Final Exam : Improvements & Stock Data Analysis

Final Exam : EE4708 Data Analytics Laboratory

Madhur Jindal
Inter Disciplinary Dual Degree Program in Data Science
Indian Institute of Technology (IIT) Madras
Chennai, India
me18b059@smail.iitm.ac.in

***Abstract*—This document is the final exam for EE4708 Data Analytics Laboratory including improvements using different methods in the previous assignments and a time series Stock prediction analysis problem. We are provided with stock data for different companies and our goal is to get meaningful insights from the different features present and finally predict risk and associated gain values for the different company stocks. We also try to apply different methods like upsampling, downsampling, polynomial features and outlier treatment using IQR method and try to get better insights and performance on the previous assignments.**

I. Introduction

Time series analysis is a specific way of analyzing a sequence of data points collected over an interval of time. In time series analysis, analysts record data points at consistent intervals over a set period of time rather than just recording the data points intermittently or randomly. Stock market analysis enables investors to identify intrinsic worth of a security even before investing in it. By using stock analysis, investors and traders arrive at equity buying and selling decisions. Outlier Analysis is a process that involves identifying the anomalous observation in the dataset. Outliers are caused due to the incorrect entry or computational error, is-reporting, sampling error, Exceptional but true value error. Most data mining methods discard outliers noise or exceptions, however, in some applications such as fraud detection, the rare events can be more interesting than the more regularly occurring one and hence, the outlier analysis becomes important in such case. The main two methods that are used to tackle the class imbalance is upsampling/oversampling and down-sampling/undersampling. Upsampling is a procedure where synthetically generated data points (corresponding to minority class) are injected into the dataset. Downsampling is a mech-anism that reduces the count of training samples falling under the majority class. Variance Threshold is a univariate approach to feature selection. It removes all features whose variance doesn't meet some threshold. Polynomial features are those features created by raising existing features to an exponent.

Identify applicable funding agency here. If none, delete this.

II. Final Exam

A. Linear Regression

Linear regression is used for finding linear relationship between target and one or more predictors. Linear regression is a task in which the model being learnt is assumed to be a linear function of parameters. There are three types of linear regression - Simple and Multiple, which involve one and more independent variables respectively. The third one is Polynomial regression which is a generalisation case of multiple linear regression in which the input features include higher powers. One is a predictor and other is the response. It looks for statistical relationship but not deterministic re-lationship. Relationship between two variables is said to be deterministic if one variable can be accurately expressed by the other. For example, using temperature in degrees it is possible to accurately predict Fahrenheit. Statistical relationship is not accurate in determining relationship between two variables. For example, relationship between height and weight.

B. Logistic Regression

Classification is the process of learning a mapping from related input features to discrete data classes or categories. Logistic regression, despite its name, is a technique that can be used to handle binary classification problems. Logistic regression is a process of modelling the probability of a discrete outcome given an input variable. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. The probability is computed by taking the logistic of a linear regression function. The binary logistic model has a dependent variable with two possible values, wherein the corresponding probability of the value labeled '1' can vary between 0 and 1, hence the function that converts the log-odds to probability is the logistic function. The logistic regression model itself simply models probability of output in terms of input and does not perform statistical classification, though it can be used to make a classifier, for instance by choosing a cutoff value and classifying the outputs with probability greater than the cutoff as one class, below the cutoff as other.

C. Gaussian Naive Bayes

In statistics, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features (see Bayes classifier). They are among the simplest Bayesian network models,[1] but coupled with kernel density estimation, they can achieve higher accuracy levels.Another assumption made here is that all the predictors have an equal effect on the outcome. Naïve Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. Naïve Bayes models are also known as simple Bayes or independent Bayes. All these names refer to the application of Bayes' theorem in the classifier's decision rule. Naïve Bayes classifier applies the Bayes' theorem in practice. This classifier brings the power of Bayes' theorem to machine learning.

D. Decision Trees

Decision Trees are versatile Machine Learning algorithms that can perform both classification and regression tasks, and even multi-output tasks. They are very powerful algorithms, capable of fitting complex datasets. Decision Trees split the instances into two or more homogeneous sets based on most significant splitter / differentiator in input variables. Decision Trees are also the fundamental components of Random Forests, which are among the most powerful Machine Learning algorithms available today. It uses a tree like structure and their possible combinations to solve a particular problem. A decision tree is a structure that includes a root node, branches, and leaf nodes. Each internal node denotes a test on an attribute, each branch denotes the outcome of a test, and each leaf node holds a class label. The topmost node in the tree is the root node.

E. Random Forests

Bootstrap aggregating, also called bagging (from bootstrap aggregating), is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting. Although it is usually applied to decision tree methods, it can be used with any type of method. Bagging is a special case of the model averaging approach. Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.

F. Time Series and Stock Prediction

Time series analysis is a specific way of analyzing a sequence of data points collected over an interval of time. In time series analysis, analysts record data points at consistent intervals over a set period of time rather than just recording the data points intermittently or randomly. However, this type of analysis is not merely the act of collecting data over time. What sets time series data apart from other data is that the analysis can show how variables change over time. In other words, time is a crucial variable because it shows how the data adjusts over the course of the data points as well as the final results. It provides an additional source of information and a set order of dependencies between the data. Time series analysis typically requires a large number of data points to ensure consistency and reliability. An extensive data set ensures you have a representative sample size and that analysis can cut through noisy data. It also ensures that any trends or patterns discovered are not outliers and can account for seasonal variance. Additionally, time series data can be used for forecasting—predicting future data based on historical data. Stock market prediction and analysis are some of the most difficult jobs to complete. There are numerous causes for this, including market volatility and a variety of other dependent and independent variables that influence the value of a certain stock in the market. These variables make it extremely difficult for any stock market expert to anticipate the rise and fall of the market with great precision. Stock market analysis enables investors to identify intrinsic worth of a security even before investing in it. By using stock analysis, investors and traders arrive at equity buying and selling decisions.

G. Outlier Analysis & IQR Method

An outlier is an object that deviates significantly from the rest of the objects. They can be caused by measurement or execution error. The analysis of outlier data is referred to as outlier analysis or outlier mining. Most data mining methods discard outliers noise or exceptions, however, in some applications such as fraud detection, the rare events can be more interesting than the more regularly occurring one and hence, the outlier analysis becomes important in such case.

To explain IQR Method easily, let's start with a box plot.

A box plot tells us, more or less, about the distribution of the data. It gives a sense of how much the data is actually spread about, what's its range, and about its skewness. As you might have noticed in the figure, that a box plot enables us to draw inference from it for an ordered data, i.e., it tells us about the various metrics of a data arranged in ascending order.

In the above figure, 1. minimum is the minimum value in the dataset,

2. and maximum is the maximum value in the dataset.

So the difference between the two tells us about the range of dataset.

The median is the median (or centre point), also called second quartile, of the data (resulting from the fact that the data is ordered).

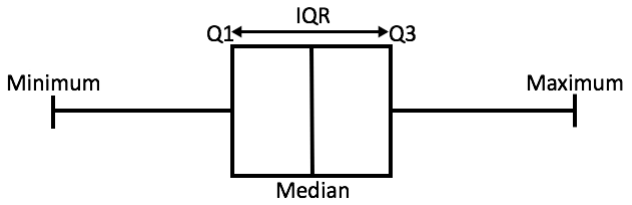


Fig. 1.

1. Q1 is the first quartile of the data, i.e., to say 25% of the data lies between minimum and Q1.

2. Q3 is the third quartile of the data, i.e., to say 75% of the data lies between minimum and Q3.

The difference between Q3 and Q1 is called the Inter-Quartile Range or IQR.

$$IQR = Q3 - Q1$$

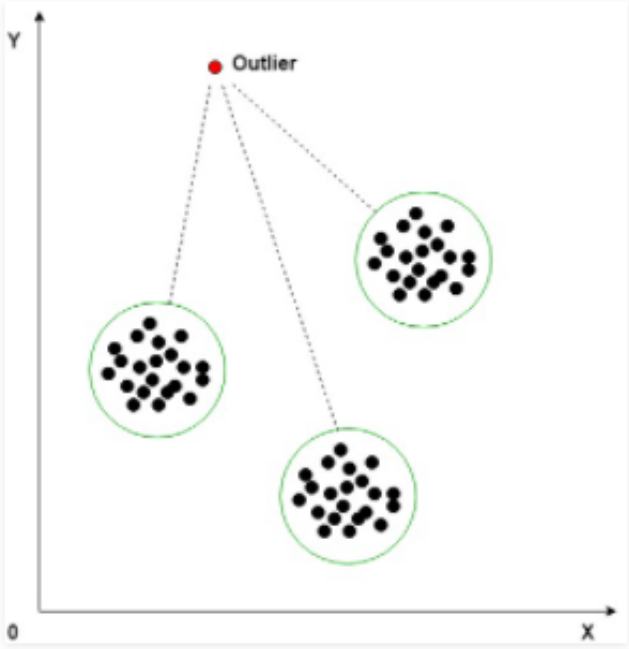


Fig. 2. Outlier Detection

To detect the outliers using this method, we define a new range, let's call it decision range, and any data point lying outside this range is considered as outlier and is accordingly dealt with. The range is as given below:

$$LowerBound : (Q1 - 1.5 * IQR)$$

$$UpperBound : (Q3 + 1.5 * IQR)$$

Any data point less than the Lower Bound or more than the Upper Bound is considered as an outlier.

H. Upsampling and Downsampling

In the real world, the data we gather will be heavily imbalanced most of the time. so, what is an Imbalanced Dataset?. The training samples are not equally distributed across the target classes. For instance, if we take the case of the personal loan classification problem, it is effortless to get the 'not approved' data, in contrast to, 'approved' details. As a result, the model is more biased to the class which has a large number of training instances which degrades the model's prediction power.

It also results in an increase in Type II errors, in the case of a typical binary classification problem. This stumbling block is not just limited to machine learning models but can also be predominantly observed in computer vision and NLP areas as well. These hiccups could be handled effectively by using distinct techniques for each area respectively.

The main two methods that are used to tackle the class imbalance is upsampling/oversampling and downsampling/undersampling. The sampling process is applied only to the training set and no changes are made to the validation and testing data. Imblearn library in python comes in handy to achieve the data resampling.

Upsampling is a procedure where synthetically generated data points (corresponding to minority class) are injected into the dataset. After this process, the counts of both labels are almost the same. This equalization procedure prevents the model from inclining towards the majority class. Furthermore, the interaction(boundary line)between the target classes remains unaltered. And also, the upsampling mechanism introduces bias into the system because of the additional information.

Downsampling is a mechanism that reduces the count of training samples falling under the majority class. As it helps to even up the counts of target categories. By removing the collected data, we tend to lose so much valuable information.

Sklearn.resample is Scikit learn's function for upsampling/downsampling. From sklearn documentation, the function sklearn.resample, resamples arrays or sparse matrices in a consistent way and the default strategy implements one step of the bootstrapping procedure. In simple terms, sklearn.resample doesn't just generate extra data points to the datasets by magic, it basically creates a random resampling(with/without replacement) of your dataset. This equalization procedure prevents the Machine Learning model from inclining towards the majority class in the dataset.

I. Feature Selection Using Variance Threshold

Variance Threshold is a univariate approach to feature selection. It removes all features whose variance doesn't meet some threshold. By default, it removes all zero-variance features, i.e. features that have the same value in all samples. As an example, suppose that we have a dataset with boolean features, and we want to remove all features that are either one or zero (on or off) in more than 80% of the samples. Boolean features are Bernoulli random variables, and the variance of such variables is given by The below approach removes variable which have more than 80% values are either 0 or 1.

The biggest challenge of Machine Learning is to create models that have robust predictive power by using as few features as possible. But given the massive sizes of today's datasets, it is easy to lose the oversight of which features are important and which ones aren't. That's why there is an entire skill to be learned in the ML field — feature selection. Feature selection is the process of choosing a subset of the most important features while trying to retain as much information as possible.

Variance, as the name suggests, shows the variability in a distribution in a single metric. It shows how spread out the distribution is and shows the average squared distance from the mean. Manually computing variances and thresholding them can be a lot of work. Fortunately, Scikit-learn provides VarianceThreshold estimator which can do all the work for us. Just pass a threshold cut-off and all features below that threshold will be dropped.

J. Polynomial Feature Transformation

Often, the input features for a predictive modeling task interact in unexpected and often nonlinear ways.

These interactions can be identified and modeled by a learning algorithm. Another approach is to engineer new features that expose these interactions and see if they improve model performance. Additionally, transforms like raising input variables to a power can help to better expose the important relationships between input variables and the target variable.

These features are called interaction and polynomial features and allow the use of simpler modeling algorithms as some of the complexity of interpreting the input variables and their relationships is pushed back to the data preparation stage. Sometimes these features can result in improved modeling performance, although at the cost of adding thousands or even millions of additional input variables.

Polynomial features are those features created by raising existing features to an exponent.

For example, if a dataset had one input feature X, then a polynomial feature would be the addition of a new feature (column) where values were calculated by squaring the values in X, e.g. X^2. This process can be repeated for each input variable in the dataset, creating a transformed version of each.

As such, polynomial features are a type of feature engineering, e.g. the creation of new input features based on the existing features.

The "degree" of the polynomial is used to control the number of features added, e.g. a degree of 3 will add two new variables for each input variable. Typically a small degree is used such as 2 or 3. It is also common to add new variables that represent the interaction between features, e.g a new column that represents one variable multiplied by another. This too can be repeated for each input variable creating a new "interaction" variable for each pair of input variables.

A squared or cubed version of an input variable will change the probability distribution, separating the small and large values, a separation that is increased with the size of the exponent.

This separation can help some machine learning algorithms make better predictions and is common for regression predictive modeling tasks and generally tasks that have numerical input variables.

Typically linear algorithms, such as linear regression and logistic regression, respond well to the use of polynomial input variables.

K. Metrics for model evaluation

1) R-Squared value: This value ranges from 0 to 1. Value '1' indicates predictor perfectly accounts for all the variation in Y. Value '0' indicates that predictor 'x' accounts for no variation in 'y'.

1. Regression sum of squares (SSR)

This gives information about how far estimated regression line is from the average of the actual output.

$$\text{Error} = \sum_{i=1}^{n}(\hat{y} - \bar{y})^2$$

2. Sum of Squared error (SSE)

How much the target value varies around the regression line (predicted value).

$$\text{Error} = \sum_{i=1}^{n}(y - \hat{y})^2$$

3. Total sum of squares (SSTO)

This tells how much the data point move around the mean.

$$\text{Error} = \sum_{i=1}^{n}(y - \bar{y})^2$$

$$R^2 = 1 - \frac{SSE}{SSTO}$$

2) Null-Hypothesis and P-value: The p-value for each term tests the null hypothesis that the coefficient is equal to zero (no effect). A low p-value (¡ 0.05) indicates that you can reject the null hypothesis. In other words, a predictor that has a low p-value is likely to be a meaningful addition to your model because changes in the predictor's value are related to changes in the response variable.

Conversely, a larger (insignificant) p-value suggests that changes in the predictor are not associated with changes in the response.

3) Confusion Matrix: A confusion matrix is a technique for summarizing the performance of a classification algorithm. Classification accuracy alone can be misleading if you have an unequal number of observations in each class or if you have more than two classes in the dataset. It gives you insight not only into the errors being made by your classifier but more importantly the types of errors that are being made.

		Prediction outcome		
		p	n	total
actual value	p′	True Positive	False Negative	P′
	n′	False Positive	True Negative	N′
	total	P	N	

we can assign the event row as “positive” and the no-event row as “negative“. We can then assign the event column of predictions as “true” and the no-event as “false“.

This gives us:

“true positive” for correctly predicted event values. “false positive” for incorrectly predicted event values. “true negative” for correctly predicted no-event values. “false negative” for incorrectly predicted no-event values.

III. Problem - Understanding and Modelling

A. Stock Prediction - Time Series Analysis

We start with reading the six datasets with the different time series values for different stocks with the opening, high, low and closing prices and the volume of the stocks. We are required to analyse the given data and get meaningful insights from the same using different techniques taught in the course or newer techniques. We try to use raw data exploration techniques along with creating different graphs and analyzing the outputs.

We take a look at the different datasets available and find out that the organisations Cognizant, HCL, HDFC, Infosys and SBI have six columns namely date, open, high, low, close and volume, while the USD dataset not having the volume column but an adj close column.

