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 downsampling/undersampling. Upsampling is a procedure where synthetically generated data points (corresponding to minority class) are injected into the dataset. Downsampling is a mechanism 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. 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.

B. 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.

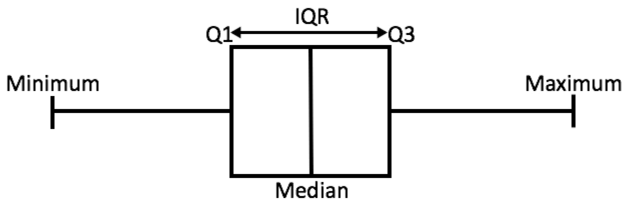


Fig. 1.

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).

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.

$$IQR = Q3 - Q1$$

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.

C. 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.

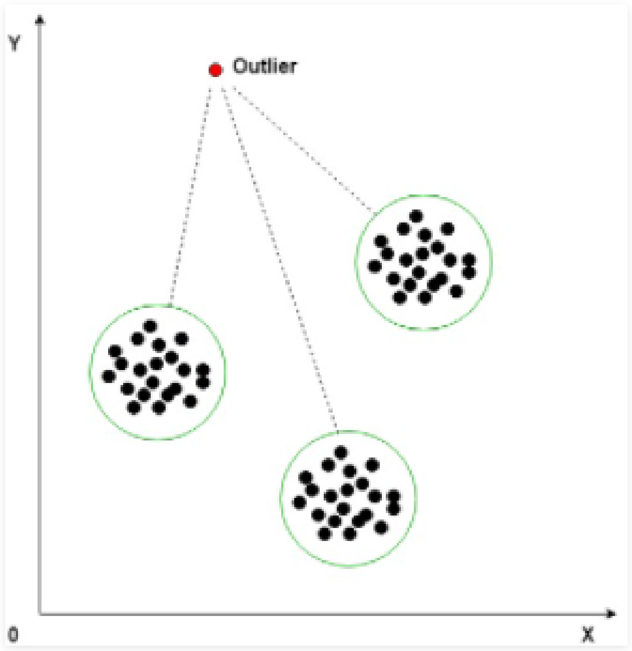


Fig. 2. Outlier Detection

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.

D. 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.

E. 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.

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.

