]
 [6263 5984]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.59      0.74      0.66     11971
         1.0       0.66      0.49      0.56     12247

    accuracy                           0.61     24218
   macro avg       0.62      0.62      0.61     24218
weighted avg       0.62      0.61      0.61     24218

accuracy_score:
0.6149145263853332
```

```
#Random Forest Model with SMOTETomek with hyperparameter tuning

confusion_matrix:
[[11561   410]
 [  162 12085]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.99      0.97      0.98     11971
         1.0       0.97      0.99      0.98     12247

    accuracy                           0.98     24218
   macro avg       0.98      0.98      0.98     24218
weighted avg       0.98      0.98      0.98     24218

accuracy_score:
0.9763812040630936
```

```
# XGBClassifier Model with SMOTETomek without hyperparameter tuning

confusion_matrix:
[[11714   257]
 [  174 12073]]
classification_report:
              precision    recall  f1-score   support

         0.0       0.99      0.98      0.98     11971
         1.0       0.98      0.99      0.98     12247

    accuracy                           0.98     24218
   macro avg       0.98      0.98      0.98     24218
weighted avg       0.98      0.98      0.98     24218

accuracy_score:
0.9822033198447436
```

Model	Accuracy	Recall	Precision	F1 score
DT	0.92	0.94	0.91	0.93
DT(Tuned)	0.93	0.94	0.92	0.93
RF	0.98	0.98	0.97	0.98
RF(Tuned)	0.98	0.99	0.97	0.98
LR	0.61	0.49	0.66	0.56
AdaBoost	0.84	0.83	0.84	0.84
GB	0.9	0.9	0.9	0.9
XGB	0.98	0.99	0.98	0.98
XGB(Tuned)	0.98	0.98	0.98	0.98
Stacking	0.98	0.98	0.98	0.98

Hyperparameter tunning:

- From the base models we found out that by using SMOTE better and more reliable scores were generated.
- For further enhancing the model predictions parameters were tunned. Random Forest, GradientBoostClassifier and XGBoostClassifier were selected for Hyperparameter tunning based on their results.
- Random Forest:
- A customised grid search was done for finding out the n_estimators by looking at the bias and variance error for each estimator. Along with parameters such as max_depth and criterion. Other parameters were searched for using gridsearch but the model performance didn't improve and were therefore kept as default.
- XgBoost:
- Gridsearch was done for finding out the best parameters of XgBoost. Parameters used were eta or learning rate, max_depth and n_estimators.
- The purpose of restricting parameters was computational power.

Model Building after hyperparameter tuning

- After finding out optimum parameters from GridSearch and custom search different iterations of models were tried with random forest and XgBoost.
- Models were built keeping into mind the business problem of keeping the false negatives as low as possible.
- For these iterations included 1:1 SMOTE. (Majority: Minority)
- The minority class was oversampled to match the majority class.
- 1:0.33 and 1:0.32 SMOTE partial oversampling of minority class was done. The above ratios were found out after comprehensive search by checking the metrics for each ratio of SMOTE.
- Then SMOTE Tomek was done which uses over-sampling and under sampling together. Oversampling using SMOTE and under sampling using Tomek Links.
- Here also different ratios of majority and minority class were used for model to get desired results.

33% partial Smote Full Data:

```
# A Tuned XGBOOST MODEL -- SMOTETomek Done and ALL  FEATURES USED. ---- 33% SMOTETomek
```

```
==========================================================================================
0.9997856607008895
[[28324     0]
 [    8  8992]]
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     28324
         1.0       1.00      1.00      1.00      9000

    accuracy                           1.00     37324
   macro avg       1.00      1.00      1.00     37324
weighted avg       1.00      1.00      1.00     37324

==========================================================================================
0.9845596049259236
[[12137    40]
 [  207  3613]]
              precision    recall  f1-score   support

         0.0       0.98      1.00      0.99     12177
         1.0       0.99      0.95      0.97      3820

    accuracy                           0.98     15997
   macro avg       0.99      0.97      0.98     15997
weighted avg       0.98      0.98      0.98     15997

==========================================================================================
```

Train

Test

32% Partial Smote Full Data

```
Tuned XGBOOST MODEL -- SMOTETomek Done and ALL  FEATURES USED. ---- 32% SMOTETomek
```