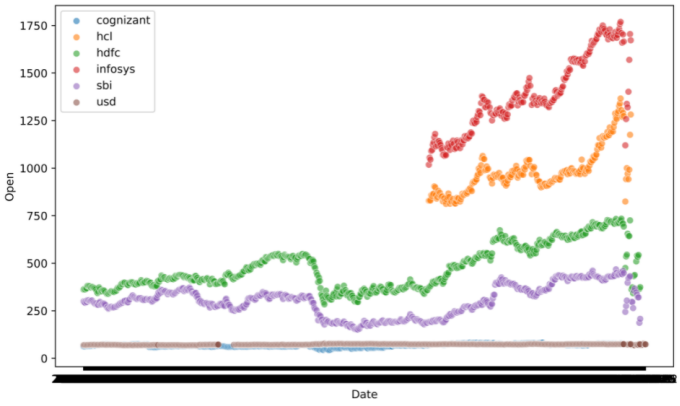


Fig. 3. Time Series: Date vs Other Cols

Starting with the exploratory data analysis, we create a scatterplot date vs (Open, Close, High & Low) for all the

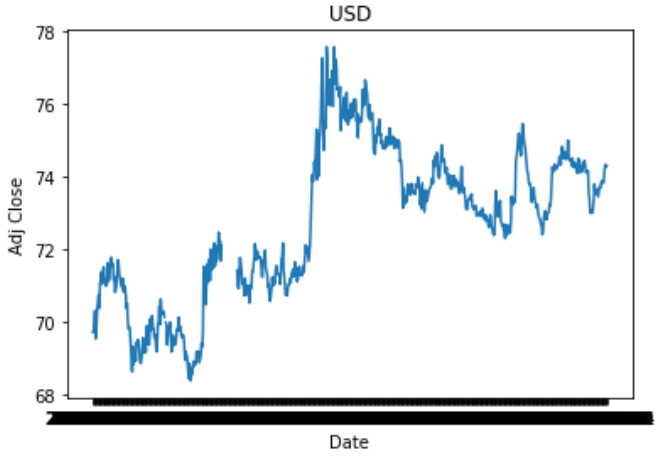


Fig. 4. Time Series: USD — Date vs Adj Close

different datasets to see a general standing of the values. We see that Infosys leads the bunch with the highest opening, closing, high and low stock prices followed by HCL, HDFC, SBI, USD and Cognizant in the decreasing manner. But this does not help a new customer interested to buy as this does not comment on the relative changes of the stock prices, but only the absolute values. We also plot the closing prices for the stocks and observe nice patterns here. For cognizant we almost see a flat average with the values dipping and rising at intervals but the average over a period being the same. For HCL and Infosys we see a steep increasing graph with negligible valleys and a constantly increasing graph overall. We see a similar trend with HDFC and SBI, where we see an overall increase in the share price over time but locally we see peaks and valleys with wide variation.

Moving on, we plot the variation in the Adj Close prices with time for USD and see quite a graph with many local peaks and valleys. Thus if carefully planned, could lead to high profits or else turn into complete disaster. Similarly we plot the Volume vs data for the five organisations and observe very high volumes at some distinct peaks, which could be the times when the stock prices were on a very steep decline.

Moving on to Feature Generation, we create a new feature as the percent change between the close and open values for that date and plot it as a scatter plot. We see that SBI has the highest mean thus on average SBI stocks fall during the day and finally lead to lower close while cognizant is the most negative meaning that its stocks are usually to rise during the open hours. COmputing the standard deviations we see that USD is most certain with the lowest std while the highest variability in SBI, Following this we do a similar analysis with High and low values during the day, the results of which are posted below in the figure.

Following this we go to to compute the moving average on the closing stock prices with the interval of average being 10, 20 and 50 days. We see that the larger intervals tend to normalize the changes over shorter periods and thus shows the overall trend while the shorter ones tend to keep into account

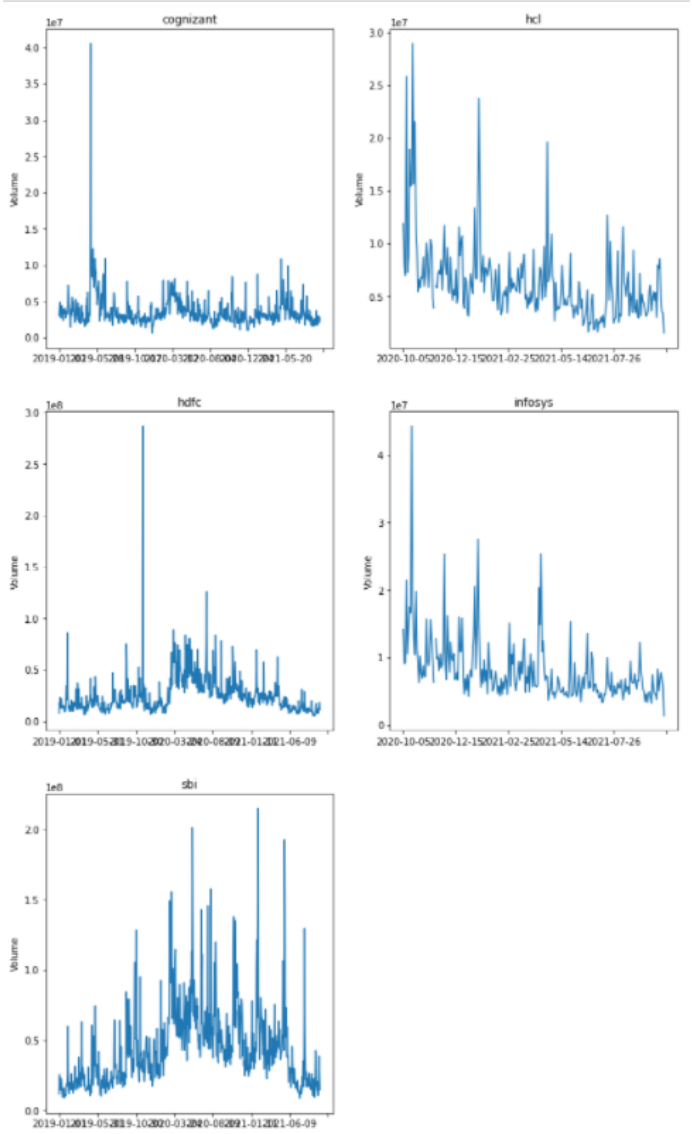


Fig. 5. Time Series: Date vs Vol

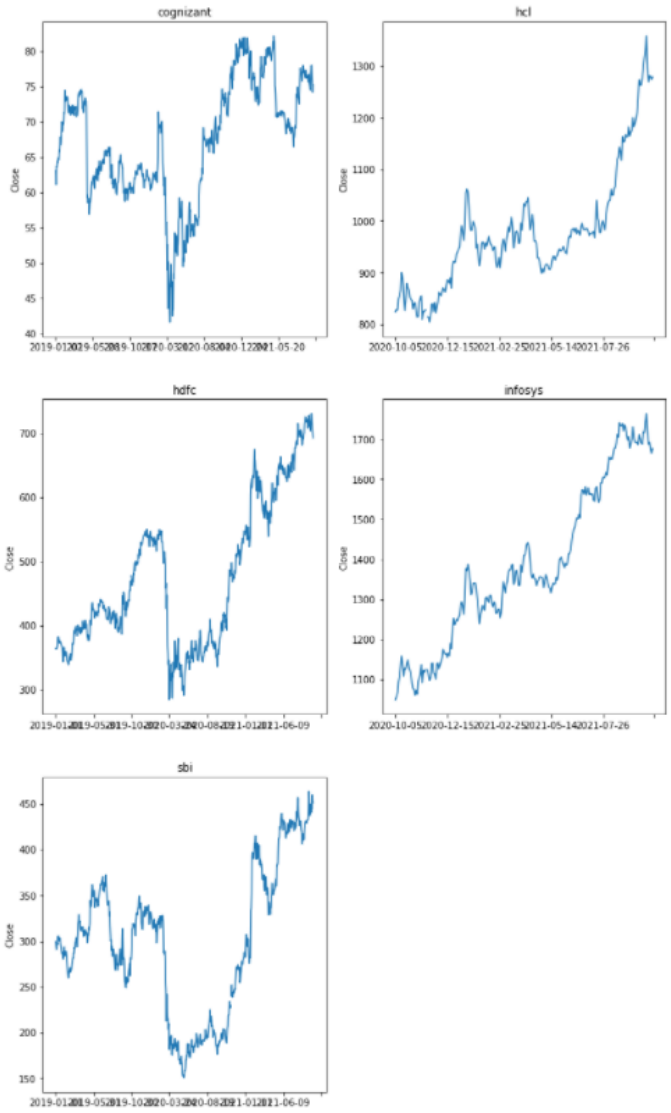


Fig. 6. Time Series: Date vs Close

the variability locally. We now get to the most important part of the stock analysis, computing the daily return of the stock using the percentage change of the closing price. As expected we see both positive and negative values, thus expected losses and profits, with very high peaks and low valleys. Following this we create distplots and analyze the mean and standard deviation of the daily return values and observe that all the stocks have a positive return averaged over the whole period, however small it might be.

We then move on to compute the correlation between the different percentage change values and create pairplots and pairgrids for the same. We observe high correlation between HDFC and SBI stocks which could be as both are banks they might be affected from same causes. We also observe high correlation between HCL and Infosys which might be due to both being from tech/software backgrounds. These high correlation values suggest that the variability (increase or decrease) in the stock prices for the corresponding entities follows a similar trend.

Finally, we try to model the risk and expected return of the entities, with risk modelled as the standard deviation of the corresponding daily return values (higher the variability, higher is the risk) and expected return being its mean. We see that SBI has the highest expected return with the highest risk associated, while USD has the lowest associated risk and also the lowest returns. Particularly, we see that betting on Cognizant is a bad deal as the associated risk is high with the expected return being low where infosys can be a better bet with similar risk and much higher expected returns.

B. Assignment 1: Linear Regression Updates

The assignment 1 revolves around cancer data analysis with the aim to model two continuous variables the average incidence and the deaths. We are given Poverty, Income

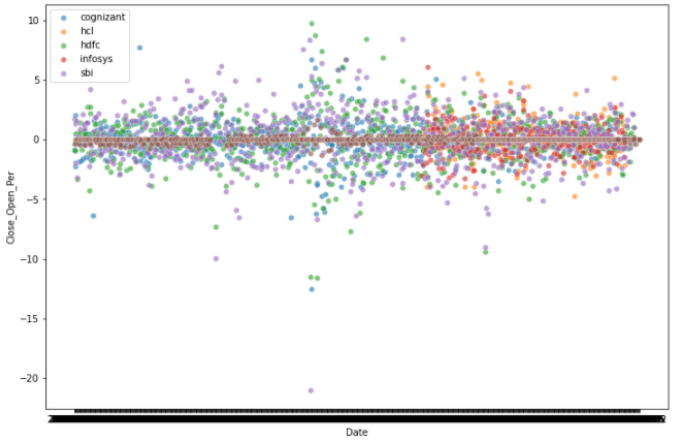


Fig. 7. Time Series: Feature Gen Close minus Open Percentage

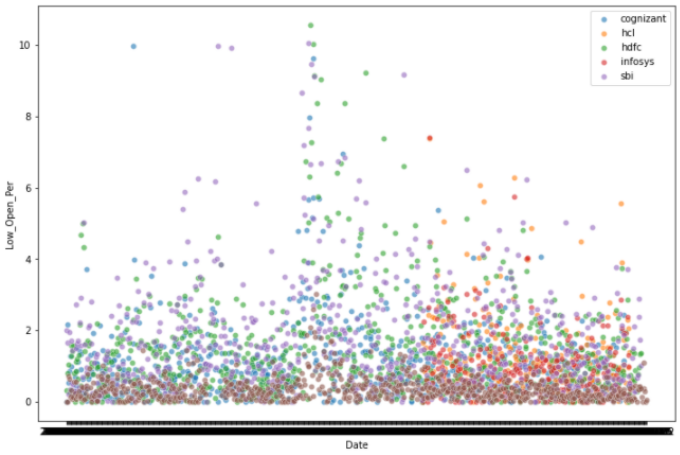


Fig. 8. Time Series: Feature Gen Low minus Open Percentage

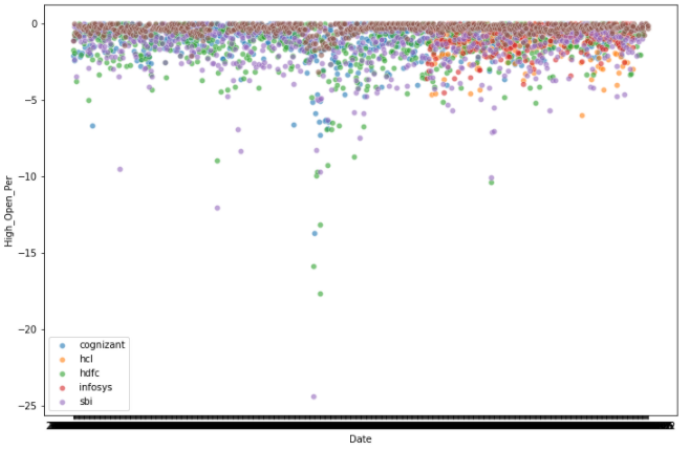


Fig. 9. Time Series: Feature Gen High minus Open Percentage

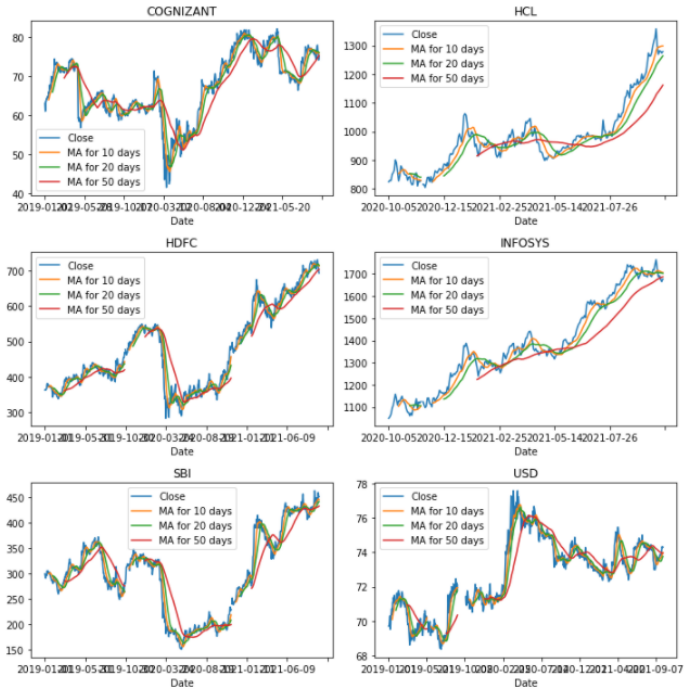


Fig. 10. Time Series: Moving Average

and Insurance data as metrics for socioeconomic status and Incidence rate, Average incidence, mortality rate & average deaths as the dependent variables on which the hypothesis is to be tested, for different areas in the United States identified by the FIPS number.

We have information about the state, the name of the area, poverty - total people, males, females below the poverty line; Income - median income of all people, white, black, native americans, hispanic, asians; Insurance - All people with and without insurance, including males and females with and without insurance. We have a total of 3134 data points.

1) Pre-Processing: We start with checking the effect of social status on the median income as this is the only social data, thus we try to create a line-plot for the median incomes (Figure 1) for the different communities in different states, and thus see a trend wherein the Asians usually are seen to have a higher income, while the income for the blacks is on the lower side. We can see that the different social group have different mean incomes for different states, hence if we can prove that the median income is a valid factor determining the Avg Ann Incidence or Deaths, then we can also assume that social status would also be a valid factor.

We then check on the number of null values in each of the columns, and we find out that the columns with Median Income values for the social groups like Native American, Asian, Black, Hispanic have too many missing values, ranging from 20 % to 55 % thus we proceed to delete all these columns. We then check for columns with values that are not numeric, and thus find out that the columns Incidence Rate, Avg Ann Incidence, recent trend, mortality rate and Avg Ann deaths has values that are not purely numeric.