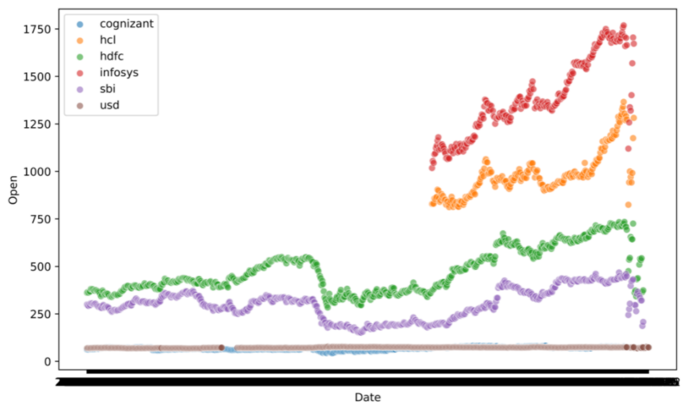


Fig. 3. Time Series: Date vs Other Cols

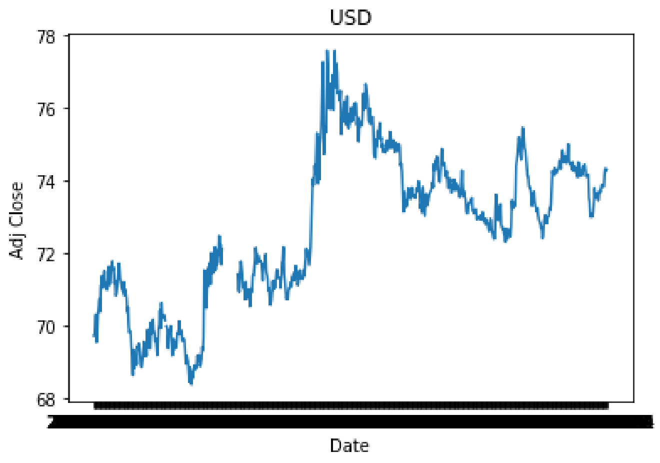


Fig. 4. Time Series: USD — Date vs Adj Close

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

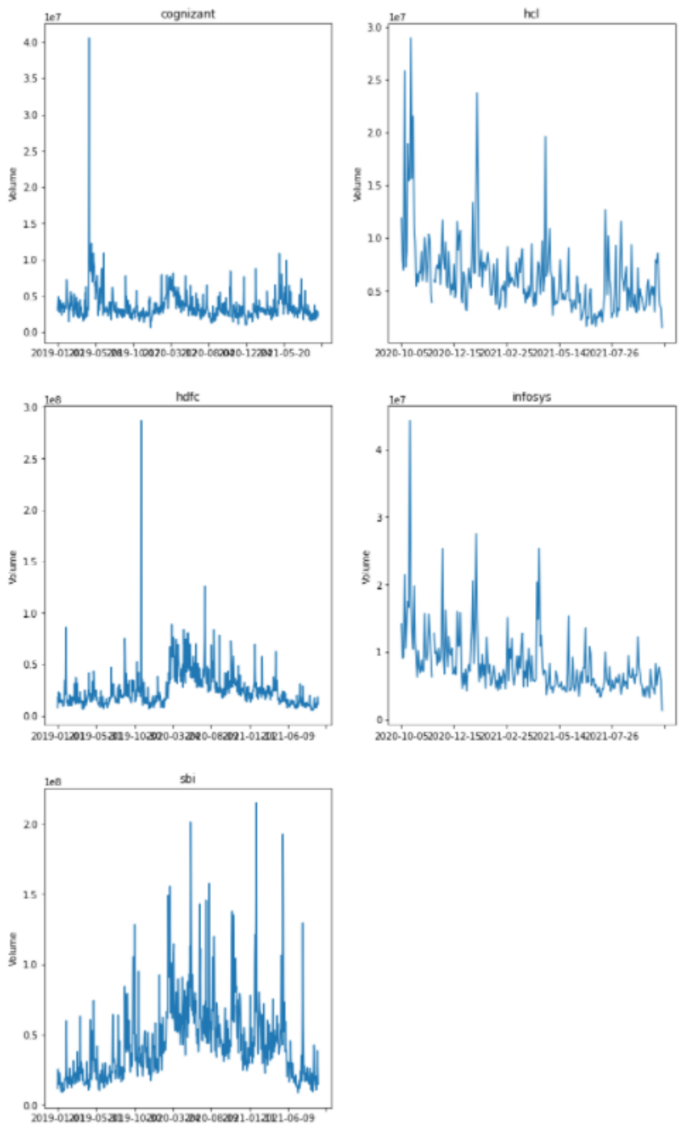


Fig. 5. Time Series: Date vs Vol

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

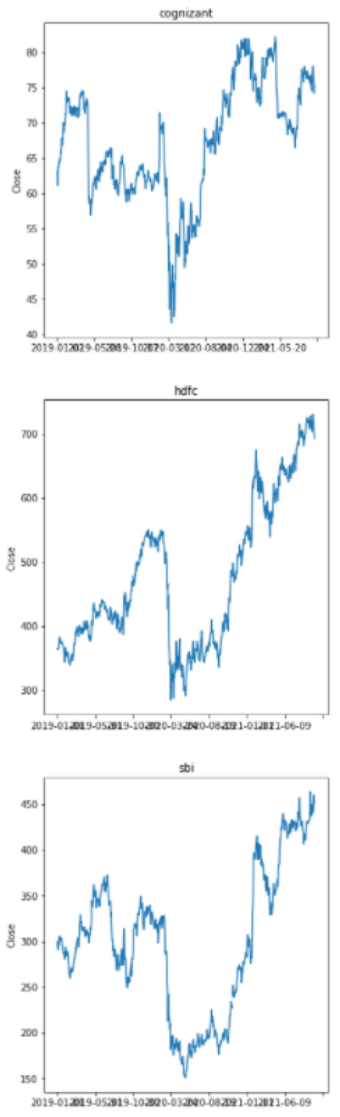


Fig. 6. Time Series: Date vs Close

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.

