MTL 782 DATA MINING ASSIGNMENT 1

POLISH BANKRUPTCY DATA DATASET

2nd March, 2020

Vinit Saini	2017MT60789
Prajay Sapkale	2017MT10743
Sri Krishna Sahoo	2017MT60786

INDEX

1. INTRODUCTION

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.

Dataset characteristic	Multivariate			
Number of Features	64			
Number of Instances	**Data**	**Total Instances**	**Bankrupt instances**	**Non-bankrupt instances**
	1st year	7027	271	6756
	2nd year	10173	400	9773
	3rd year	10503	495	10008
	4th year	9792	515	9227
	5th year	5910	410	5500
Feature characteristics	Real values			
Has missing data?	Yes			
Associated tasks	Classification			
Date donated	04-11-2016			

Prediction of an enterprise bankruptcy is of great importance in economic decision making. The purpose of the bankruptcy prediction is to assess the financial condition of a company and its future perspectives within the context of long-term operation on the market.
The dataset we have considered for addressing the bankruptcy prediction problem is the Polish bankruptcy data, hosted by UCI Machine Learning Repository. The dataset is very apt for our assignment about bankruptcy prediction because it has useful econometric indicators as attributes (features) and comes with a huge number of samples of Polish companies that were analyzed in 5 different timeframes.

2. EXPLORATORY DATA ANALYSIS

Knowing our dataset is very crucial for a successful machine learning model. Hence, we have tried to gain insights by plotting various correlation matrices for

our dataset. Also, getting to know the number of missing values was very much needed to decide whether to drop those rows, or perform imputation.
We started by first getting obtaining the basic information such as :

```
year1.info(verbose=False)
```

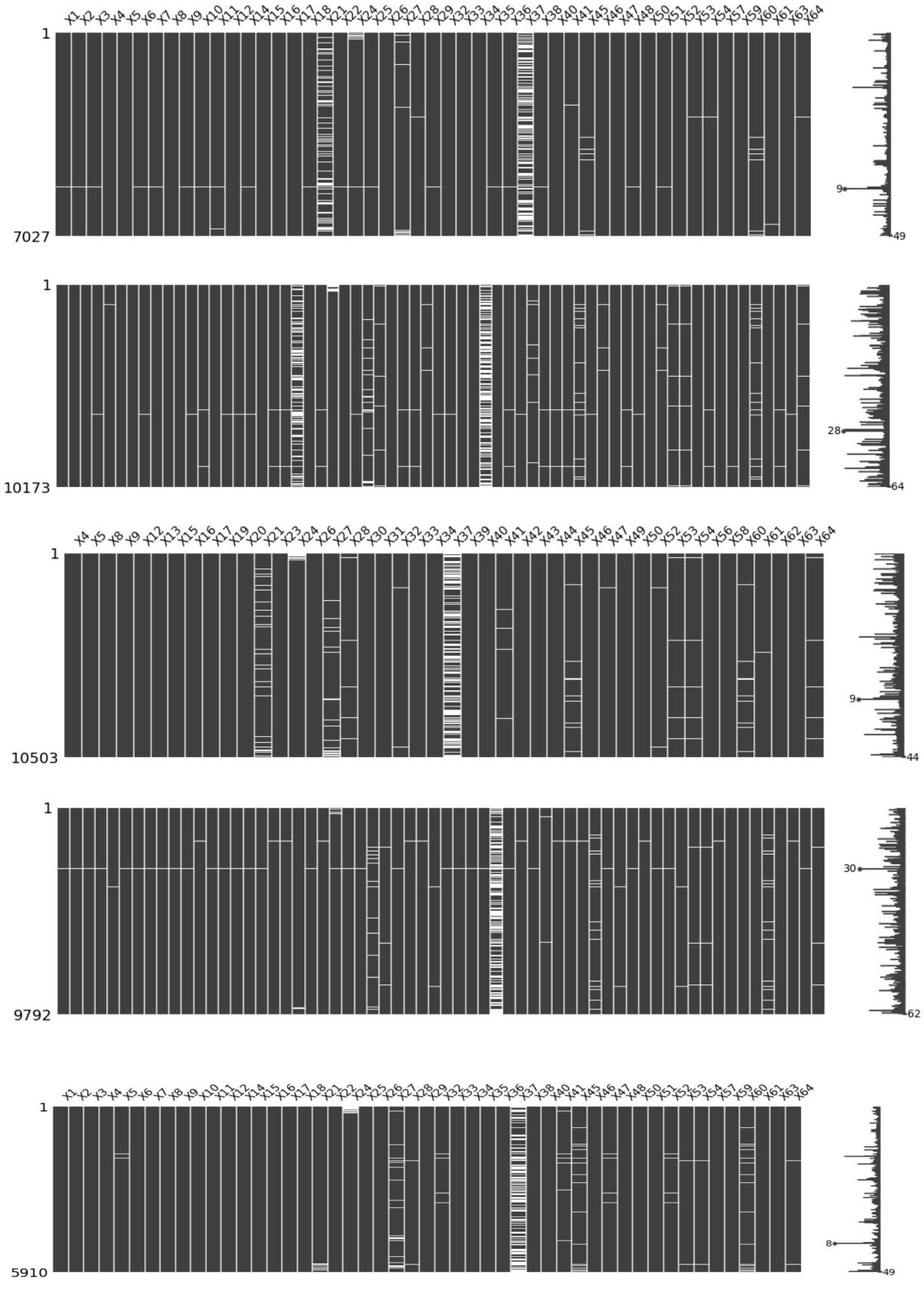
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7027 entries, 0 to 7026
Columns: 65 entries, X1 to Y
dtypes: float64(64), int64(1)
memory usage: 3.5 MB
```