A very important observation is that all the independent columns are not normalized by the population and we also do not have population data, thus it is better to train our model

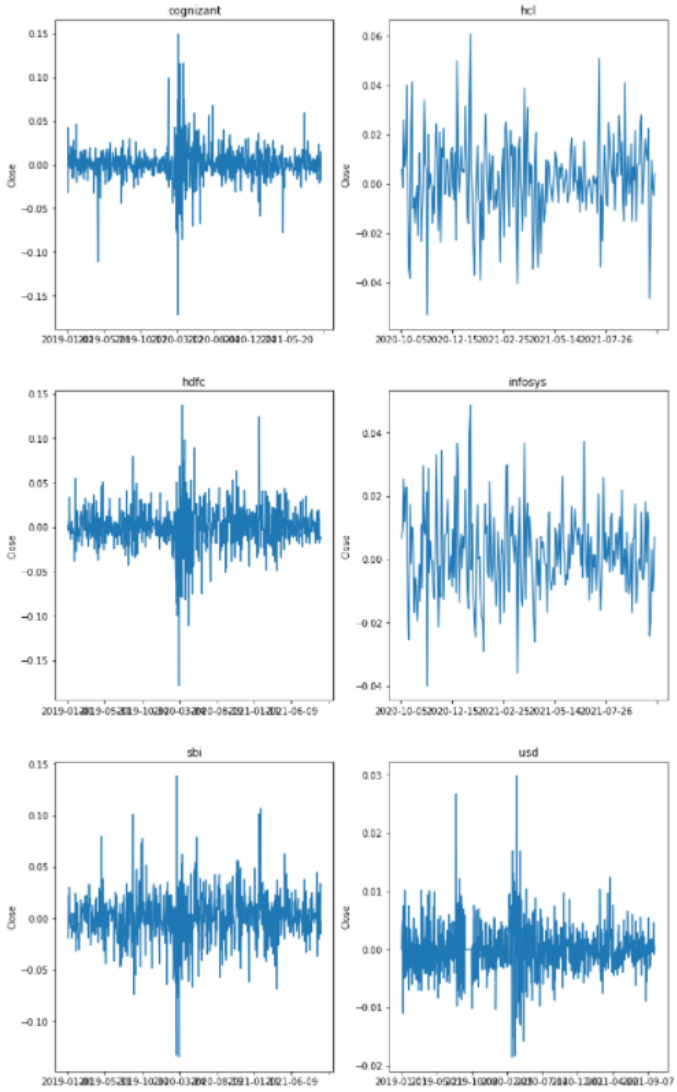


Fig. 11. Time Series: Daily Return Lineplot

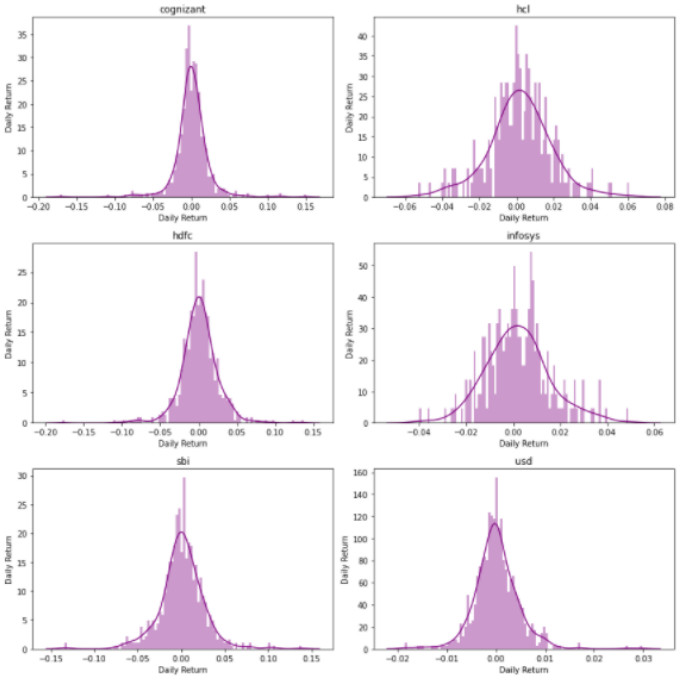


Fig. 12. Time Series: Daily Return distplot

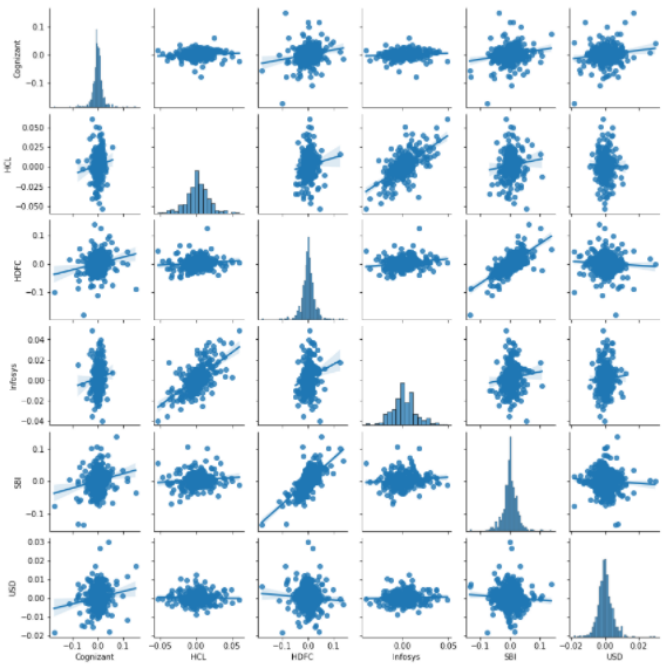


Fig. 13. Time Series: Daily Return Pairplot

on Avg Ann incidence and Avg Ann deaths, rather than using Incidence rate and Mortality rate data as these are just the normalized versions of the average values. Thus we drop these two dependent columns.

On evaluating the data in Avg Ann Deaths column, we see 325 data-points with an asterisk. It represents the data that has been suppressed due to confidentiality when fewer than 16 cases were reported. So we need to deal with this missing data, let's evaluate all the other columns and then proceed to treat the missing data. On evaluating Avg Ann incidence column we see three types of data other than numeric data including '3 or fewer', '_', '__'. which is not much compared to the total data. We replace the '3 or fewer' rows with 3 and the other with null values.

We also use feature extraction to create two new columns corresponding to rising and falling of the recent trend and drop the recent trend column. Now we have some of the data that is null and we need to treat it before training our model which can be done in some ways- One can be to remove the rows (data points) corresponding to the NULL values, other can be imputing the data with the median of the data grouping using the state data. One other method can be using a model that can handle missing data, and this is the method we are going to use in this paper.

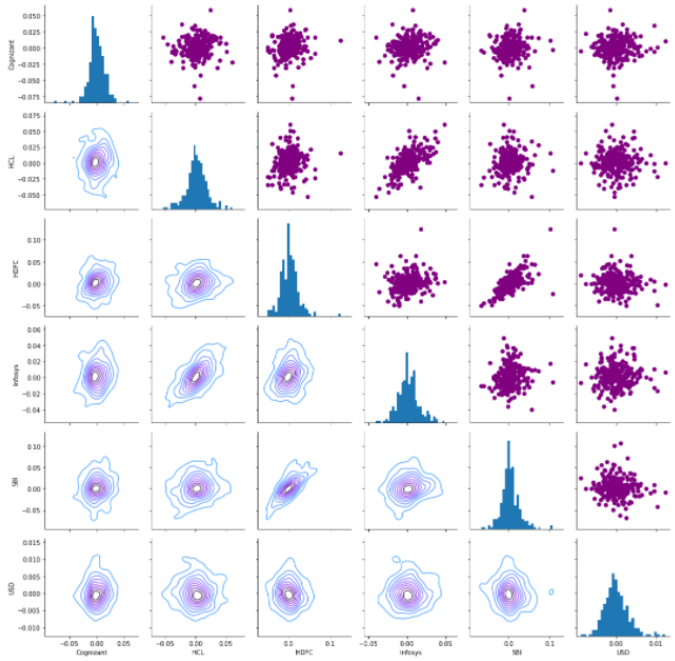


Fig. 14. Time Series: Daily Return Pairgrid

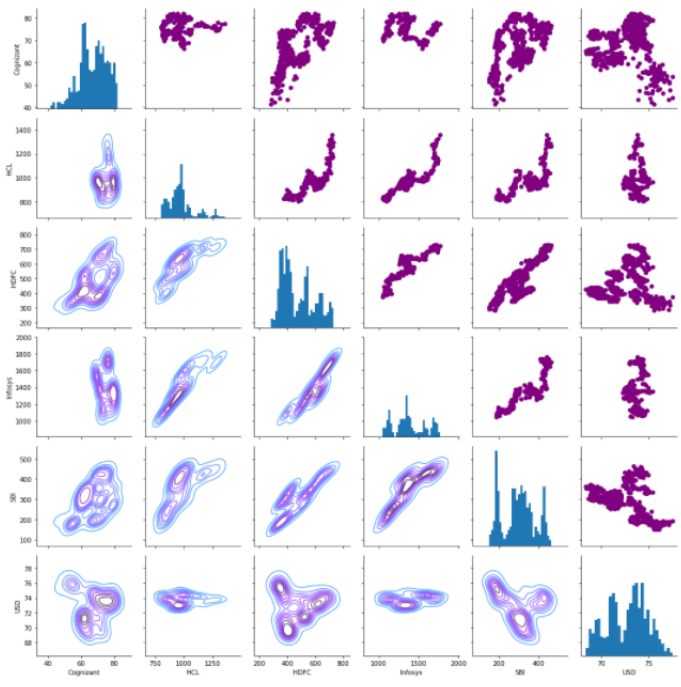


Fig. 15. Time Series: Closing Df Pairplot

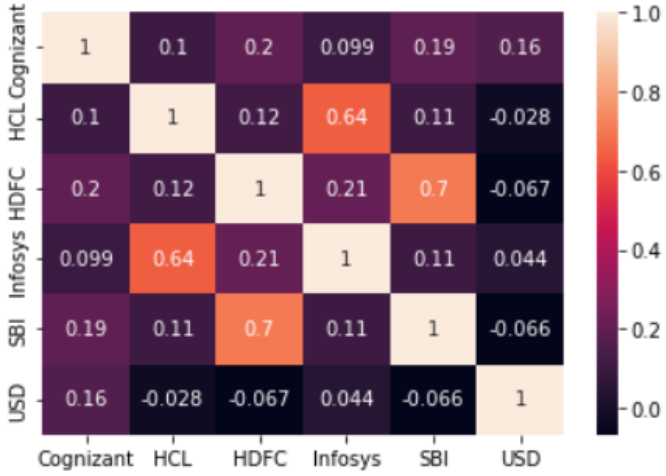


Fig. 16. Time Series: Daily Return Correlation Plot

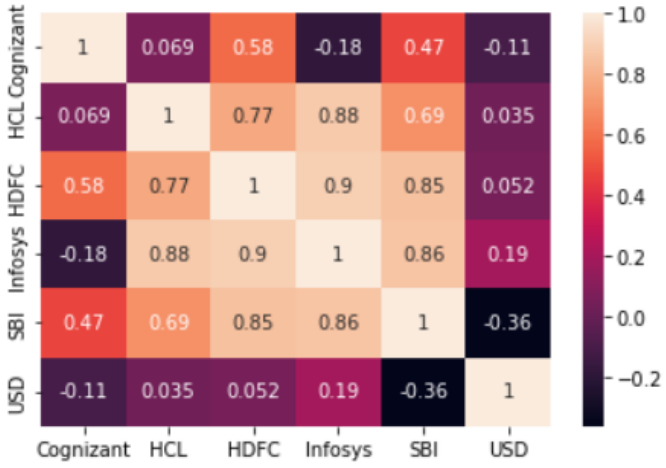


Fig. 17. Time Series: Closing Df Correlation Plot

2) Visualization: We start by plotting scatter plots for Avg Ann Incidence vs Avg Ann Deaths (Figure 2) and see that there is very high correlation between these, hence testing on one of these would lead to similar results on the second, this assumption can be made. We then follow by creating a pair-plot between all the poverty columns and the median income column. We also see a high correlation (as backed up by the correlation heat-map) between the poverty columns i.e. All poverty, M poverty and F Poverty (Figure 3), thus this could lead to the problem of high multicollinearity and thus we need to take care of this, thus we remove the M poverty and F poverty columns. We then repeat the same process with the insurance columns and then using similar results proceed to dropping M with, M without, F with, F without columns. We then proceed with creating pair-plots (Figure 4) for the remaining All Poverty, Med Income, All with, All without columns and see that the columns All poverty and Insurance columns have high correlation (Figures 5, 6, 7, 8), but we will deal with it later. We then look at the scatter plots of the independent columns with the dependent columns.

3) Statistical Linear Regression Modelling & Updates: We then model a statistical linear regression model with the parameters treated as random variables and get a good adjusted R squared score of 0.862. We then check for the multi-collinearity in the data using the VIF and see that one of

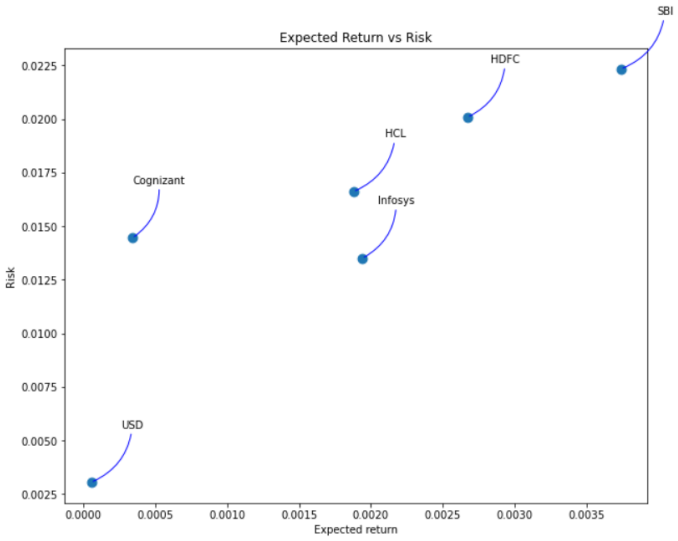


Fig. 18. Time Series: Expected return vs Risk

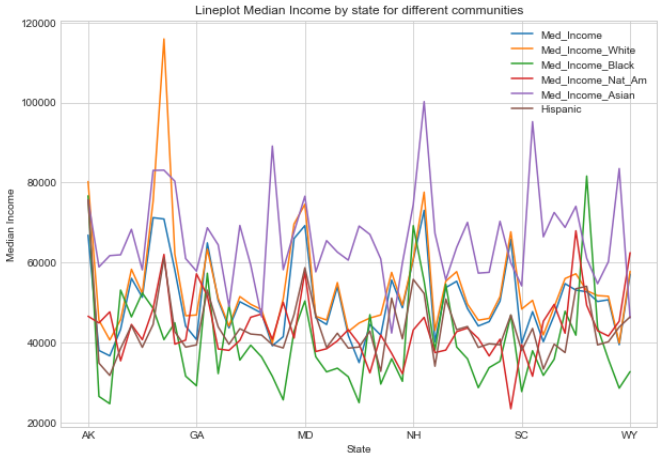


Fig. 19. Assignment 1: Line-plot median income by state

the columns has VIF of around 15 and thus needs to be dropped. Thus on dropping the feature we get a model which is more stable and with an adjusted R squared value of 0.86. Following this we use the IQR (Inter Quartile Range) method for outlier removal and thus analyse outliers for different variables using boxplots for each variable, We observe large number of outliers for ALL Povery, All With and All Without which is shown in the figure below. After the outlier removal by computing the 75^{th} and 25^{th} percentile and finally get an improved adjusted R squared of 0.927 which is a huge improvement.

Using the QQ plots, comparing QQ plots for the Old and New (Outlier free) data, we see that the new data is much well suited for the t distribution, which was way off especially at the extremes in the original data. Thus we get a data that is outlier free and hence our model more robust. We also see a much better boxplot after the removal, with lesser outliers. Finally we also check for the residual scatter plot which is also

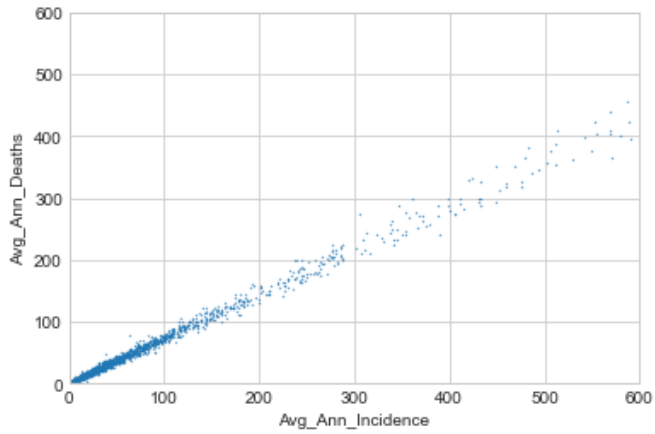


Fig. 20. Assignment 1: Scatter-Plot

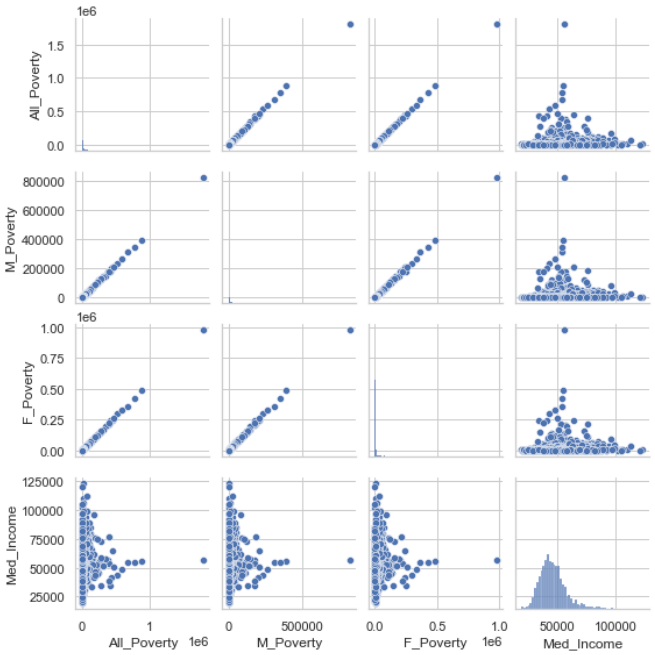


Fig. 21. Assignment 1: Pair-plot

better (closer to the zero line) which is better as the regression line is fit better to our processed data and hence shows lower errors (residuals).