```
==========================================================================================
0.9995402423193422
[[28386     0]
 [   17  8573]]
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     28386
         1.0       1.00      1.00      1.00      8590

    accuracy                           1.00     36976
   macro avg       1.00      1.00      1.00     36976
weighted avg       1.00      1.00      1.00     36976

==========================================================================================
0.9858026249369005
[[12035    38]
 [  187  3588]]
              precision    recall  f1-score   support

         0.0       0.98      1.00      0.99     12073
         1.0       0.99      0.95      0.97      3775

    accuracy                           0.99     15848
   macro avg       0.99      0.97      0.98     15848
weighted avg       0.99      0.99      0.99     15848
```

Train

Test

100% smote Full Data

```
# A Tuned XGBOOST MODEL -- SMOTETomek Done and ALL  FEATURES USED. ---- 100% SMOTETomek
```

```
====================================================================================================
0.9998944089541207
[[28495     0]
 [    6 28322]]
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     28495
         1.0       1.00      1.00      1.00     28328

    accuracy                           1.00     56823
   macro avg       1.00      1.00      1.00     56823
weighted avg       1.00      1.00      1.00     56823
```

Train

```
====================================================================================================
0.9919927729643165
[[11969   124]
 [   71 12189]]
              precision    recall  f1-score   support

         0.0       0.99      0.99      0.99     12093
         1.0       0.99      0.99      0.99     12260

    accuracy                           0.99     24353
   macro avg       0.99      0.99      0.99     24353
weighted avg       0.99      0.99      0.99     24353
```

Test

- For the above models different feature sets were used.
- In all the purpose of doing the above iterations was to take into consideration the business problem where A creditor must not lose money on bankrupting companies and for this False Negatives (type II error) was to be kept as minimum as possible.

Hyperparameter tuning with 32 features:

```
# A Tuned XGBOOST MODEL -- 100% SMOTETomek Done and TOP 32 FEATURES from correlation method
```

```
====================================================================================================
 Train data
0.9997876594765807
[[28294     6]
 [    6 28207]]
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     28300
         1.0       1.00      1.00      1.00     28213

    accuracy                           1.00     56513
   macro avg       1.00      1.00      1.00     56513
weighted avg       1.00      1.00      1.00     56513

====================================================================================================
Test data
0.9855497295735106
[[11829   238]
 [  112 12042]]
              precision    recall  f1-score   support

         0.0       0.99      0.98      0.99     12067
         1.0       0.98      0.99      0.99     12154

    accuracy                           0.99     24221
   macro avg       0.99      0.99      0.99     24221
weighted avg       0.99      0.99      0.99     24221

====================================================================================================
 Entire DATA- verification
0.9905310448105057
[[40985   329]
 [   82  2009]]
              precision    recall  f1-score   support

         0.0       1.00      0.99      1.00     41314
         1.0       0.86      0.96      0.91      2091