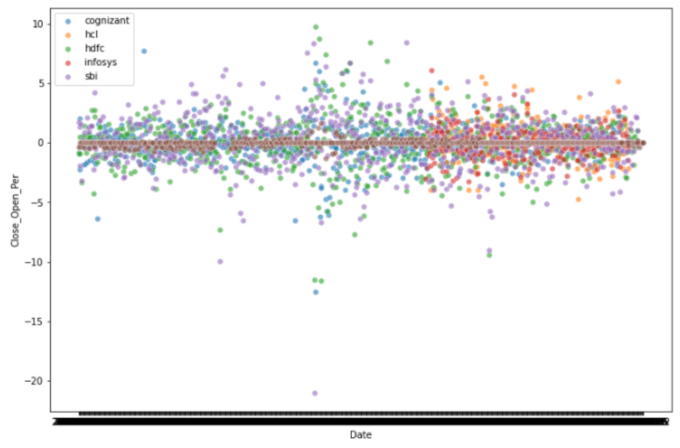


Fig. 7. Time Series: Feature Gen Close minus Open Percentage

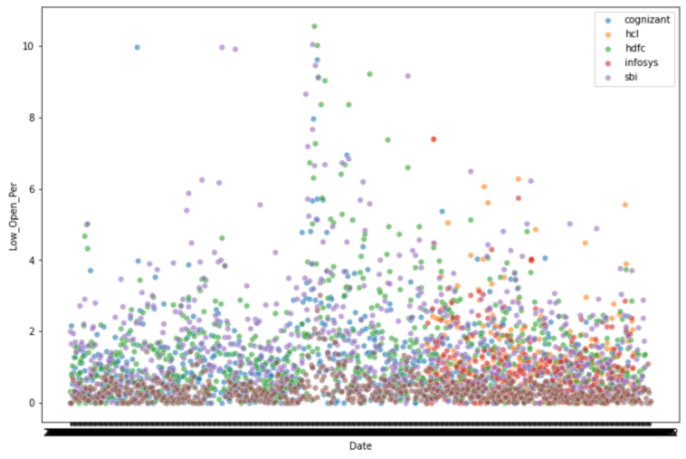


Fig. 8. Time Series: Feature Gen Low minus Open Percentage

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. In the first version we observe some columns with too many missing values like Med Income Black, Native American, Asian and thus drop these columns. All the independent columns are not normalized by population and we also do not have population data, it is better to delete the Mortality rate and Incidence rate columns as these are just the average values normalized by population and hence can be dropped. We also pre-process the data that contains some * marked invalid values due to missing or insignificant data.

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 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