C. Assignment 2: Logistic Regression

We are presented with a problem involving shipwreck data of the Titanic. On April 15, 1912, during her maiden voyage, the widely considered "unsinkable" RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren't enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

We are given attributes of the passengers of the Titanic including Name, Sex, Age, Passenger class, Number of Siblings/ Spouse onboard, Parents and children onboard, Ticket number,

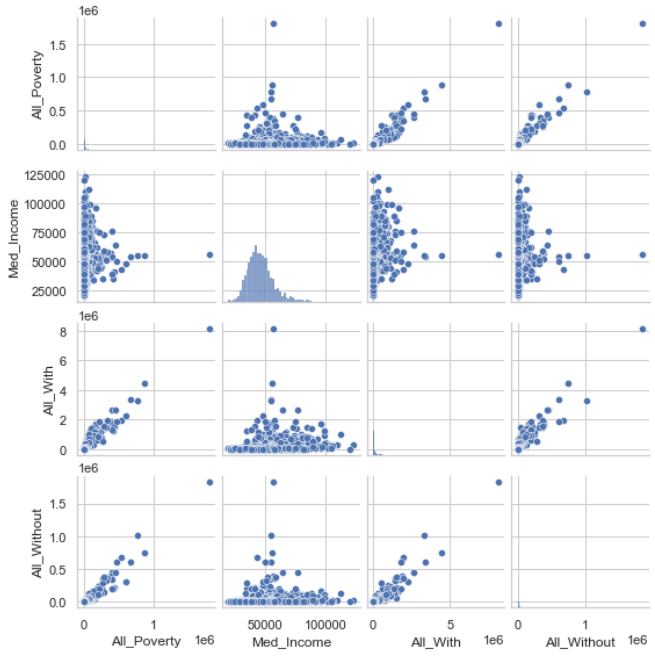


Fig. 22. Assignment 1: Pair-plot

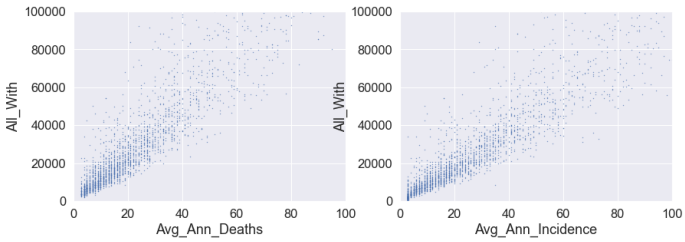


Fig. 23. Assignment 1: Scatter-plot

Fare, Cabin, and the port where they embarked and the final dependent column being whether the particular passenger survived or not. We need to build a predictive model that answers the question: "what sorts of people are more likely to survive?" using the given passenger data, and discuss any insights and observations.

1) Data Pre-processing & Feature Generation: We start with reading the data to a pandas dataframe. We observe a total of 891 data points, with 12 columns in total, including the different features related to the person onboard. On visualizing the distributions of the people survived we see that around 38% of the total passengers survived the wreck. (Figure 1) We then move on to checking for the null values in the data and find that we have three columns Age, Cabin and Embarked with missing data.

Handling Age, we see that there are a total of 177 missing values that we need to take care of. We tend to check the correlation of Age with the other columns and find out that Pclass has the highest absolute correlation of about 0.37, thus we group age by Pclass and Sex and impute the missing values by the median computed for each group.

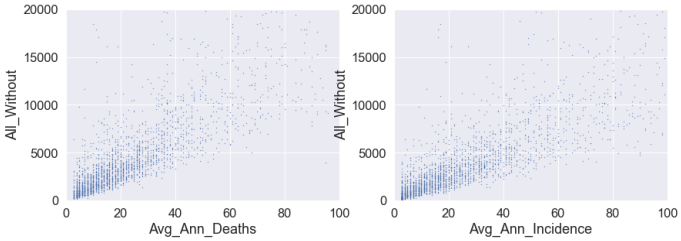


Fig. 24. Assignment 1: Scatter-plot

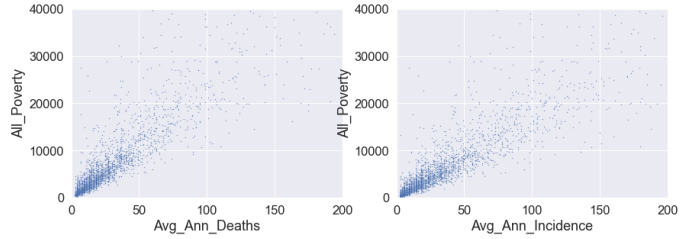


Fig. 25. Assignment 1: Scatter-plot

We then check for the missing values in Embarked and see only two values are missing. Checking Cabin we see that most of the values (77%) are missing. Thus we need to create a new class M (missing), and also extract the cabin class from the cabin column which includes cabin class followed by number. On visualizing the final Cabin column we see that most of the people in the missing class belong to the Pclass 3 and have little chance of survival. We see a higher chance of survival in most of the other cabin classes, with the least being in cabin A and the highest chance being in the cabin class B. We also observe that most of the people in the classes C, E, G, D, A, B belong to the Pclass 1, and have high chances of survival, thus indicating that Pclass might be a factor resulting in higher chances of survival.

On exploring the feature Embarked, we see that it contains three different classes S, C and Q. while most of the people embarked from the S class and majority of them being from Pclass 3. The Passengers from C look to be lucky as a good proportion of them survived. The reason for this maybe the rescue of all the Pclass1 and Pclass2 Passengers. The Embark S looks to the port from where majority of the rich people boarded. Still the chances for survival is low here, that is because many passengers from Pclass3 around 81% didn't survive. Port Q had almost 95% of the passengers were from Pclass3.

We move on to feature generation, we start with checking that many people had the same Ticket numbers, thus we create a new feature representing the number of passengers having the same ticket number, which might be a representation of what was the size of the group in which they were travelling. On careful observation we see that different ticket frequency has different rates of survival with the highest being for the group of two and the chance for groups of more than 4 being very less.

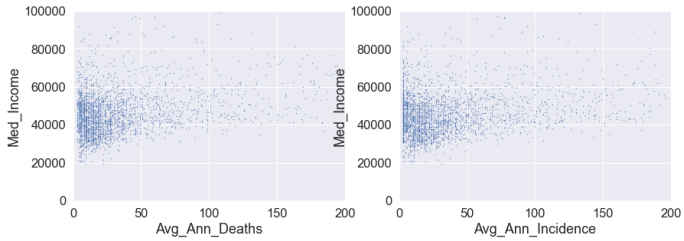


Fig. 26. Assignment 1: Scatter-plot

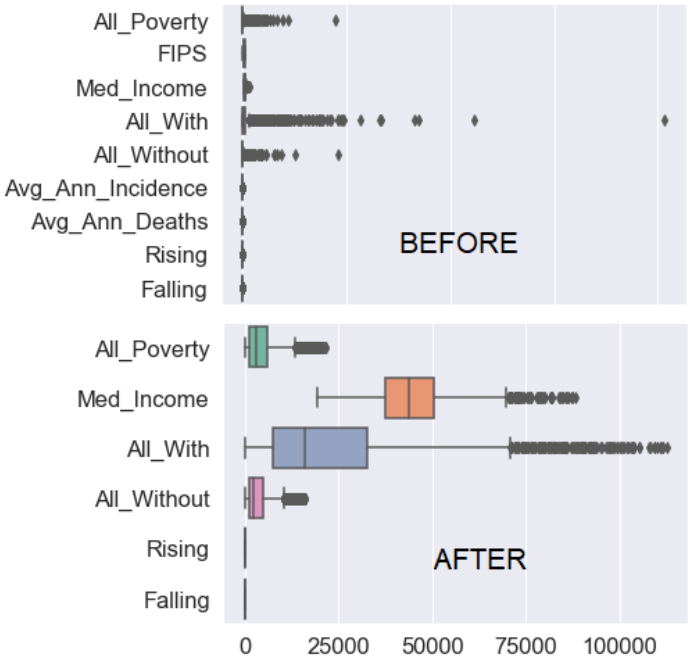


Fig. 27. Ass 1: Boxplot before vs after

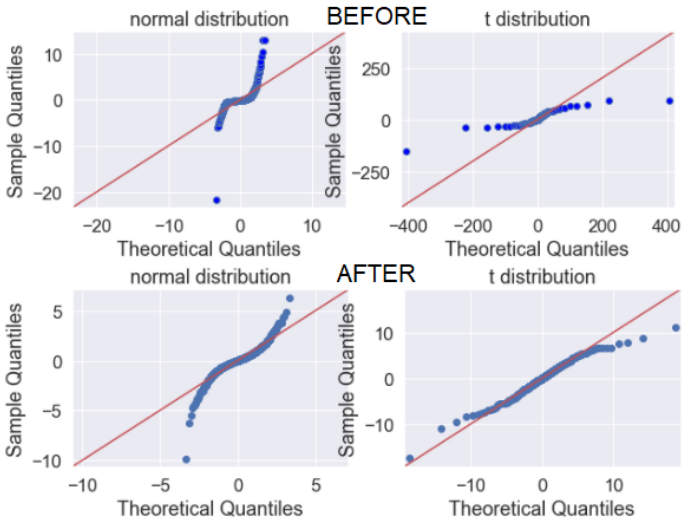


Fig. 28. Ass 1: QQ plot before vs after

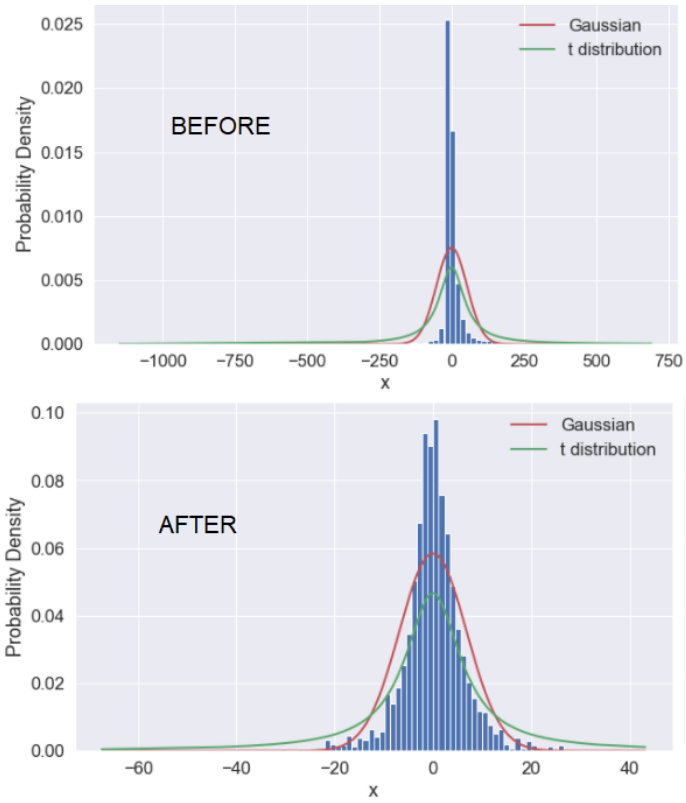


Fig. 29. Ass 1: Residual Plot before vs after

Following that we extract the title for each of the passengers from the Name column and find out that people with different titles, had different chances of getting away. We create three categories with the largest being Mr, then clubbing Miss, Mrs and Miss into one, followed by Master and the others grouped into one. We can see that most of the people are form Mr group where the probability is low while the probability of Miss, Mrs, Ms and Master is higher for survival. We have very less people belonging to other classes. We also create a new feature followed from the title classes as Married wherein every passenger with Mrs class is termed as married and we see that the probability of survival for the married class is high.

We created a Child feature for people below 18 years of age, now we can see that children have a higher chance of getting out (Figure 6). We also create a new feature Family size, which is essentially No. of Parents + Children + Siblings + Spouse + 1, post which we create three different categories for the family size class as Alone, Small and Big. We see that people with small family size have the highest chance of Surviving, and the ones with a big family, which the lease chance.

D. Visualization

1) Visualization: We start by creating countplots for the categorical variables with hue as the target columns, to check for the distribution of each of the categories in the classes, plus commenting on whether the different categories have different survival probabilities. Starting with Pclass, We see

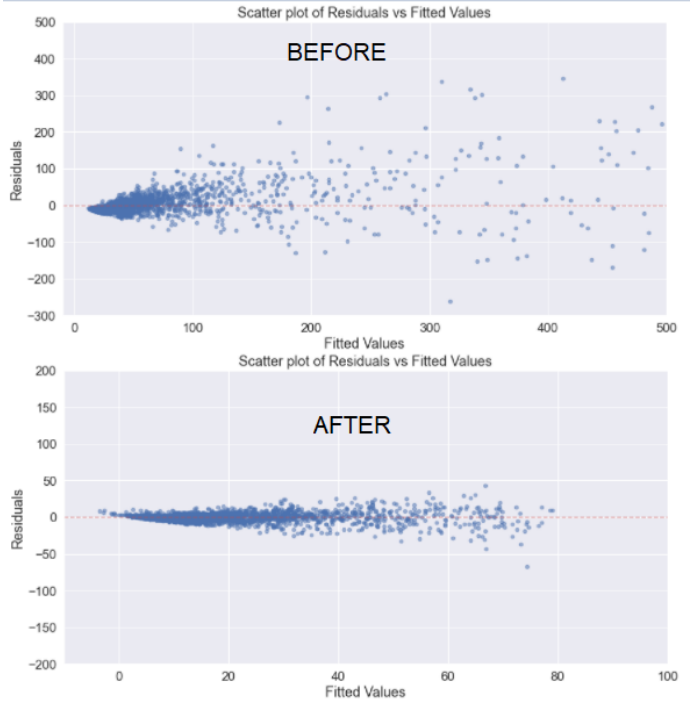


Fig. 30. Ass 1: Residual Scatter before vs after

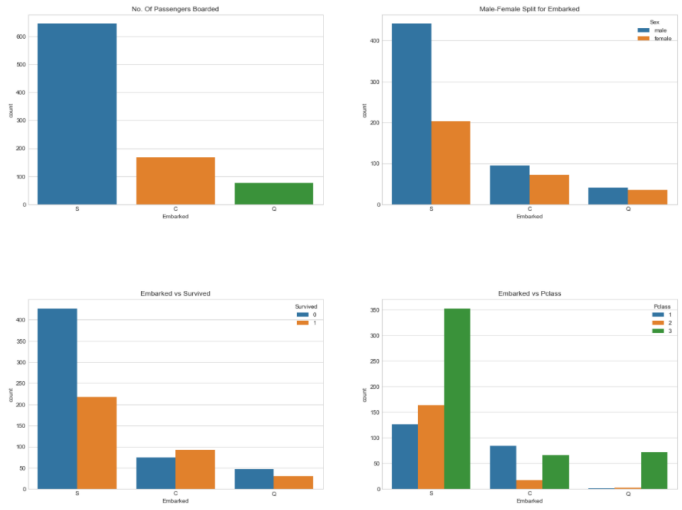


Fig. 31. Ass 2: Embarked

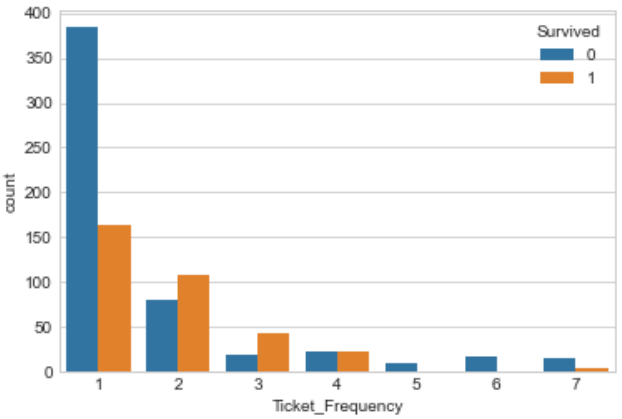


Fig. 32. Ass 2: Count-plot for Ticket Frequency

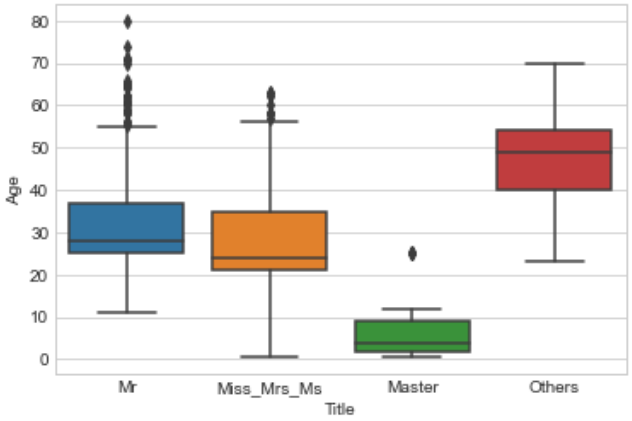


Fig. 33. Ass 2: Boxplot of various titles

that the people belonging to the Pclass 1 have a higher chance of survival, which keeps on decreasing as we go down the classes. Following which we explore Sex, wherein we observe Proportion of male and female: 2/3 vs 1/3. Male is much less likely to survive, with only 20% chance of survival. For females, ¿70% chance of survival. Obviously, Sex is an important feature to predict survival.