```
year1.isnull().sum().sum()
```

```
5835
```

It was palpable that there were missing values in the dataset. Now we thought of checking how these values were distributed column-wise. If it were the case that the NaN were dense in a few columns, we would have decided to drop those columns. But that wasn't the case. Inshort, plotting these graphs helped to check the distribution of the missing values.

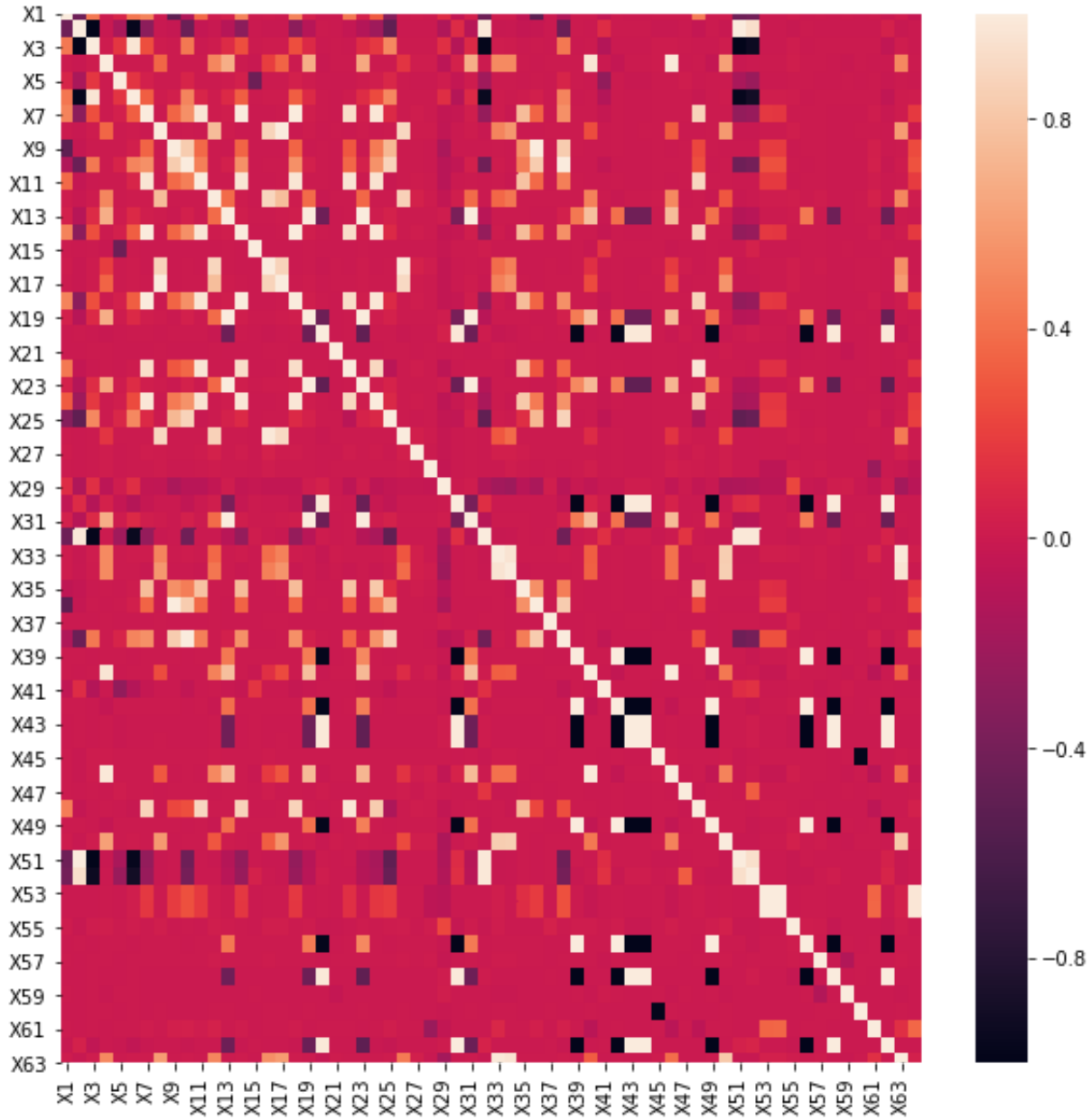
In the following images which show the missing values, It is observed that the columns that are dense in missing values are different for each of the years. Thus, we had to do feature selection for the years separately.


X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X14 X15 X16 X17 X18 X21 X22 X24 X25 X26 X27 X28 X29 X32 X33 X34 X35 X36 X37 X38 X40 X41 X45 X46 X47 X48 X50 X51 X52 X53 X54 X57 X59 X60 X61 X63 X64
1
7027
9
49
1
10173
28
64
X4 X5 X8 X9 X12 X13 X15 X16 X17 X19 X20 X21 X23 X24 X26 X27 X28 X30 X31 X32 X33 X34 X37 X39 X40 X41 X42 X43 X44 X45 X46 X47 X49 X50 X52 X53 X54 X56 X58 X60 X61 X62 X63 X64
1
10503
9
44
1
9792
30
62
X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X14 X15 X16 X17 X18 X21 X22 X24 X25 X26 X27 X28 X29 X32 X33 X34 X35 X36 X37 X38 X40 X41 X45 X46 X47 X48 X50 X51 X52 X53 X54 X57 X59 X60 X61 X63 X64
1
5910
8
49

HEATMAPS

We plotted heatmaps that show the correlations among features to try to figure out which features were redundant, as in giving same information as some other features. These are from the mean imputed datasets.

A) First Year



2. IMPUTATION

Missing data causes 3 problems :

- Missing data can introduce a substantial amount of bias.
- Makes the handling and analysis of the data more difficult.
- Create reductions in efficiency.