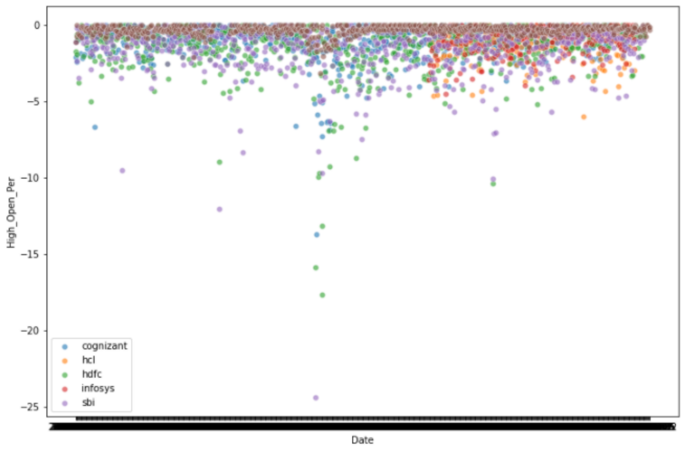


Fig. 9. Time Series: Feature Gen High minus Open Percentage

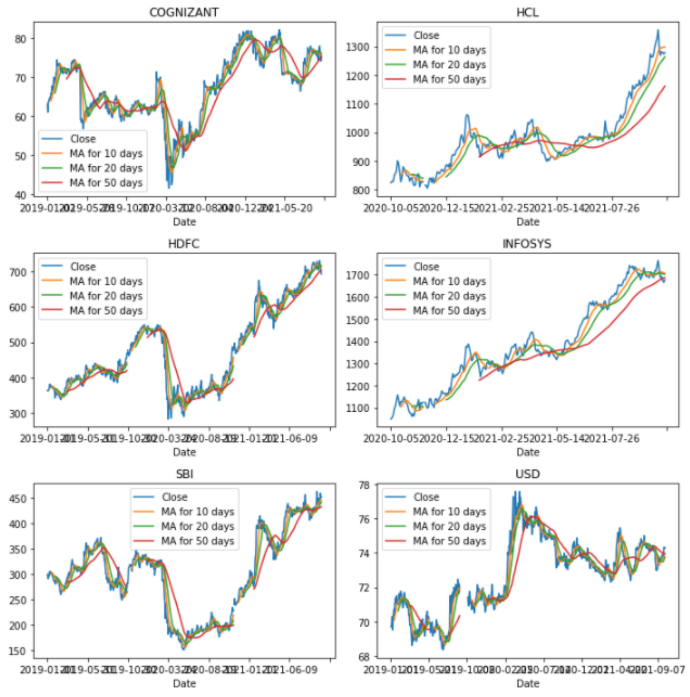


Fig. 10. Time Series: Moving Average

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

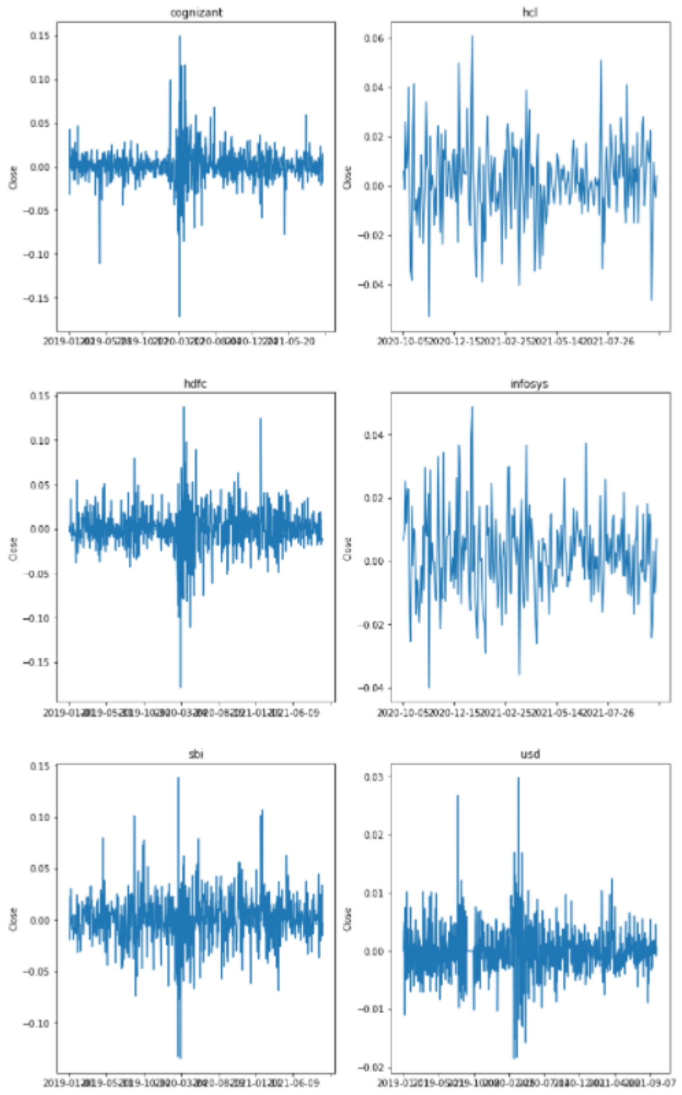


Fig. 11. Time Series: Daily Return Lineplot

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

Assignment 2 is the famous titanic survival prediction problem wherein we are presented with user data and we need to predict whether the person survived or not using classification techniques. We observe that we have only 38 % of the total population survived and hence this is a class imbalance problem. We observe some features with null values and hence are required to treat them. For the cabin feature with the most number of null values we create a new class M which stands for missing.