We then explore Age , which is not a categorical variable, thus we create histogram and boxplots for the same, resulting in insights such as passengers are mainly aged 20-40 and younger passengers tends to survive. on exploring SibSp we see that most of the passengers are travelling alone and the ones with 1 sibling/spouse are more likely to survive. Visualizing Parch we see that ¿70% passengers travel without parents/children, wherein passengers travelling with parents/children are more likely to survive than those not. Creating distplots and boxplots for Fare, we see that there are people paying too much for their tickets (outliers) and we can see that money gets you a better chance of survival as evident from the graphs.

Finally as part of postprocessing, we convert the age variable into bins of 5 and also convert the Fare variable into bins of 4. Finally we create dummies for the categorical variables so that they are meaningfull to the machine.

2) Statistical Logistic Regression Modelling & Updates: We use the Statsmodels library in python to model the Logistic regression as it gives us the added benefits of getting the significance values of the regression coefficients. Once we have processed all the independent variables we feed them into the statsmodels logit formula, and then proceed to analyze the summary. We get a Log-Likelihood value of -350, with the method being maximum likelihood estimation. We see many

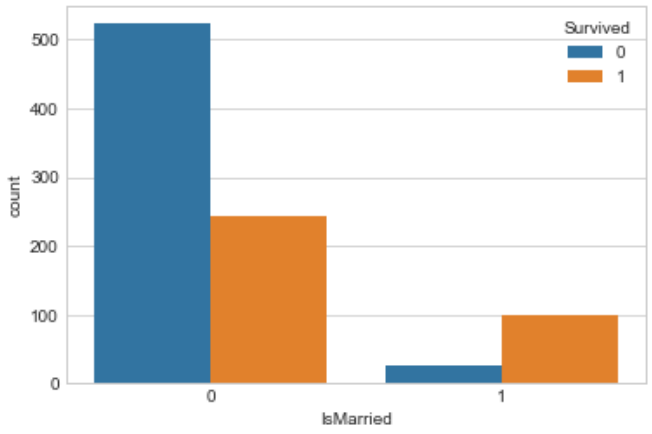


Fig. 34. Ass 2: Countplot - Married or Not

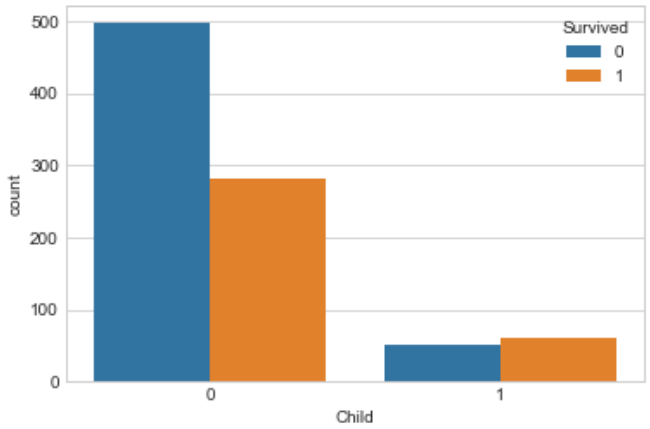


Fig. 35. Ass 2: Countplot - Child or Not

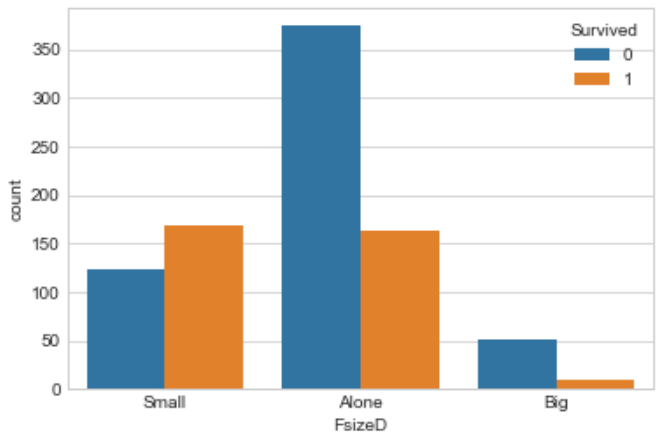


Fig. 36. Ass 2: Countplot for Family Size

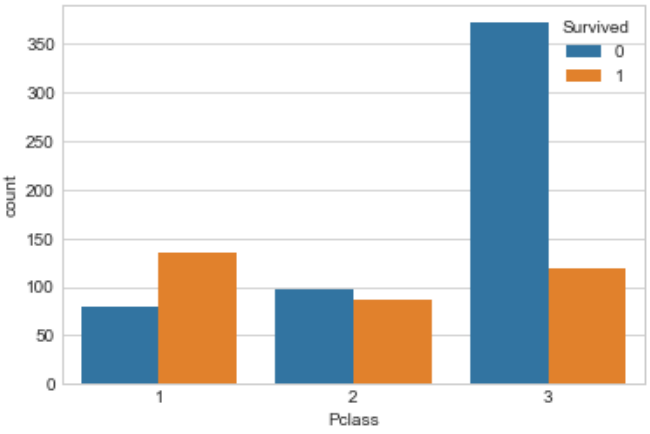


Fig. 37. Ass 2: Countplot for Passenger Class

coefficients having p values of greater than 0.05 (corresponding to 95% significance), suggesting that the coefficients are not statistically significant. We see a pseudo R squared value of 0.4089 and an accuracy value of 0.79 at the threshold of 0.41.

We now add a new feature called AgeClass which is the multiplication of Pclass and Age band, hence adding the notion of polynomial features to the setting. As observed we see that this is a class imbalance problem thus we try our hands on upsampling and downsampling using sklearn's resample. Thus from the original 891 samples with 38 % as the positive, we get 1098 samples for the upsampled dataset and 684 for the downsampled dataset as this is done using sampling the already available minority class with or without replacement. Training on the upsampled set we get a Pseudo R squared of 0.45 and 0.448 for the downsampled set. This is quite an improvement over the original value of 0.4089, which is achieved using only simple bootstrapping.

E. Assignment 3: Naive Bayes Classifier

We are presented with a problem involving census data of certain given attributes of the people of various countries including Sex, Age, education, Marital Status, Occupation, Relationship, Finalwgt, Race, Captial gain & loss, working hours per week, workclass and their native country. Using all these features, we are to predict whether the given person makes over 50K per year or not, and also identify relations between the features and the target class. We also see that this is a class imbalanced dataset with only 24 percent of the people with salary more than 50K, but as we have already looked at upsampling and downsampling, we won't be using it here, and will resort to newer techniques.

1) Data Pre-processing: Out of the total 15 features, 9 are categorical and 6 are numerical. We then move on to checking for the null values in the data and find that we do not have any NaN values, but three of the columns Workclass, Occupation and native country have '?' marks as some data points, which need to be treated. The first step is to replace these '?' marks with NaN values and then finally imputing them with the mode

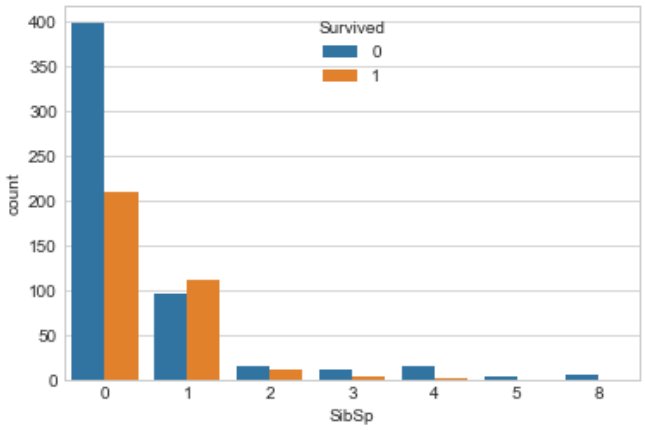


Fig. 38. Ass 2: Countplot - No. of Siblings/Spouse

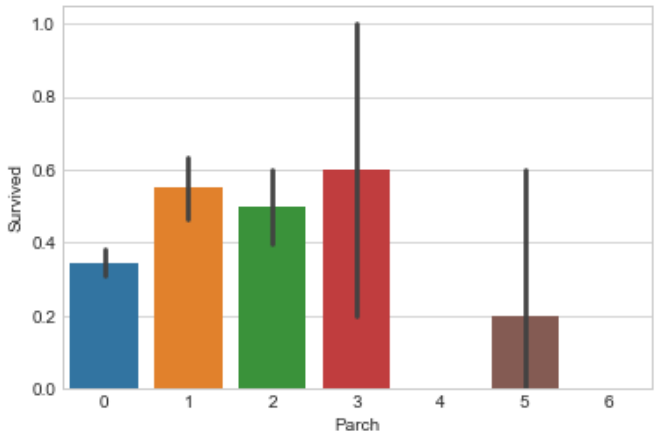


Fig. 39. Ass 2: Probability of Survival vs No. of Parents/Siblings

of each of the columns as the number of null values is very low and thus no special treatment is necessary. We then check for the cardinality of each of the categorical features, that is the number of different unique values each feature can take as a higher number can cause problems, we find that most of the features do not have more than 7 attributes, wherein the native country feature has the highest number.

2) Visualization and Feature Generation: Now we move on to visualizing each of the features one by one. Starting with workclass, we see that the main types are government employees, Private, Self employed, without pay and never worked classes, with the rate of higher income being different for each of the classes, with the highest rate for self employed inclusive, and the least for private. Moving on to education, the main classes are university level, school level, and postgraduate level, with the trend being lesser income for lesser level and the ones with doctorate and professional school enjoying the highest of salaries.

Moving on to marital status, we see that people married and with spouse enjoy the highest of incomes, while all the

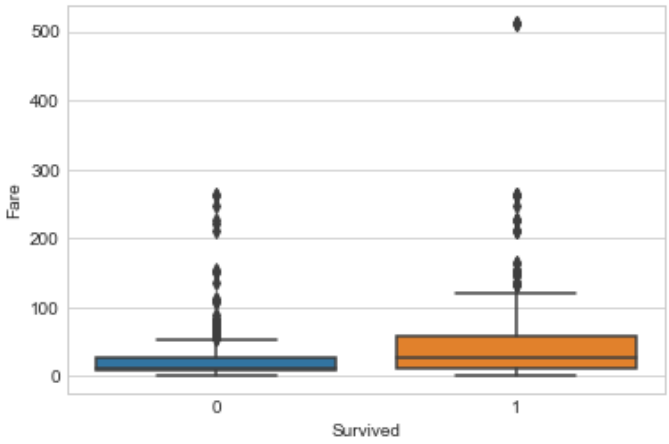


Fig. 40. Ass 2: Boxplot - Fare

Dep. Variable:	Survived	**No. Observations:**	891
Model:	Logit	**Df Residuals:**	866
Method:	MLE	**Df Model:**	24
Date:	Wed, 01 Dec 2021	**Pseudo R-squ.:**	0.4089
Time:	05:55:35	**Log-Likelihood:**	-350.70
converged:	True	**LL-Null:**	-593.33
Covariance Type:	nonrobust	**LLR p-value:**	1.920e-87

BEFORE

Dep. Variable:	Survived	**No. Observations:**	1098
Model:	Logit	**Df Residuals:**	1047
Method:	MLE	**Df Model:**	50
Date:	Thu, 02 Dec 2021	**Pseudo R-squ.:**	0.4493
Time:	03:25:23	**Log-Likelihood:**	-419.10
converged:	False	**LL-Null:**	-761.08
Covariance Type:	nonrobust	**LLR p-value:**	3.453e-112

AFTER

Fig. 41. Ass 2: Model stats before vs after

others having very low possibilities of high income. Checking occupation, we see that the number of classes increases tremendously, with each class enjoying different rates, with Executive managers and professional speciality enjoying the highest incomes. Following this, we move on to relationship feature, and find out that husbands and wifes have the highest probability of having high income, as evident in the society, while single people without family or unmarried do not have such high incomes. Moving on to race, we see a bias towards white people having higher salaries, as compared to the others. A yet disturbing fact is seen when sex is visualized, with males enjoying more income on average than females.

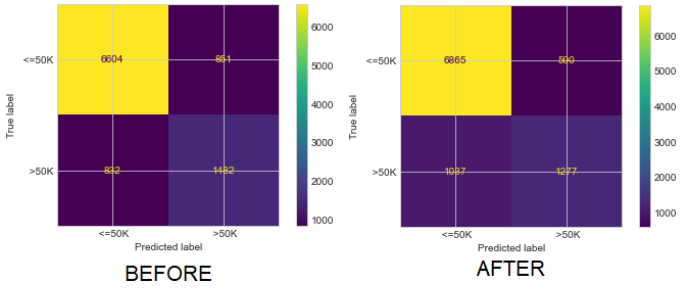


Fig. 42. Ass3: Confusion Matrix Before vs After

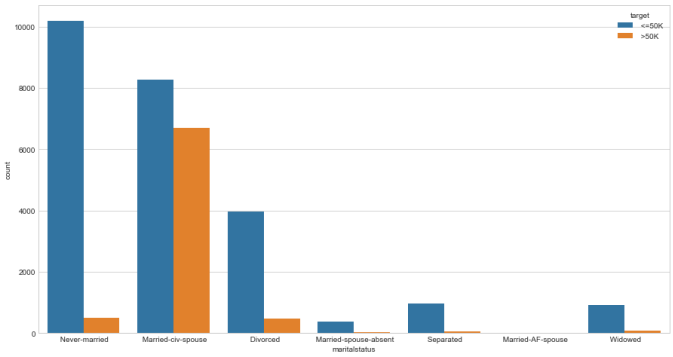


Fig. 43. Ass3: Count-plot Marital Status

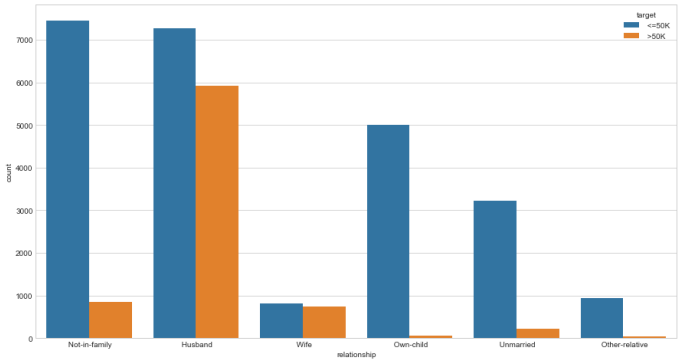


Fig. 44. Ass3: Count-Plot Relationship

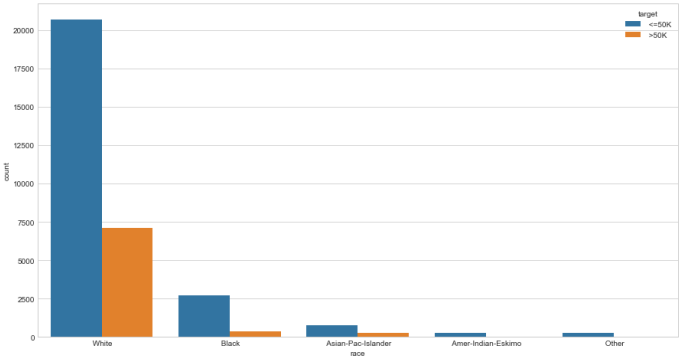


Fig. 45. Ass3: Countplot Race

On exploring Age using histograms and boxplots, we see that people of the higher age in general are seen to have higher incomes, which is explainable as most people are not paid well in the starting of their careers. Using this information, we create a new variable isChild that specifies if the person is less than 24 years of age, and we see that children have zero chance of higher income, from the given data. Post this, we also create a feature senior citizen, which is ticked after being 50 years of age or more. We see that senior citizens have more chance of having higher incomes.

We then move forward to education-num which specifies the number of education levels, and see a trend that higher income is captured by people with higher education levels, owing to which we create a new feature highly educated. Similarly, we also explore hourspw which is number of hours worked per week, and see that the people with higher working hours usually tend to have higher incomes, thus creating a new variable work more.