Dropping all the rows with missing values, introduces bias and affects representativeness of the results. The only viable alternative to is Imputation. 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. In our project we explored two techniques of imputation, and we will see them in the subsequent sections. 1. Mean Imputation 2. k-Nearest Neighbours Imputation

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. Mean imputation attenuates any correlations involving the variable(s) that are imputed. This is because, in cases with imputation, there is guaranteed to be no relationship between the imputed variable and any other measured variables. Thus, mean imputation has some attractive properties for univariate analysis but becomes problematic for multivariate analysis. Hence we opted for Mean Imputation as a baseline method. We achieved mean imputation using scikit-learn's SimpleImputer class.

K-NEAREST NEIGHBOURS IMPUTATION

The k-nearest neighbors algorithm or k-NN, is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. 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. We used the KNNImputer library to perform k-NN data imputation, and we used 50 nearest neighbors for the process.

3. SPLITTING

After data imputation, for each year, we split the data into training and testing sets in the ratio 0.85 : 0.15. This was done to preserve the notion of 'unseen' test data. The test data remains untouched (no oversampling) other than the feature reduction which has to be done on test set too before prediction as the model is rained on reduced feature train set.

4. SAMPLING

The dataset is highly imbalanced, i.e., there are a huge number of instances with target value '0' as compared with '1'. Hence, we employ the technique of Oversampling, i.e., increasing the class distribution of the minority class label. This was achieved using the SMOTE library. Below, we can see the class distribution of the 4year dataset.