Following this we also create new features as Ticket frequency (the number of people with the same ticket id), Title (Extracted from name), isMarried, isChild and FamSize (then grouped into bins). Finally all the pre-processing is done

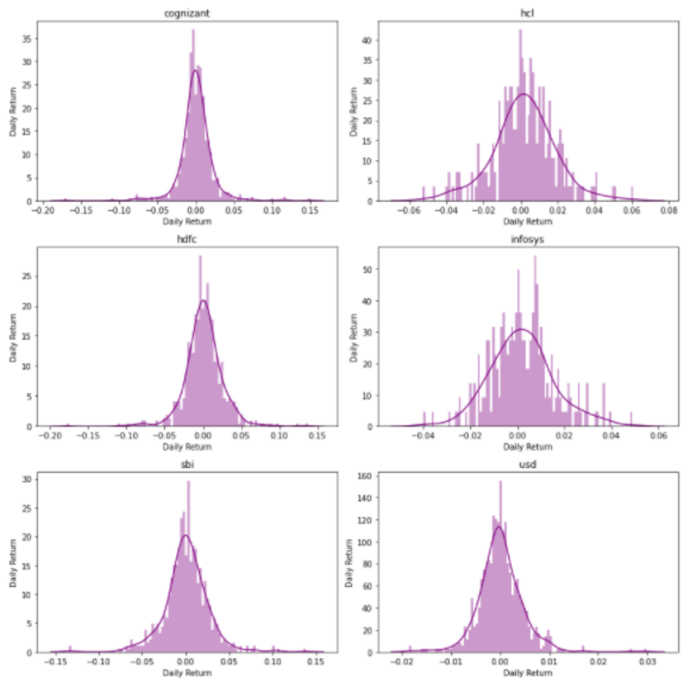


Fig. 12. Time Series: Daily Return distplot

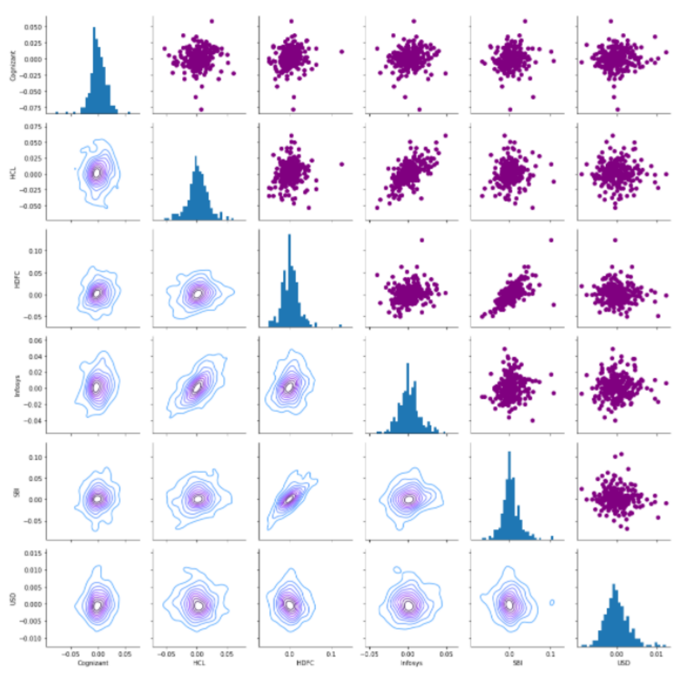