3) Guassian Naive Bayes Classifier Modelling & Updates: We now move on to modelling using the sklearn library's implementation of the gaussian naive bayes classifier. The first step is to divide the data into a train and a validation set, so that we could test on unseen data. The next step is to scale the features for which we use sklearn's robust scaler. The next task is to get a model with the right set of hyperparameters, which is decided by using randomized search croos validation technique from sklearn's model selection toolkit. The hyperparameter being fine-tuned is var smoothing in the logarithmic scale. Moving on to the updates we create bins for education level and hours worked per week, categorizing them into Low, Medium, High and VeryHigh. We also create a new feature as Occupation Category, with two values as lowskill and highskill, and same with Race category with two values as white or other.

Post this we encode the variables and perform all the pre-processing so as to convert the data into model readable form. Following this we explore a new technique called feature selection using variance threshold as explained in the section 2. This technique helped us to check for the important features and then select them, thus keeping only the most important ones and helping to reduce the variance. Finally we are left with a total of 24 columns after the selection. Post this we train our Naive Bayes model and use random search to find the best hyperparameters and thus get the best performance. We get an updated train and test set accuracy of 0.829 and 0.833 as compared to 0.823 and 0.825 before the updates. We also see improvements in the TP, TN, FP and FNs and get better f1 score of 0.64 as compared to 0.61 initially.

F. Assignment 4 & 5: Tree Methods

We tackle both assignment 4 and 5 in this section as both are based on the same dataset containing various parameters of a car such as buying price, maintenance cost,no. of doors, person capacity, luggage boot size and safety rating with the target being to classify the car as unacceptable, acceptable, good or very good. Thus this a multiclass classification problem with

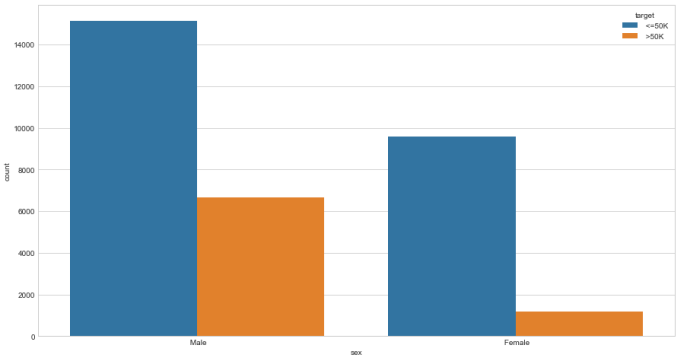


Fig. 46. Ass3: Countplot - Sex

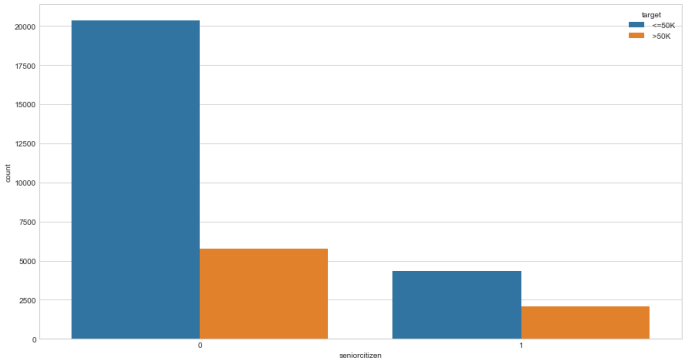


Fig. 48. Ass3: Countplot Senior Citizen

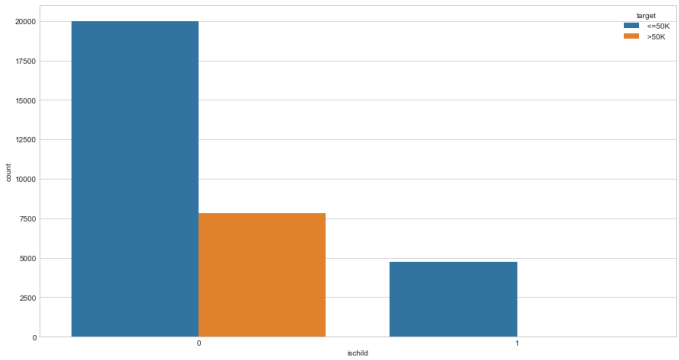


Fig. 47. Ass3: Countplot Children

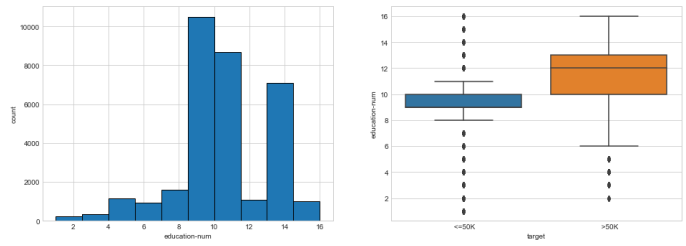


Fig. 49. Ass3: Education Number distribution

an imbalanced dataset with target percentages as 3.8, 4, 22.2 and 70 percent.

1) Data Pre-processing: We start with reading the data to a pandas dataframe. We observe a total of 1728 data points, with 7 columns in total, including the different features related to the car. On visualizing the distributions of the target variable, we see a multi-class imbalanced dataset problem, wherein around 70% of the total cars are classified unacceptable, 22.2% just acceptable, 4% good and 3.8% very good (Figure 1). We see that all the 6 features are categorical and most certainly ordinal. We then move on to checking for the null values in the data and find that we do not have any NaN values. We then go on to checking for any features with high cardinality, that is the number of different unique values each feature can take as a higher number can cause problems, but then find out that most of the features have 3/4 unique classes, and also most of the classes are balanced, hence we conclude that this is good clean data.

2) Exploratory Data Analysis: We start with a univariate analysis, wherein we create histplots for each of the six features with hue as the target column, using the seaborn library. We see that some of the classes in some features have certain target classes missing which is essentially very useful for our decision tree model as out model tries to get pure leaves. For e.g. considering buying, we see that high and very high buying costs only leads to unacceptable and acceptable cars; small luggage boot capacity leads to not being very good; 2 person cars are only seen as unacceptable, very high maintenance cars leads to unnaceptable and acceptable, while high maintenance costs lead to no very good classified cars. We see that low safety cars are unacceptable too.

Moving on to bivariate analysis, wherein we create stripplots for each of the 15 feature pairs with the hue as the target column, using the seaborn library, using jitter and dodge values as true. From the analysis we see a similar trend as the univariate analysis wherein some target classes are missing which is expected. I have provided the whole analysis for further observation.

3) Postprocessing: As the dataset has categorical features and hence they needs to be encoded in the appropriate form. There are two main method of encodig:

1. One hot encoding
2. Label encoding

As we have categorial features that are ordinal in nature i.e that can be ranked (ordered) hence label encoding will solve our purpose.Had there been nominal features we could have preferred one hot encoding. We use category encoders library for this.

Following this we split the data using a 70/30 split, with the final dataset being 1209 examples in training set and 519 examples in crossvalidation set.

4) Decision Tree Modelling & Updates: We start with modelling Decision tree classifier first. We use grid search for finding the best parameters four our decision tree model and checking for model classification accuracy on the cross validation dataset. Starting with class weight hyperparameter,

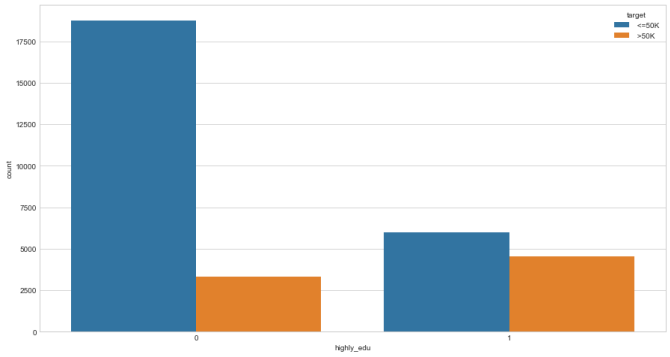


Fig. 50. Ass3: Countplot Highly Educated

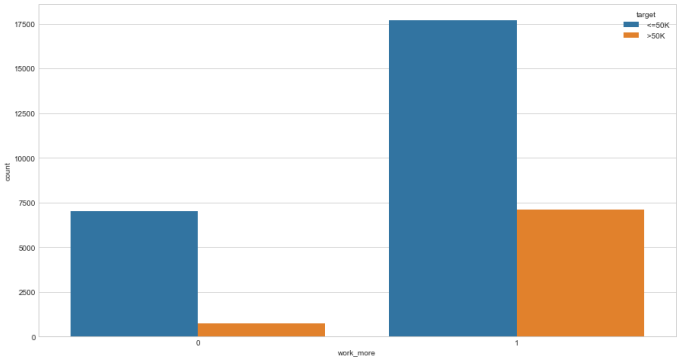


Fig. 52. Ass3: Count-Plot Working More

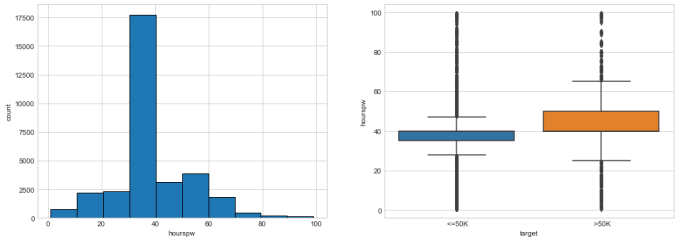


Fig. 51. Ass3: Hours per week distribution

```
accuracy_score on train dataset :  0.8292383292383292
accuracy_score on test dataset :  0.8334527587265841
```

BEFORE

```
accuracy_score on train dataset :  0.8229203229203229
accuracy_score on test dataset :  0.8257754120176067
```

AFTER

Fig. 53. Ass3: Accuracy Before vs After

we keep it as balanced as this automatically calculates the class weights according to their distribution in the train dataset. number of jobs is kept as -1 so that it chooses all the CPU cores available for fitting the model. The scoring metric used by us is accuracy. We use grid search for finding the best parameters by defining a range to check in. The parameters are Max depth, The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min samples split samples. Min samples split: The minimum number of samples required to split an internal node: If int, then consider min samples split as the minimum number. If float, then min samples split is a fraction and ceil(min samples split * n samples) are the minimum number of samples for each split. Min samples leaf: The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min samples leaf training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression. Finally we get the best parameters as max depth 9, min samples leaf 2, min samples split 5.

On initial analysis, without the updates we get train accuracy scores of 0.983 and test accuracy scores of 0.969 for the Decision tree model (Assignment 4). As an update step we create polynomial features using the Polynomial Features method of the sklearn library. We set the degree hyperparameter to 4 so that we could get all the feature combinations upto degree 4. Using this we convert the initial 6 features to a total of 207 features. Finally applying the decision tree classifier on the new dataset we get an improved score of 0.987 on the train set and 0.977 on the test set which is about a 1 percent increase which is very significant. We also try SMOTE for upsampling which results into a conversion to a 3408 sample dataset after upsampling and leads to the test set score of 0.971 which is not an improvement over the previous score but an improvement over the initial vanilla model.

5) Random Forest Modelling & Updates: Then we move on to modelling using the Random Forest classifier which is a bagged version of Decision trees. In this case we use randomized search as this does not search the whole space but tries to get to the optimal using random sampled values of hyperparameters as this model is expensive to train. The hyperparameters being searched are: n estimators = number of trees in the forest, max features = max number of features considered for splitting a node, max depth = max number of levels in each decision tree, min samples split = min number of data points placed in a node before the node is split, min samples leaf = min number of data points allowed in a leaf node, bootstrap = method for sampling data points (with or without replacement).Fitting 3 folds for each of 100 candidates, totalling 300 fits we find the best parameters as 'n estimators': 800, 'min samples split': 3, 'min samples leaf': 1, 'max features': 'auto', 'max depth': None, 'bootstrap': False. Using this we are able to achieve a train set accuracy of 1 and test accuracy of 0.969. We then proceed to check for the feature importance as this is provided by the implementation of random forests in the sklearn library. We see that safety has the highest importance with doors having the least.

Moving on using the same treatement on the Random Forest Model and using Random Search for looking for the best hyperparameters, we get a score of 1 on the train set and

BEFORE	precision	recall	f1-score	support
<=50K	0.89	0.89	0.89	7455
>50K	0.64	0.64	0.64	2314
accuracy			0.83	9769
macro avg	0.76	0.76	0.76	9769
weighted avg	0.83	0.83	0.83	9769

AFTER	precision	recall	f1-score	support
<=50K	0.87	0.92	0.89	7455
>50K	0.68	0.55	0.61	2314
accuracy			0.83	9769
macro avg	0.78	0.74	0.75	9769
weighted avg	0.82	0.83	0.83	9769

Fig. 54. Ass3: Classification Report Before vs After

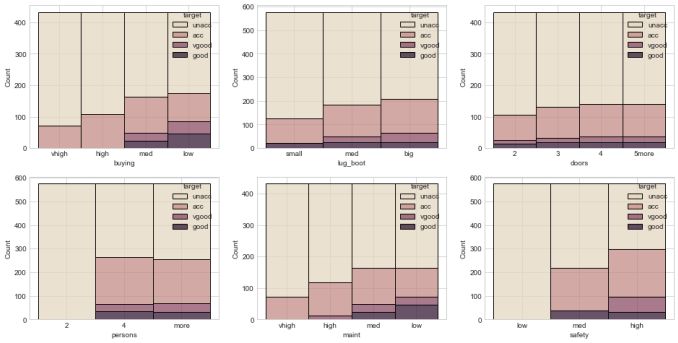


Fig. 55. Ass4&5: Univariate Analysis

Fig. 56. Ass4&5: Multivariate Analysis 1

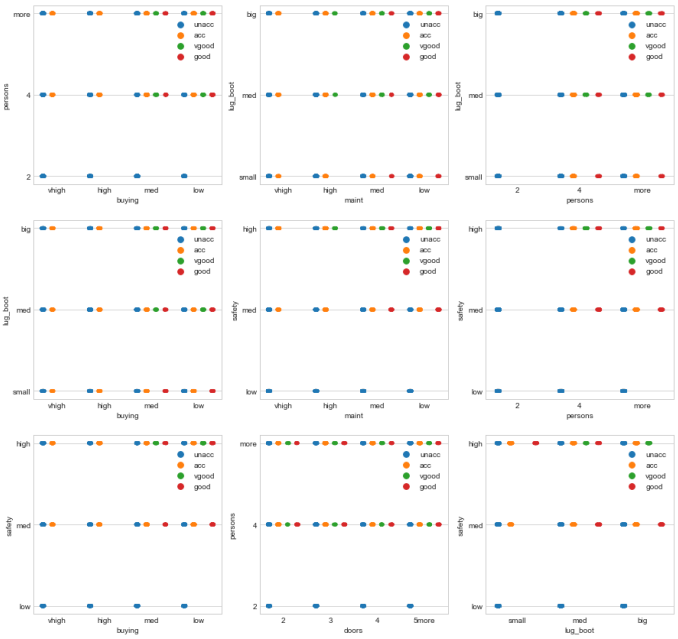


Fig. 57. Ass4&5: Multivariate Analysis 2

0.985 on the test set from an initial score of 1 on the train set and 0.965 on the test set which is an huge improvement of 2 percent. Here we see that our model is overfitting thus we try remove some features in order to reduce variance. Hence using the feature scores, we eliminate the features with an improtance score of less than 0.01. Using this we are able to achieve a train set accuracy of 0.99 and test set accuracy of 0.96. We then try SMOTE for upsampling and hence are able to get further improvement in the accuracy score to 0.9865.

G. Support Vector Machines

Pulsars are a rare type of Neutron star that produce radio emission detectable here on Earth. They are of considerable scientific interest as probes of space-time, the inter-stellar medium, and states of matter. Classification algorithms in particular are being adopted, which treat the data sets as binary classification problems. Here the legitimate pulsar examples form minority positive class and spurious examples form the majority negative class.