```
(8323, 64) (15792, 64)
Resampled dataset shape Counter({0.0: 7896, 1.0: 7896}), Counter({0.0: 7896, 1.0: 427})
```

5. FEATURE SELECTION

This is an important task for building a good model. There are various inbuilt methods that are available for usage such as RFE, Backward Elimination, LassoLars, etc. Since we have done the EDA task extensively, we tried to compare the feature selection technique of RFE and the one CREATED BY US, we call it 'EDA FEATURES'.

How did we create EDA features?
We used the correlation plots to delete the features that had a correlation of greater than 0.95 with another feature. These features are visible on the heatmap shown above.

Recursive Feature Elimination (RFE): Given an external estimator that assigns weights to features (e.g., the coefficients of a linear model), the goal of recursive feature elimination (RFE) is to select features by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features and the importance of each feature is obtained either through a coef_ attribute or through a feature_importances_ attribute. Then, the least important features are pruned from the current set of features. That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached.

We have done recursive elimination in steps of 2 and with minimum features = 32 as we observed not more than 20 features seemed to be redundant. In any case, at the most, 23 features were eliminated so the lower bound of 32 is reasonable.

Finally, we built models on these four pre-processed datasets:

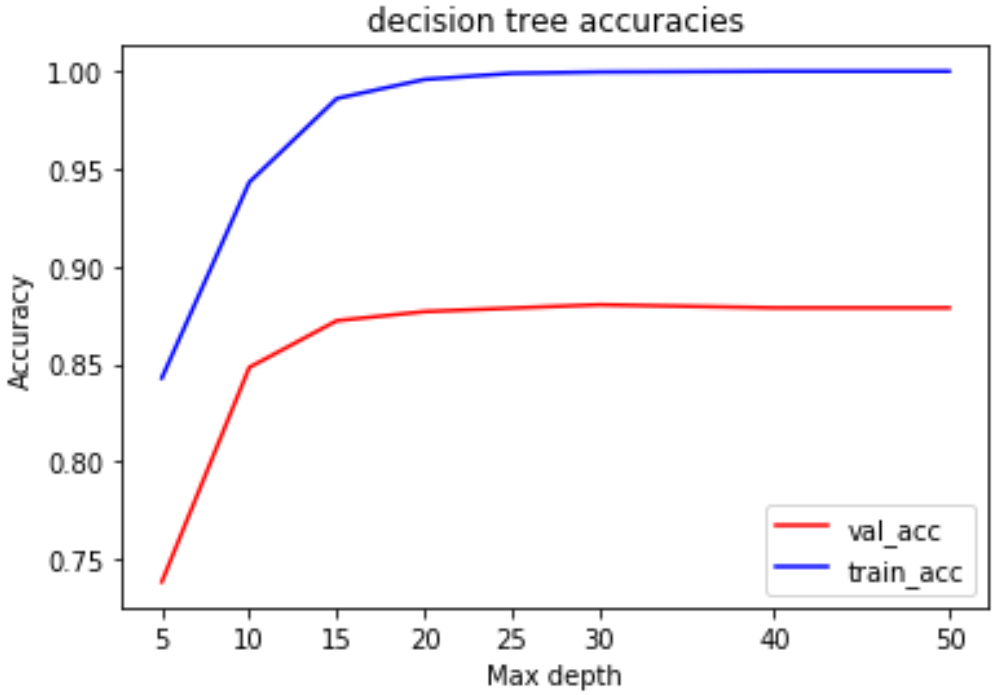
- Mean Imputed - EDA features
- Mean Imputed - RFE features
- kNN Imputed - EDA features
- kNN Imputed - RFE features

6. OBSERVATIONS

1) Decision Trees (with pruning)

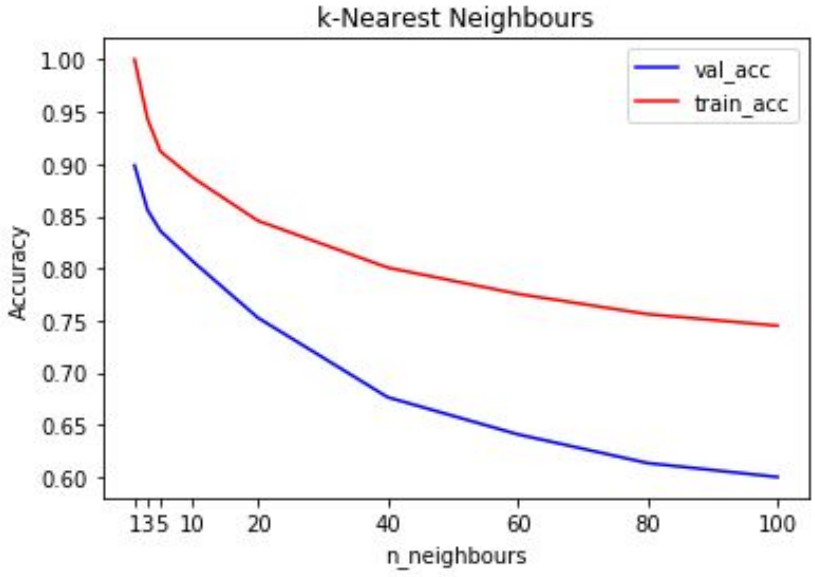
Changing different parameters in the decision tree are used to stop the tree from overfitting and finding the best parameters for estimation. For this, 5 fold cross validation has been performed. Here are graphs of train vs validation accuracies

which show the point after which train accuracies increases and validation accuracy decreases. The following graph is plotted for year 5 with mean imputation and rfe oversampling done.



We can observe that after a depth of around 25, the validation accuracies starts to decline. Hence the optimal parameter for other trees is set to 25.

2) k-Nearest Neighbours



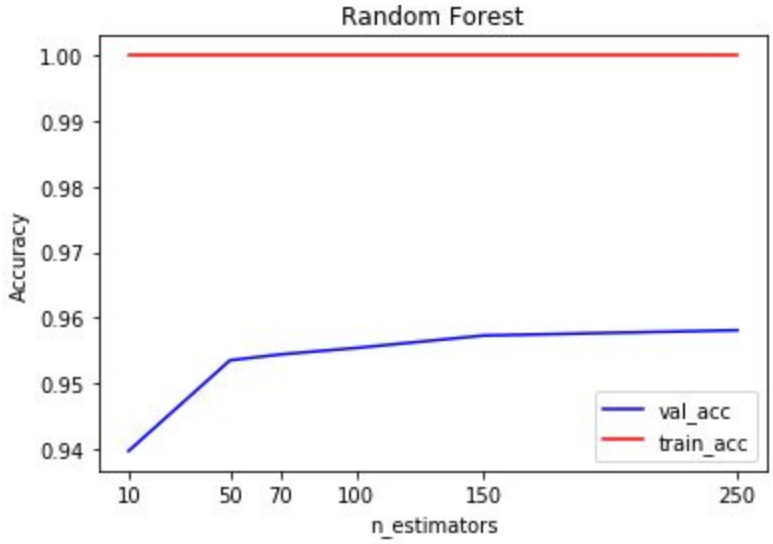
3) SVM

Linear Kernel : 40% Val_Acc 70% Train_Acc

RBF : 68% Val_Acc 80% Train_Acc

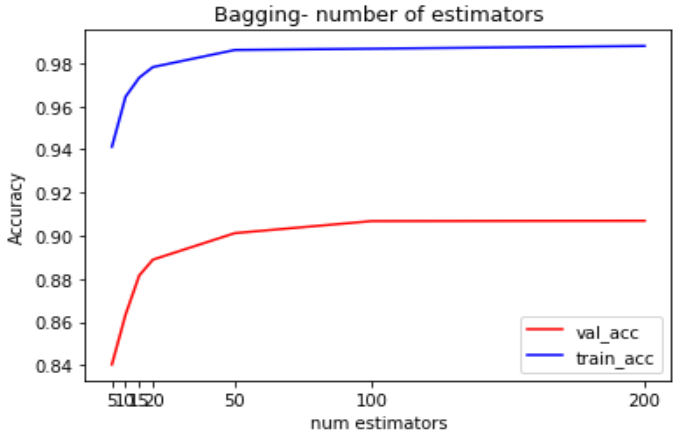
Thus, we choose the RBF kernel.

4) Random Forest



5) Bagging

For Bagging the tunable parameters are the number of estimators and the fraction of features and to be considered in one estimator. The plots for different values of the three parameters are shown here followed by the best parameters that are used to predict on the datasets.