Fig. 14. Time Series: Daily Return Pairgrid

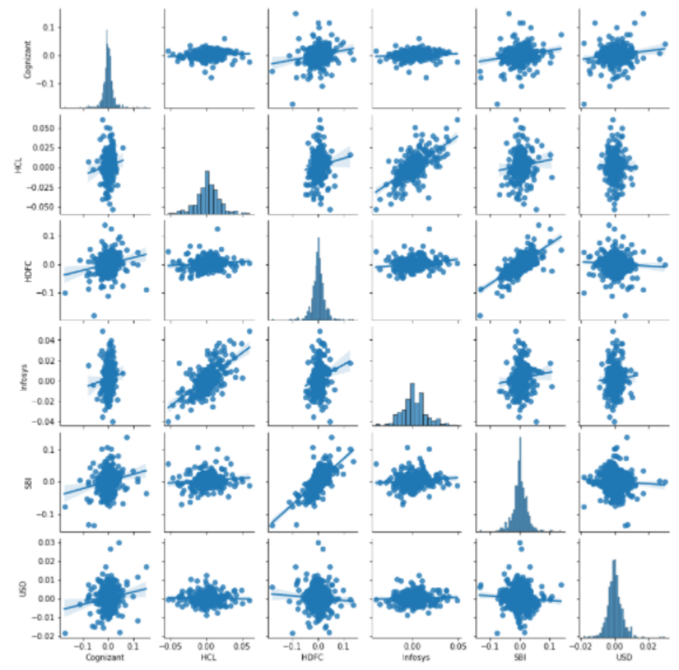


Fig. 13. Time Series: Daily Return Pairplot

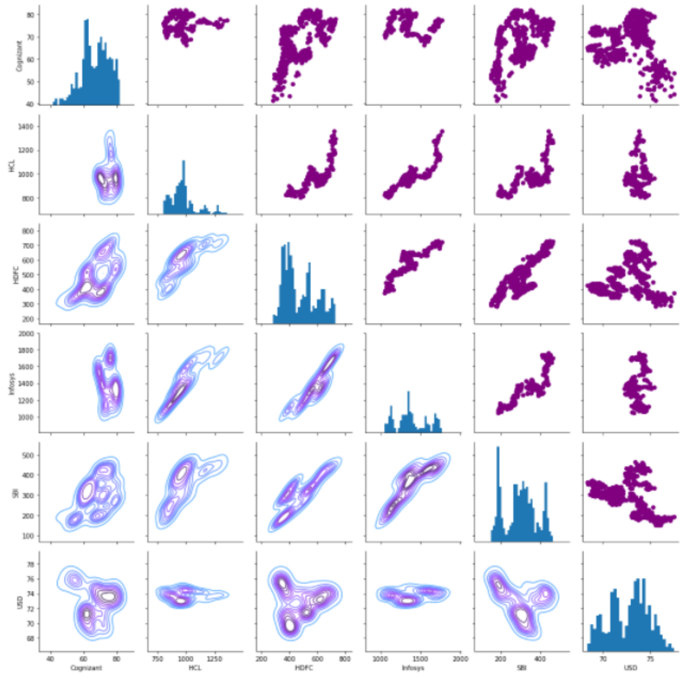


Fig. 15. Time Series: Closing Df Pairplot

(converting categorical variables to dummies) and a statistical linear regression model using the Statsmodel library is fit on the data. 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