The data set shared here contains 11376 negative examples caused by RFI/noise, and 1,152 real pulsar examples. Each row lists the variables first, and the class label is the final entry. The class labels used are 0 (negative) and 1 (positive). Each candidate is described by 8 continuous variables, and a single class variable. The first four are simple statistics obtained from the integrated pulse profile. The remaining four variables are similarly obtained from the DM-SNR curve . These are summarised below:

1. Mean of the integrated profile.
2. Standard deviation of the integrated profile.
3. Excess kurtosis of the integrated profile.
4. Skewness of the integrated profile.
5. Mean of the DM-SNR curve.
6. Standard deviation of the DM-SNR curve.
7. Excess kurtosis of the DM-SNR curve.
8. Skewness of the DM-SNR curve.
9. Class

1) Data Pre-processing: We start with reading the data to a pandas dataframe. We observe a total of 12528 data points, with 9 columns in total, On visualizing the distributions of the

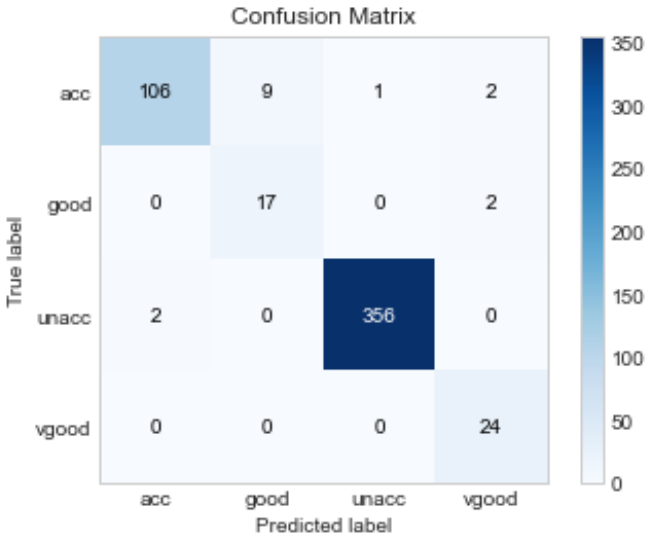


Fig. 58. Ass4&5: Confusion Matrix for Decision Tree

Fig. 59. Ass4&5: Visualization of the decision Tree (look into the codefile for bigger picture)

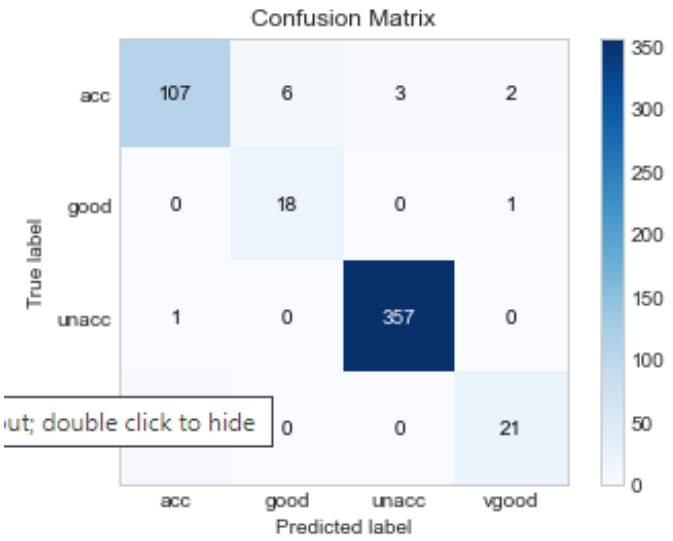


Fig. 60. Ass4&5: Confusion Matrix for Random Forest

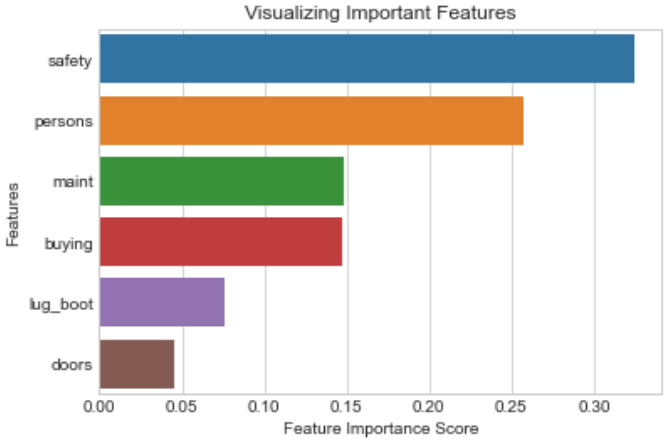


Fig. 61. Ass4&5: Feature Importance chart (higher is better)

target variable, we see an imbalanced dataset problem, wherein around 9.2% of the total stars cars are classified as pulsars, 90.8% are not (Figure 1). We see that all the 6 features are numerical. We then move on to checking for the null values in the data and find that three columns have null values in both the train and test datasets, which are Excess kurtosis of the integrated profile, Standard deviation of the DM-SNR curve and Skewness of the DM-SNR curve. Thus, we need to treat these. Hence we compute the median values for each of these columns and impute the missing values in both the training and test sets as these median values, as median values are much less likely to be affected by outliers.

2) Exploratory Data Analysis: We start with a univariate analysis, wherein we create histplots for each of the eight features with hue as the target column, using the seaborn library. We see that for features as Mean, standard deviation, kurtosis and skewness of integrated profile, the standard deviation is quite high, that is the values are distributed much more in the positive class and are concentrated in the negative class. For the remaining features we see an opposite trend that is the values are distributed much more in the class 0 and are concentrated in the 1 class. We also see that for features like mean, standard deviation of integrated profile and kurtosis, skewness of the DM-SNR curve, the mean is lower in the positive class than the negative class, while it is the opposite for the other features.

Moving on to bivariate analysis, wherein we create pairpolots for each of the columns we see that all the features have a clear boundary of separation and hence we would be able to perform well with our SVM model. We only fee some

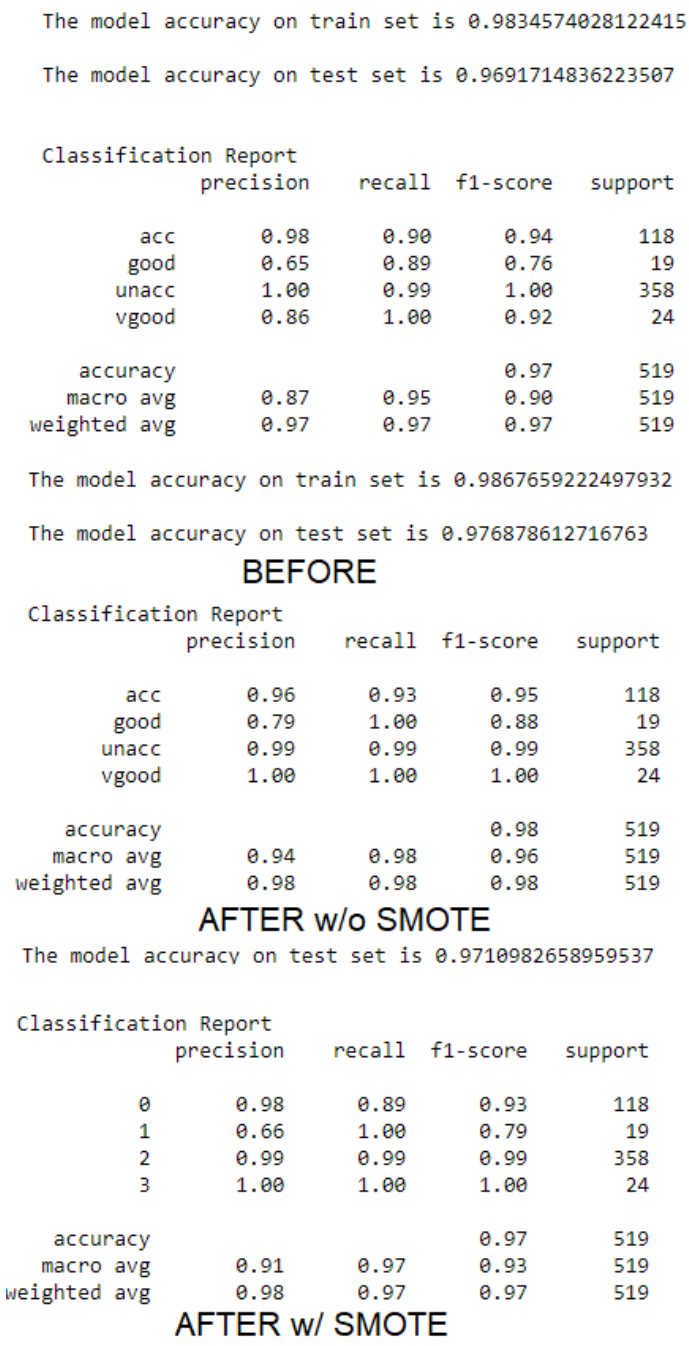


Fig. 62. Ass4: Classification Report Before vs After

overlap near the decision boundary regions which can be tuned using regularization values (C values) to find a good fit.

We then move on to creating a correlation map for the dataset and see high correlation between the values, which can cause trouble and should be handled if our accuracy or f1 values are not good enough. Some of the methods can be removing some features or creating hybrid combinations of the features.

3) Postprocessing: We split the data using a 80/20 split, with the final dataset being 10022 examples in training set and 2506 examples in crossvalidation set. We then use the Standard scaler to scale our train and validation values.

4) Support Vector Machine Modelling: We start with modelling using the default hyperparameters usnig the SVC library of sklearn. Default hyperparameter means C=1.0, kernel=rbf and gamma=auto among other parameters.We observe model accuracy score with default hyperparameters: 0.9741 and model f1 score with default hyperparameters: 0.8426.Based on the above analysis we can conclude that our classification model accuracy is very good. Our model is doing a very good job in terms of predicting the class labels. But, this is not true. Here, we have an imbalanced dataset. The problem is that accuracy is an inadequate measure for quantifying predictive performance in the imbalanced dataset problem. So, we must explore alternative metrices that provide better guidance in selecting models. In particular, we would like to know the underlying distribution of values and the type of errors our classifer is making. Thus we check for the f1 score which is helpful with such imbalanced datasets. We see that our model is performing well on the f1 score with a score of 0.84.

But we are not done yeat. We can tune the hyperparameters to get a better score and a better fit. Thus we use Grid Search to search from our defined space of hyperparameters such as trying the different kernels, with C ranging from 0.01 to 10. We also vary the degree for the linear kernel and set the class weightsa s balanced. We also check for the gamma hyperparameter with values as scale and auto. Finally using a 2 fold cross validation we get the best model with 0.87 F1 score on the train and 0.85 on the validation which is an improvement over the initial score of 0.84. The best hyperparameters that are chosen are rbf kernel with auto gamma and C = 2.51 which is higher than the default of 1. Thus we are reducing the regularization.

IV. Conclusion

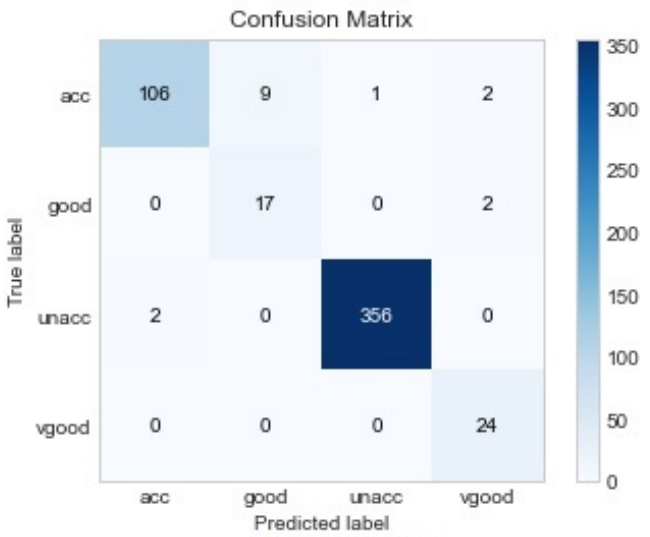
We have divided this paper into two section, one involving a detailed analysis of the stock time series prediction problem and the other part being improving on the already presented problem statements presented during the whole course of this semester. Starting with the stock analysis problem, involving certain stock attributes for entities like Cognizent, HCL, HDFC, Infosys, SBI and USD. We have been successful in answering the questions like the change in stock price over time, daily return of the stock on average, moving average of the various stocks, correlation between the different stocks and the risk by investing in the different stocks and finally a verdict on investing in what stocks will be a good bet. We conclude on the verdict that Cognizant is a bad deal as the associated risk is high with the expected return being low where infosys can be a better bet with similar risk and much higher expected returns. And SBI has the highest expected return with the highest risk associated, while USD has the lowest associated risk and also the lowest returns.

In the second part, we have applied newer techniques like Outlier Analysis and Treatment using the IQR method, Upsampling and Downsampling using Sklearn's Resample,

Feature Selection using Variance Threshold and Polynomial Feature Transformation and have achieved significant gains in terms of both accuracy metrics and robustness. In this part we have tried to use different techniques for each of the assignment so that we could cover more techniques and not repeat the techniques used for other assignments and thus have kept the analysis crisp and to the point. Having applied these newer techniques and getting the desired gains we are confident on having learnt from the course over the semester.

REFERENCES

[1] https://www.kaggle.com/gunesevitan/titanic-advanced-feature-engineering-tutorial
[2] https://www.kaggle.com/seemamishra33/titanic-survival-prediction?scriptVersionId=37753005
[3] https://www.kaggle.com/lonnieqin/stock-market-analysis-prediction-using-lstm
[4] https://towardsdatascience.com/heres-what-i-ve-learnt-about-sklearn-resample-ab735ae1abc4
[5] https://www.analyticsvidhya.com/blog/2020/11/handling-imbalanced-data-machine-learning-computer-vision-and-nlp/
[6] https://www.kaggle.com/plutosenthil/sklearn-notes#5.PolynomialFeatures
[7] https://www.kaggle.com/srinathsrinivasan1/car-evaluation-using-decision-trees-k-fold
[8] https://www.ritchieng.com/machine-learning-evaluate-classification-model/
[9] https://machinelearningmastery.com/multi-class-imbalanced-classification/
[10] https://www.tableau.com/learn/articles/time-series-analysis
[11] https://www.analyticsvidhya.com/blog/2021/10/machine-learning-for-stock-market-prediction-with-step-by-step-implementation/
[12] https://towardsdatascience.com/why-1-5-in-iqr-method-of-outlier-detection-5d07fdc82097
[13] https://towardsdatascience.com/heres-what-i-ve-learnt-about-sklearn-resample-ab735ae1abc4
[14] https://machinelearningmastery.com/polynomial-features-transforms-for-machine-learning/
[15] https://towardsdatascience.com/how-to-use-variance-thresholding-for-robust-feature-selection-a4503f2b5c3f
[16] Linear regression, Data Analytics Laboratory EE4708, July - November 2021
[17] Linear regression by Prof. Manikandan, Pattern Recognition and Machine Learning, July - November 2021
[18] https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e
[19] https://blog.minitab.com/en/adventures-in-statistics-2/how-to-interpret-regression-analysis-results-p-values-and-coefficients
[20] https://en.wikipedia.org/wiki/Student%27s_t-distribution
[21] https://www.geeksforgeeks.org/detecting-multicollinearity-with-vif-python/
[22] https://stats.stackexchange.com/questions/76441/interpretation-from-loess-graph
[23] https://stats.stackexchange.com/questions/481413/how-to-interpret-this-shape-of-qq-plot-of-standardized-residuals
[24] https://towardsdatascience.com/q-q-plots-explained-5aa8495426c0
[25] https://medium.com/evidentebm/linear-regression-using-statsmodels-d0db5fef16bb
[26] https://data.world/nrippner/cancer-linear-regression-model-tutorial/workspace/file?filename=OLS_regression_walkthrough.ipynb
[27] https://statisticsbyjim.com/regression/heteroscedasticity-regression/
[28] Logistic regression, Data Analytics Laboratory EE4708, July - November 2021
[29] Logistic regression by Prof. Manikandan, Pattern Recognition and Machine Learning, July - November 2021
[30] https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc
[31] https://en.wikipedia.org/wiki/Logistic_regression
[32] https://machinelearningmastery.com/confusion-matrix-machine-learning/
[33] Classification, Data Analytics Laboratory EE4708, July - November 2021
[34] Niave Bayes Classifier by Prof. Manikandan, Pattern Recognition and Machine Learning, July - November 2021
[35] https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c
[36] https://en.wikipedia.org/wiki/Naive_Bayes_classifier
[37] http://dataaspirant.com/2017/02/06/naive-bayes-classifier-machine-learning/
[38] https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn
[39] https://stackabuse.com/the-naive-bayes-algorithm-in-python-with-scikit-learn/
[40] Random Forests, Data Analytics Laboratory EE4708, July - November 2021
[41] Random Forests by Prof. Manikandan, Pattern Recognition and Machine Learning, July - November 2021
[42] https://en.wikipedia.org/wiki/Gradient_boosting
[43] https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74
[44] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
[45] https://www.kaggle.com/prashant111/a-guide-on-xgboost-hyperparameters-tuning
[46] https://www.kaggle.com/shubham47/random-forest-classifier-tutorial
[47] https://notebook.community/Aniruddha-Tapas/Applied-Machine-Learning/Classification/Car%20Evaluation%20using%20Decision%20trees%20and%20Random%20Forests
[48] https://www.kaggle.com/mohitcr7/decision-tree-classifier-beginner-level
[49] https://www.kaggle.com/vipulgandhi/a-guide-to-decision-trees-for-beginners
[50] https://en.wikipedia.org/wiki/Decision_tree_pruning
[51] Support Vector Machines, Data Analytics Laboratory EE4708, July - November 2021
[52] Support Vector Machines by Prof. Manikandan, Pattern Recognition and Machine Learning, July - November 2021
[53] https://en.wikipedia.org/wiki/Support-vector_machine
[54] https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python
[55] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
[56] http://dataaspirant.com/2017/01/13/support-vector-machine-algorithm/
[57] https://www.kaggle.com/prashant111/svm-classifier-tutorial
[58] https://www.ritchieng.com/machine-learning-evaluate-classification-model/
[59] https://en.wikipedia.org/wiki/Kernel_method
[60] https://en.wikipedia.org/wiki/Polynomial_kernel
[61] https://en.wikipedia.org/wiki/Radial_basis_function_kernel