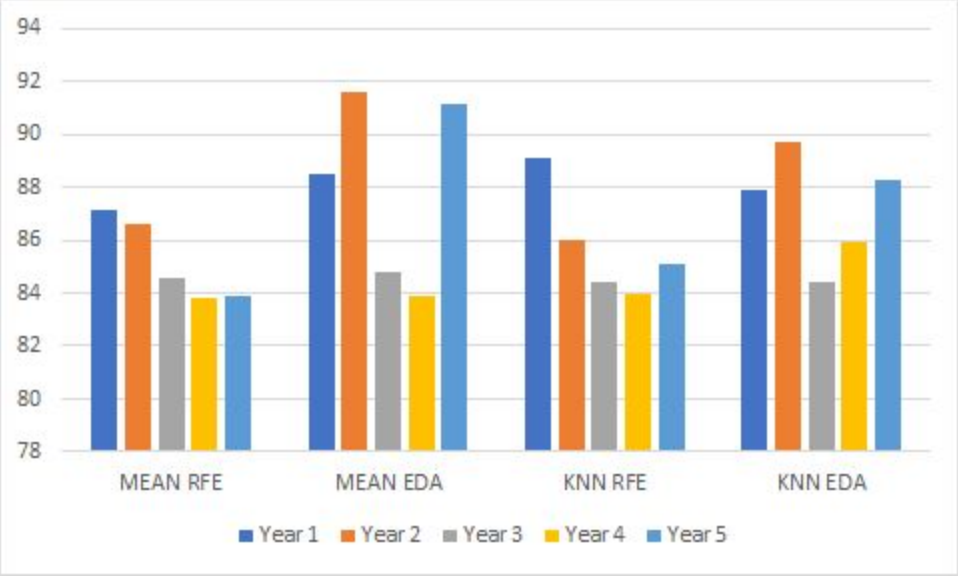
7. OUTLIER REMOVAL

We have used z-score which is the signed number of standard deviations by which the value of an observation or data point is above the mean value of what is being observed or measured. We discarded the rows that had z-score greater 2, thus treating them as outliers. Scipy.stats library was used.

8. PREDICTIONS

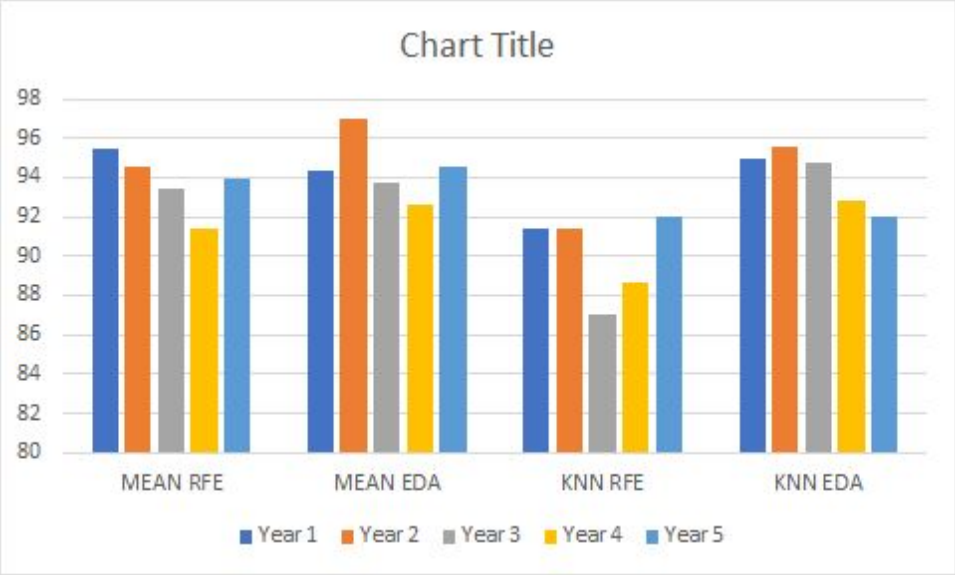
1) Decision Trees (with pruning)

		Year 1	Year 2	Year 3	Year 4	Year 5
Mean	RFE	87.11	86.57	84.58	83.80	83.88
	EDA	88.53	91.61	84.77	83.87	91.10
KNN	RFE	89.10	85.98	84.45	83.93	85.12
	EDA	87.86	89.71	84.39	85.91	88.28



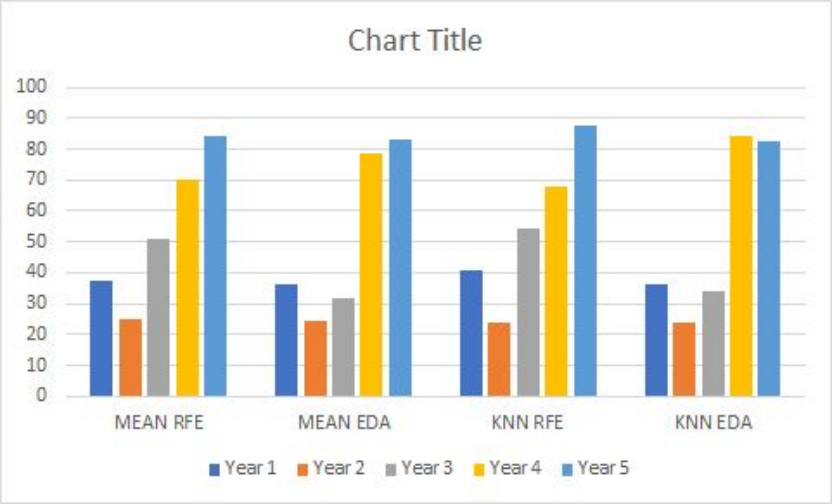
2) Random Forest

		Year 1	Year 2	Year 3	Year 4	Year 5
Mean	RFE	95.45	94.50	93.46	91.42	93.91
	EDA	94.31	96.99	93.78	92.58	94.59
KNN	RFE	91.37	91.41	86.99	88.63	92.00
	EDA	94.97	95.60	94.73	92.85	92.00



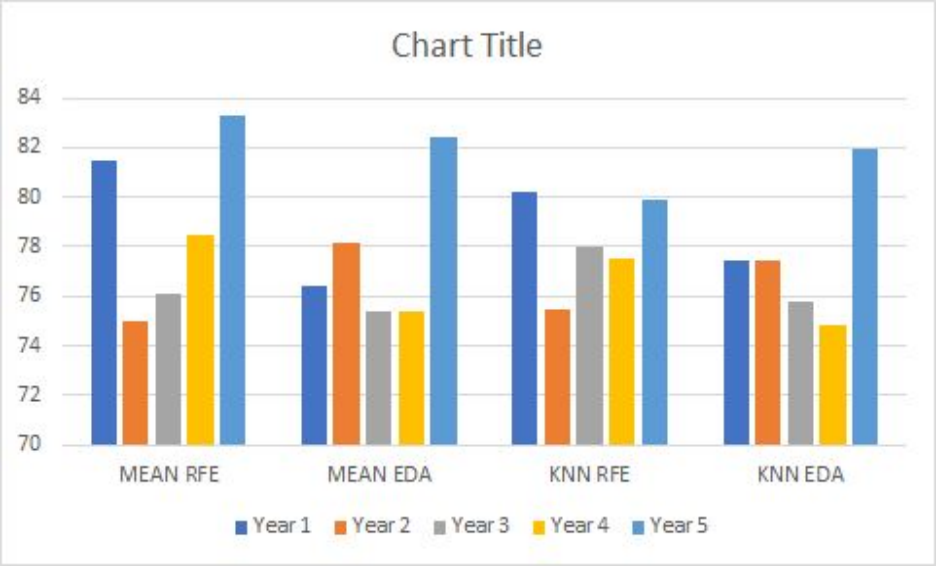
3) Naive Bayes Classifier

		Year 1	Year 2	Year 3	Year 4	Year 5
Mean	RFE	37.15	25.03	51.07	69.97	84.32
	EDA	36.39	24.31	31.59	78.69	83.08
KNN	RFE	40.85	23.91	54.56	67.93	87.37
	EDA	36.20	24.04	34.20	84.27	82.29



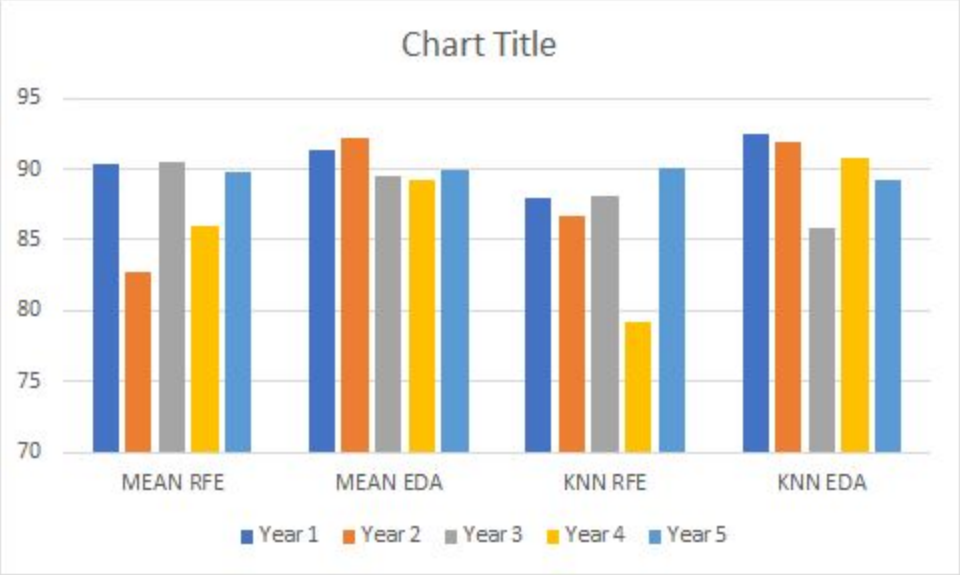
4) K-nearest neighbour

		Year 1	Year 2	Year 3	Year 4	Year 5
Mean	RFE	81.51	75.03	76.07	78.48	83.31
	EDA	76.39	78.17	75.38	75.35	82.41
KNN	RFE	80.18	75.49	77.98	77.53	79.93
	EDA	77.44	77.45	75.76	74.81	81.96