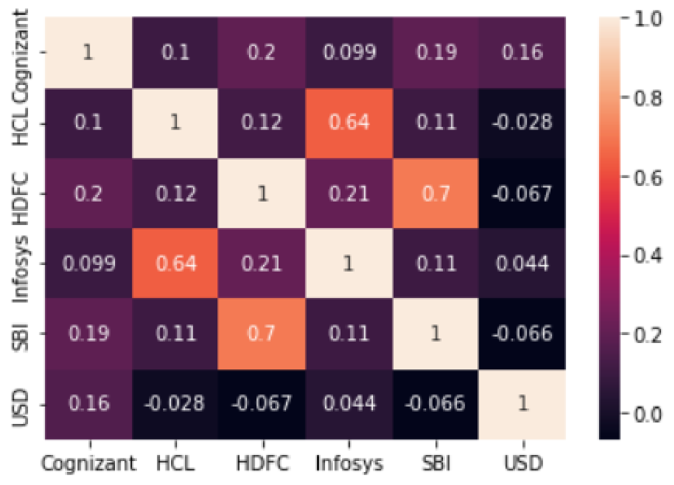


Fig. 16. Time Series: Daily Return Correlation Plot

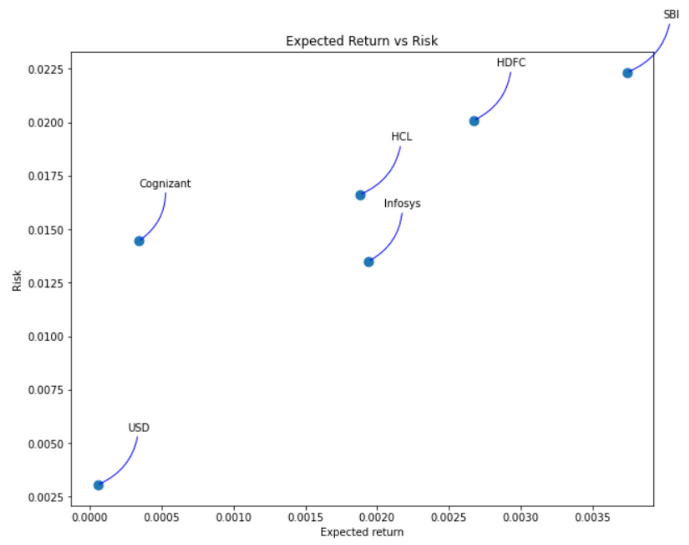


Fig. 18. Time Series: Expected return vs Risk

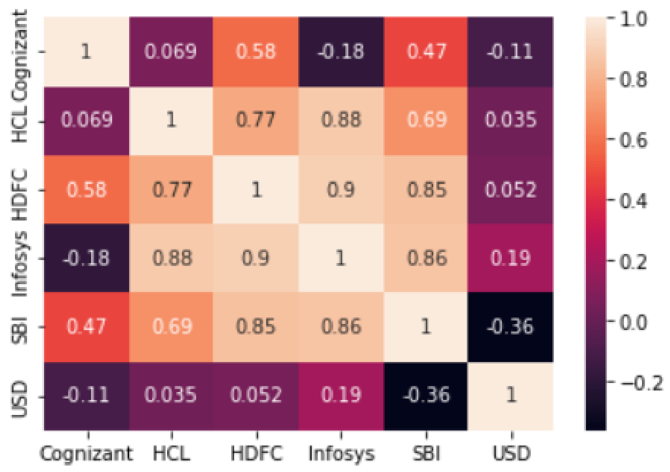


Fig. 17. Time Series: Closing Df Correlation Plot

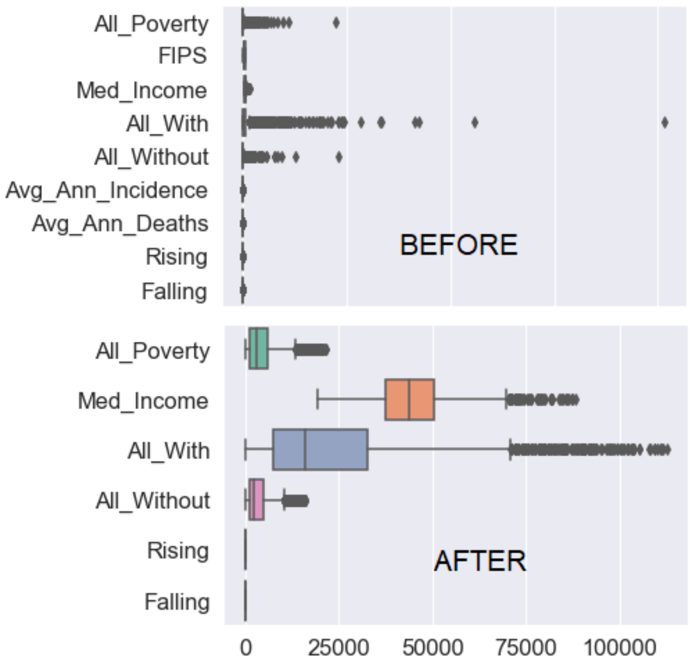


Fig. 19. Ass 1: Boxplot before vs after

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.

D. Assignment 3: Naive Bayes Classifier

Assignment 3 is also a classification problem wherein we are presented with different attributes of a man and are required to classify his/her salary as being more than 50K or not. 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.

Jumping straight to the problem set, we see that we have a mix of both categorical and numerical variables. We also see that some of the variables have invalid samples with ? marks and need to be treated. Thus we replace these invalid points with their mode values. We then create new features as IsChild, SeniorCitizen, IsHighlyEducated and worksmore which are self explanatory. 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