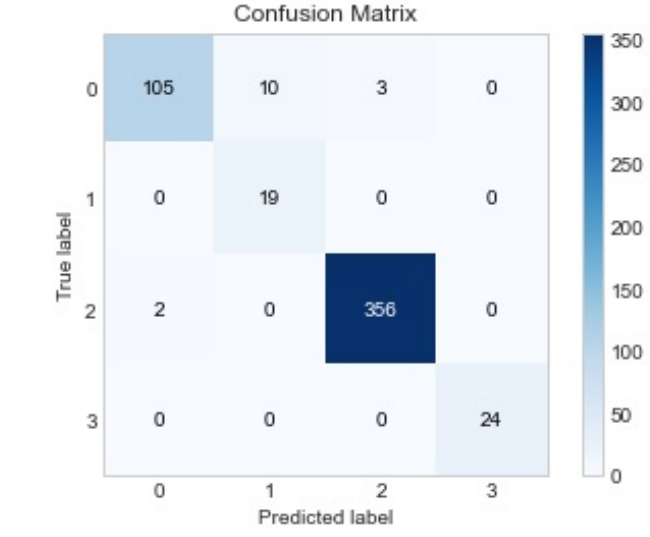


BEFORE

```
The model accuracy on train set is 0.9688850475367329

The model accuracy on test set is 0.9334500875656743

Classification Report
              precision    recall  f1-score   support

         acc       0.89      0.84      0.86       129
        good       0.56      0.90      0.69        20
       unacc       0.98      0.97      0.98       397
       vgood       0.80      0.80      0.80        25

    accuracy                           0.93       571
   macro avg       0.81      0.88      0.83       571
weighted avg       0.94      0.93      0.94       571
```

AFTER w/o SMOTE

```
The model accuracy on train set is 1.0

The model accuracy on test set is 0.9845857418111753

Classification Report
              precision    recall  f1-score   support

         acc       0.97      0.97      0.97       118
        good       0.90      1.00      0.95        19
       unacc       1.00      0.99      1.00       358
       vgood       0.96      0.92      0.94        24

    accuracy                           0.98       519
   macro avg       0.96      0.97      0.96       519
weighted avg       0.98      0.98      0.98       519
```



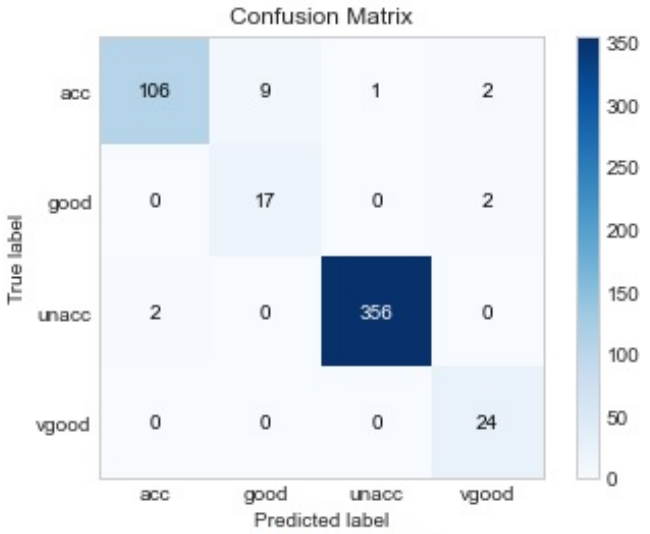
AFTER w/ SMOTE

```
The model accuracy on train set is 1.0

The model accuracy on test set is 0.9865125240847784

Classification Report
              precision    recall  f1-score   support

           0       0.97      0.97      0.97       118
           1       0.86      1.00      0.93        19
           2       1.00      0.99      1.00       358
           3       0.96      1.00      0.98        24

    accuracy                           0.99       519
   macro avg       0.95      0.99      0.97       519
weighted avg       0.99      0.99      0.99       519
```

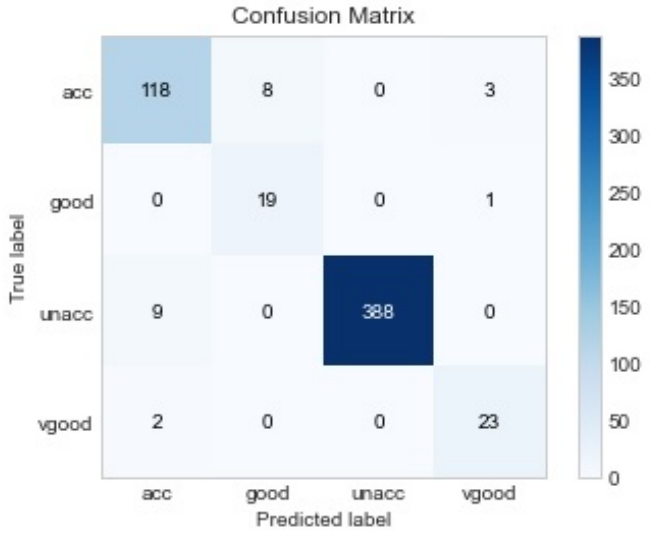
AFTER w/ SMOTE

Fig. 64. Ass5: Classification Report Before vs After

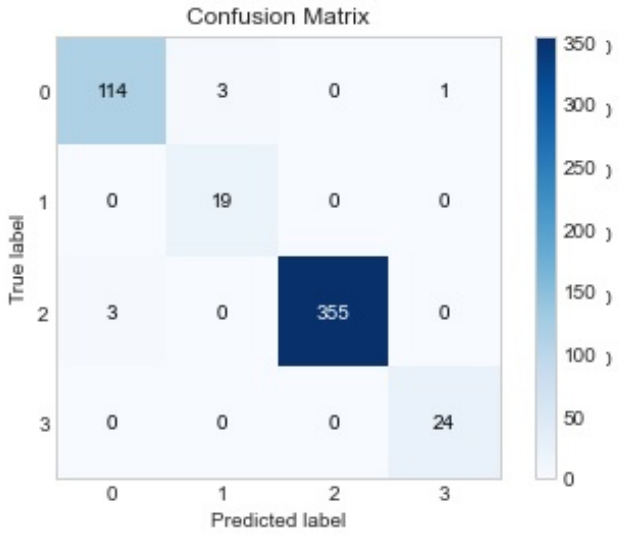
Fig. 63. Ass4: Confusion Matrix Before vs After



BEFORE



AFTER w/o SMOTE



AFTER w/ SMOTE

Fig. 65. Ass5: Confusion Matrix Before vs After

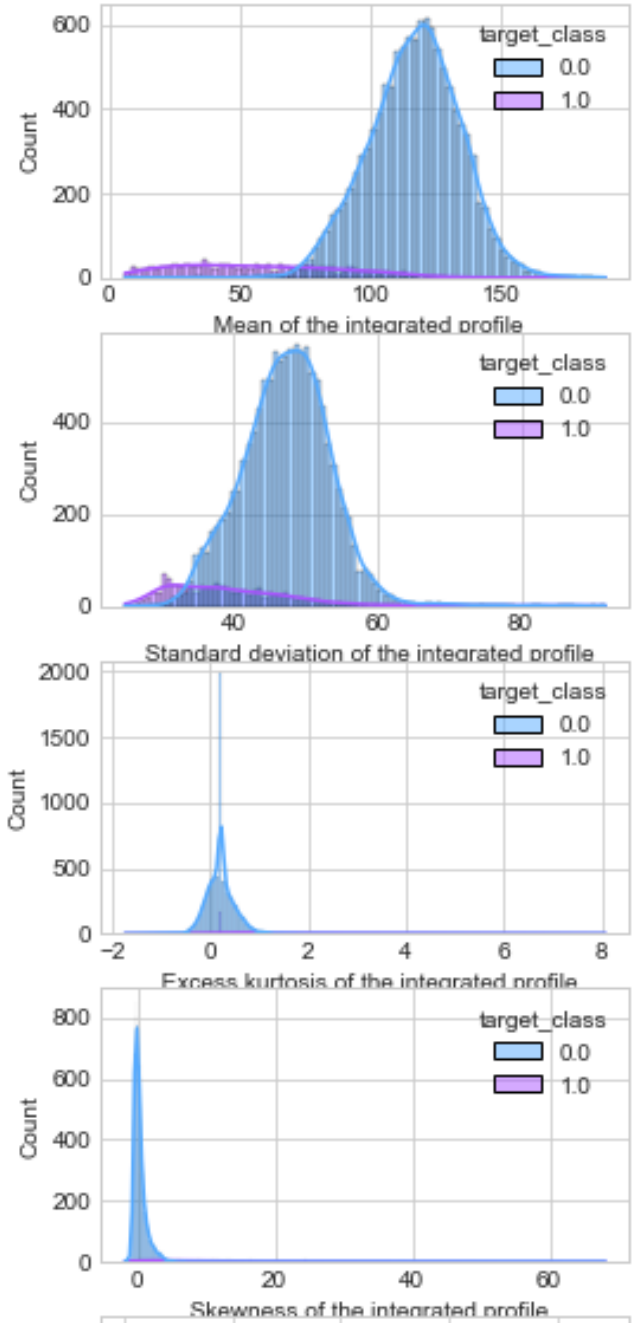


Fig. 66. Ass6: Univariate Analysis

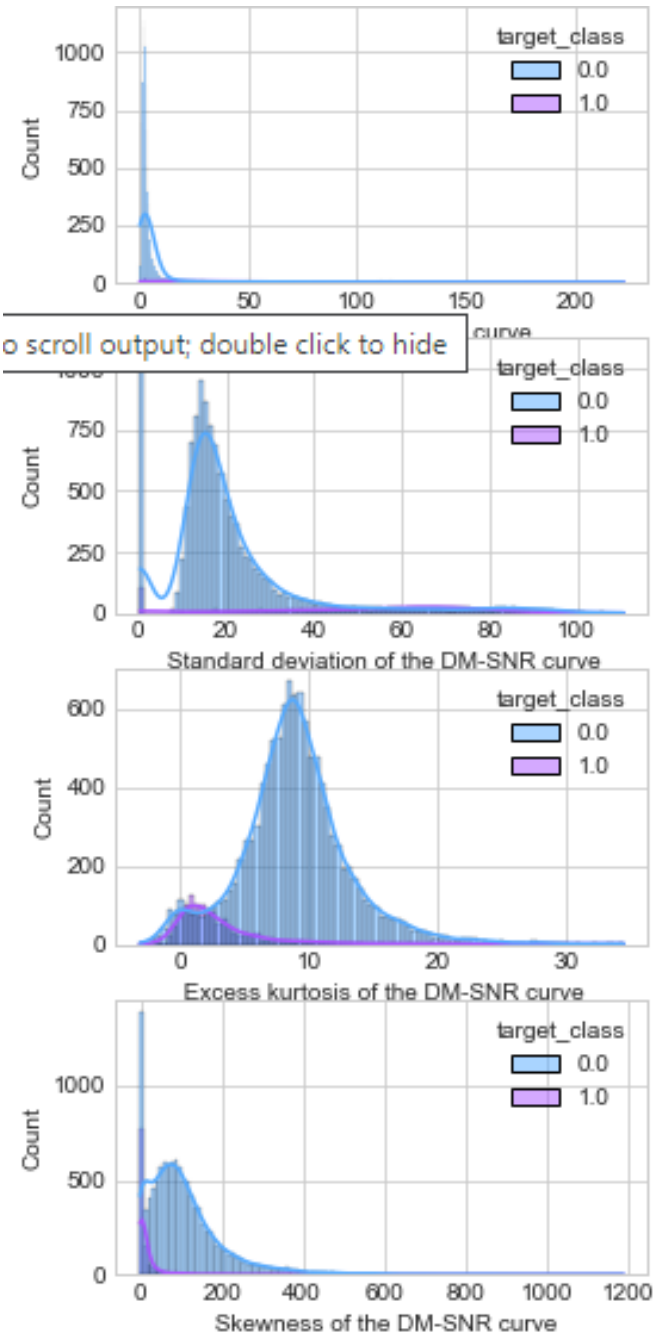


Fig. 67. Ass6: Univariate Analysis

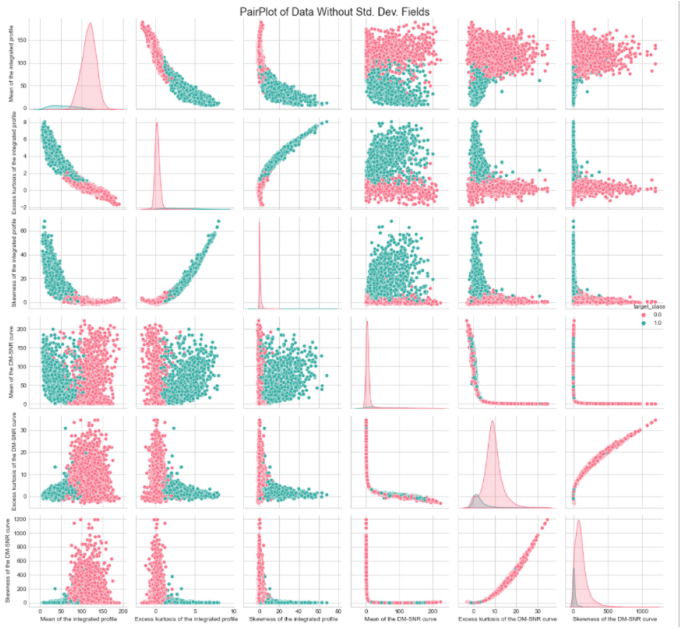


Fig. 68. Ass6: Multivariate Analysis

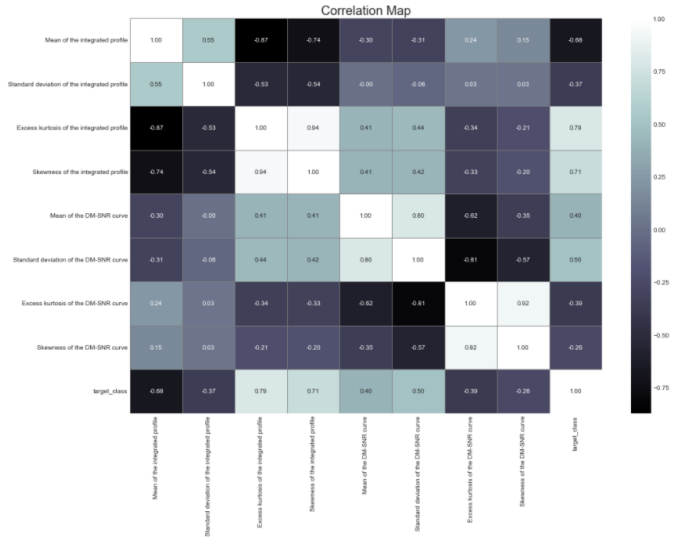


Fig. 69. Ass6: Correlation Map

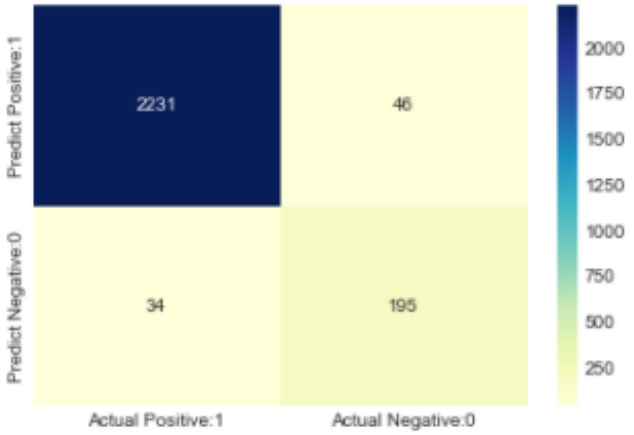


Fig. 70. Ass6: Confusion Matrix

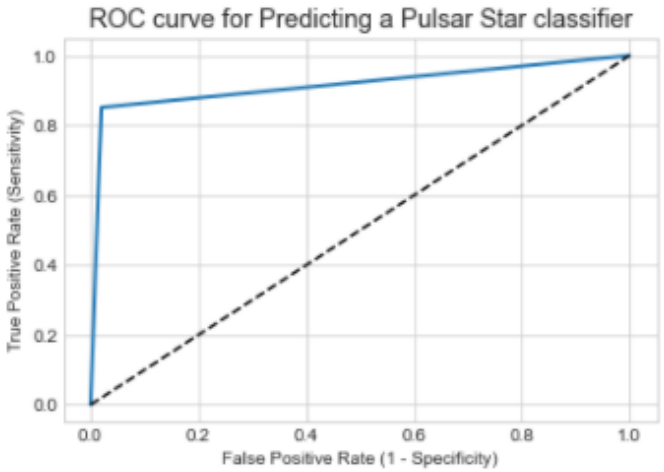


Fig. 71. Ass6: ROC Curve