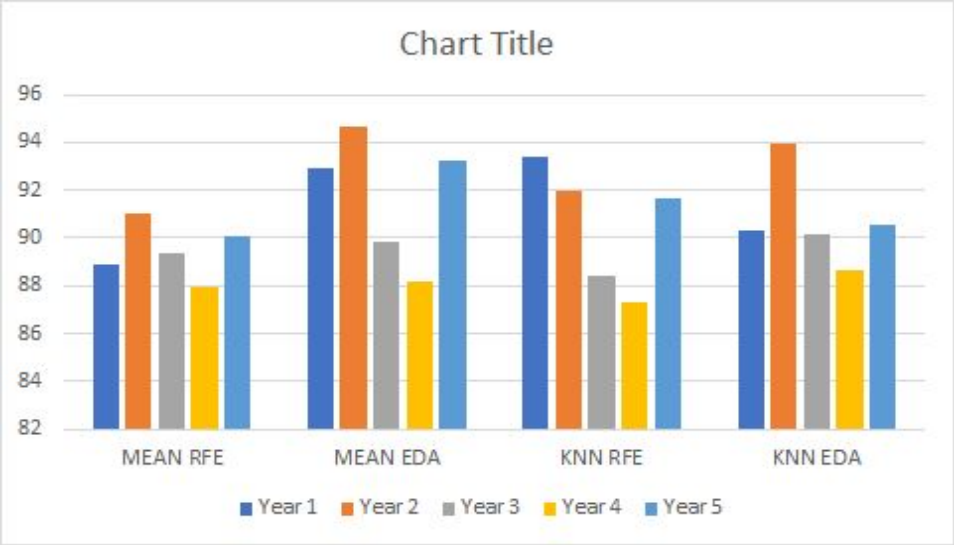
5) Artificial Neural network

		Year 1	Year 2	Year 3	Year 4	Year 5
Mean	RFE	90.33	82.70	90.55	86.02	89.74
	EDA	91.37	92.20	89.47	89.28	89.96
KNN	RFE	87.96	86.63	88.07	79.25	90.07
	EDA	92.51	91.94	85.85	90.75	89.28



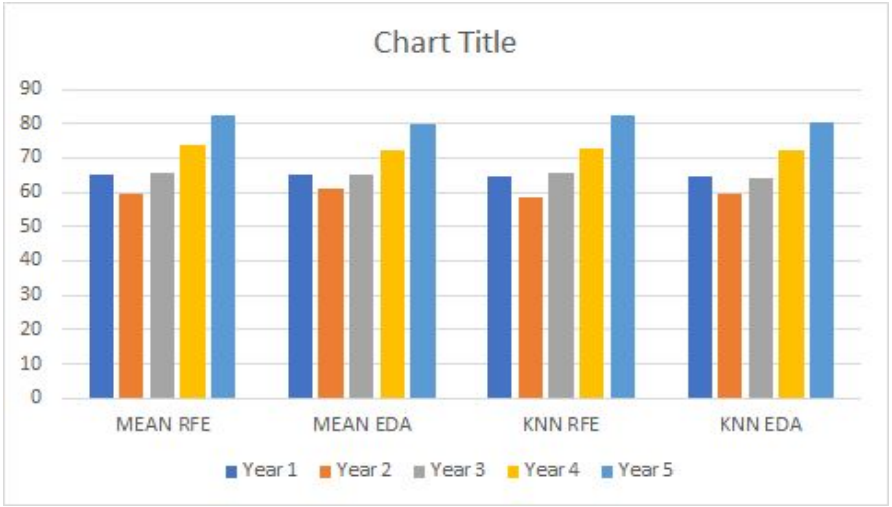
6) Bagging

		Year 1	Year 2	Year 3	Year 4	Year 5
Mean	RFE	88.91	91.02	89.40	87.95	90.08
	EDA	92.89	94.63	89.84	88.16	93.24
KNN	RFE	93.36	92.01	88.39	87.34	91.66
	EDA	90.33	93.91	90.16	88.63	90.53



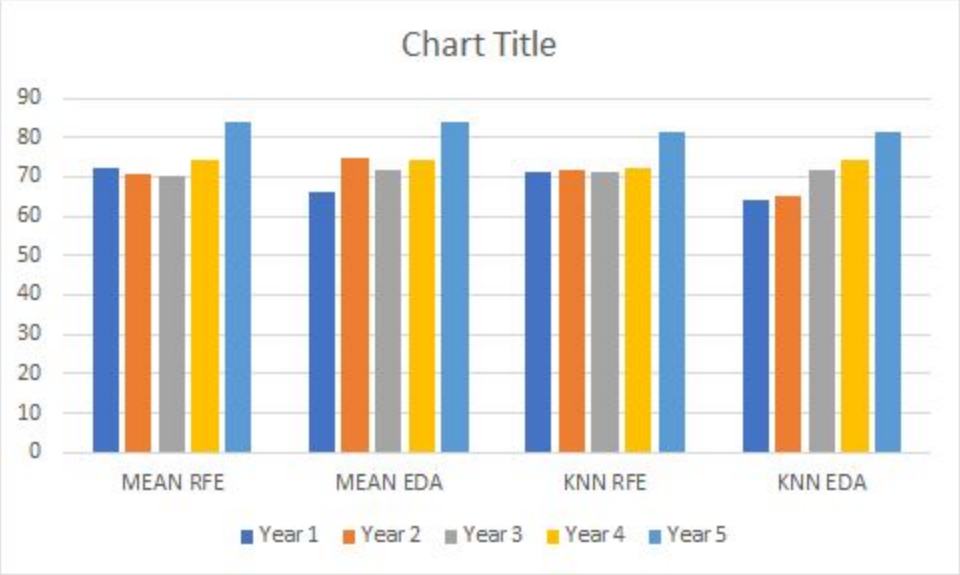
7) Logistic Regression

		Year 1	Year 2	Year 3	Year 4	Year 5
Mean	RFE	64.92	59.50	65.54	73.85	82.63
	EDA	65.21	60.81	65.10	72.36	80.04
KNN	RFE	64.73	58.45	65.73	72.90	82.63
	EDA	64.40	59.69	64.27	72.43	80.15



8) Support Vector Machine

		Year 1	Year 2	Year 3	Year 4	Year 5
Mean	RFE	72.03	70.64	69.98	74.13	84.10
	EDA	65.97	74.96	71.95	74.33	83.76
KNN	RFE	71.27	71.49	71.19	72.02	81.51
	EDA	64.36	65.00	71.70	74.06	81.51



8. Conclusion

We have successfully modelled eight classification models: Gaussian Naïve Bayes, Logistic Regression, Decision Trees, Random Forests, Support Vector Machine, ANN, k-Nearest Neighbours and Bagging classifiers. The training sets were made sure to have a balanced sets of class labels, by oversampling the minority class labels using Synthetic Minority Oversampling technique. Also, we have imputed the missing values in the data using two imputer techniques: Mean and k-Nearest

Neighbors (k-NN). The biggest challenge was to deal with the missing/sparse data. Since all the companies being evaluated for bankruptcy don't operate on the same timelines, it is difficult to gather meaningful data and organize it. Outlier removal also helped us to learn a crucial technique to increase prediction accuracy. We also employed two feature selection techniques : RFE and EDA (our nomenclature). Thus, combined with the two imputation techniques, we have FOUR DATASETS to build model on and contrast results. Following are some of the general observations:

1. Naive Bayes works better on later years. This might be due to the fact that the last conditions of the companies are most likely to give better prediction of future bankruptcy.
2. Random forest and Bagging are both among the best accuracies. This can be attributed to the fact that Random Forest provide a reliable feature importance estimate. They offer efficient estimates of the test error without incurring the cost of repeated model training associated with cross-validation.
3. We expected that later years would give better predictions as they are closer to the 5th year, but that didn't happen in most of the models.

9. References

- Exploratory Data Analysis: A Practical Guide and Template for Structured Data
- Ways to Detect and Remove the Outliers
- Polish companies bankruptcy data Data Set