    accuracy                           0.99     43405
   macro avg       0.93      0.98      0.95     43405
weighted avg       0.99      0.99      0.99     43405
```

Train

Test

Entire Data

Final Model

We have selected XgBoost model with 32 features based on collinearity. This model is not overfitting at all, also while testing it on entire data we are getting good results with accuracy of 99%. Though some corporate entities are falsely being classified as Bankrupt still from a business point of view of creditors, it is not a threat. From creditors point of view classifying probable bankrupt company as non-Bankrupt is a bigger threat.

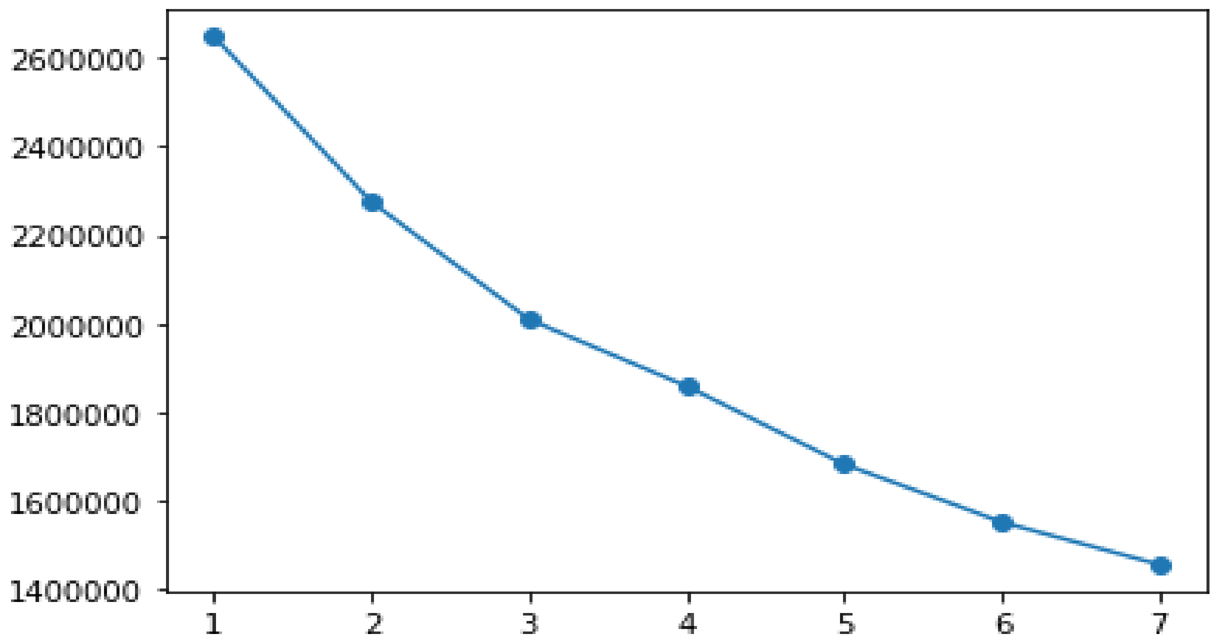
Important Features in determining class:

XGBClassifier		
Feature	**Importance**	**Description**
Attr6	0.144509	retained earnings / total assets
Attr27	0.103829	profit on operating activities / financial expenses
Attr26	0.096343	(Net profit + depreciation) / total liabilities

Attr39	0.052176	profit on sales / sales
Attr59	0.044391	long-term liabilities / equity

Key risks and Way forward:

From the clustering using KMeans we find that without scaled data we get a logarithmic elbow plot. Thus, signifying that these corporate entities come from a very different background. These may be small entities, medium entities or large entities. Even the area of operation may be different.




```
]: c=[2,3,4,5,6,7,8]
   for i in c:
       cluster = KMeans(n_clusters=i)

       # Fitting the input data
       # Getting the cluster labels

       l = cluster.fit_predict(X_sc)

       score=silhouette_score(X_sc,l,random_state=10)
       print('the silhouette score for ',i,'is:',score)

   the silhouette score for  2 is: 0.9965098934479747
   the silhouette score for  3 is: 0.9958760110056343
   the silhouette score for  4 is: 0.9880211393905831
   the silhouette score for  5 is: 0.9877083400354557
   the silhouette score for  6 is: 0.9878878944836631
   the silhouette score for  7 is: 0.9879668364745929
   the silhouette score for  8 is: 0.9841864345912715
```

Also, we came across many outliers and according to domain these outliers are bound to be there because some companies are comparatively very large while some are very small. And existence of both these entities is must for a proper model building. We see that without Outlier handling we are getting good accuracy scores.

REFERENCES (Provide at least 10 references for the project proposal)

1)Zieba, M., Tomczak, S. K., & Tomczak, J. M. (2016). Ensemble Boosted Trees with Synthetic Features Generation in Application to Bankruptcy Prediction. Expert Systems with Applications.

2) Wikipedia contributors. "Bankruptcy prediction". Wikipedia, The Free Encyclopedia, <https://en.wikipedia.org/wiki/Bankruptcy_prediction>

Notes For Project Team

Sample Reference for Datasets (to be filled by team and mentor)

Original owner of data	Sebastian Tomczak Department of Operations Research, University of Science and Technology, Poland
Data set information	Polish companies bankruptcy data Data Set
Previous relevant journals used the data set	Zieba, M., Tomczak, S. K., & Tomczak, J. M. (2016). Ensemble Boosted Trees with Synthetic Features Generation in Application to Bankruptcy Prediction. Expert Systems with Applications.

Citation	Zieba, M., Tomczak, S. K., & Tomczak, J. M. (2016). Ensemble Boosted Trees with Synthetic Features Generation in Application to Bankruptcy Prediction. Expert Systems with Applications
Link to web page	https://archive.ics.uci.edu/ml/datasets/Polish+companies+bankruptcy+data