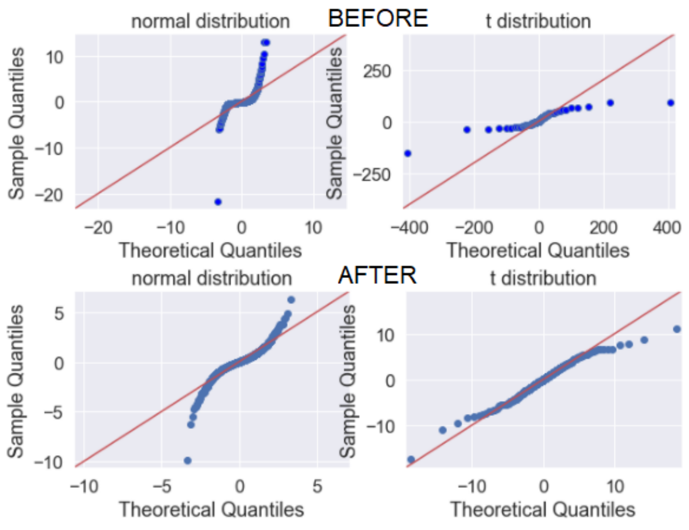


Fig. 20. Ass 1: QQ plot before vs after

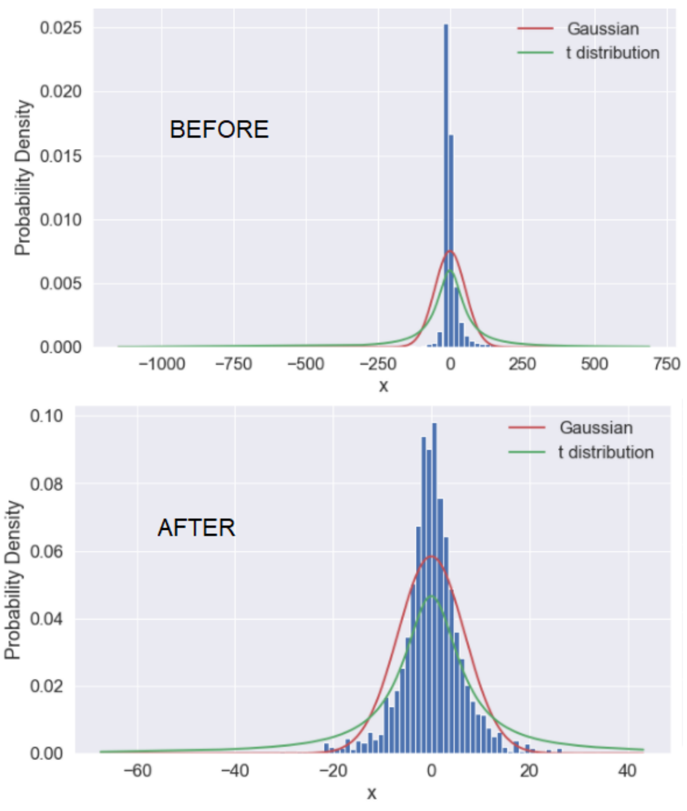


Fig. 21. Ass 1: Residual Plot before vs after

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

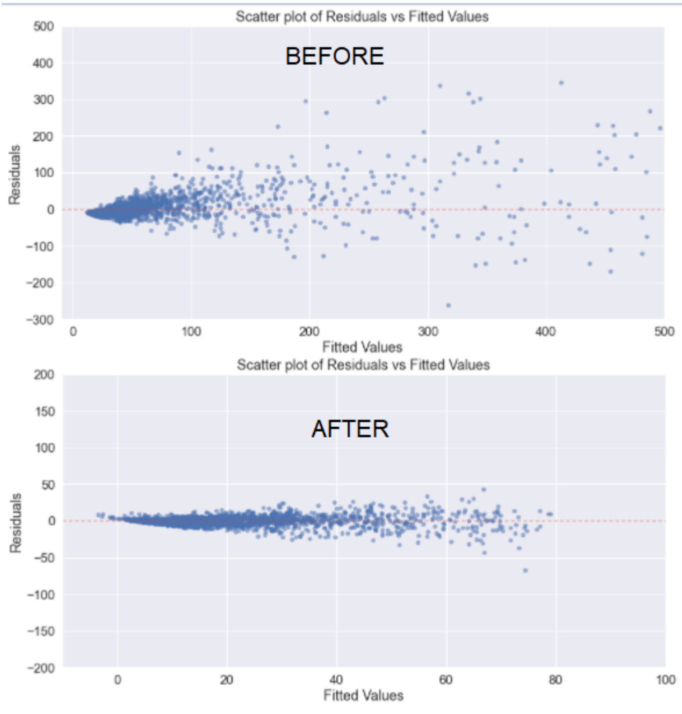


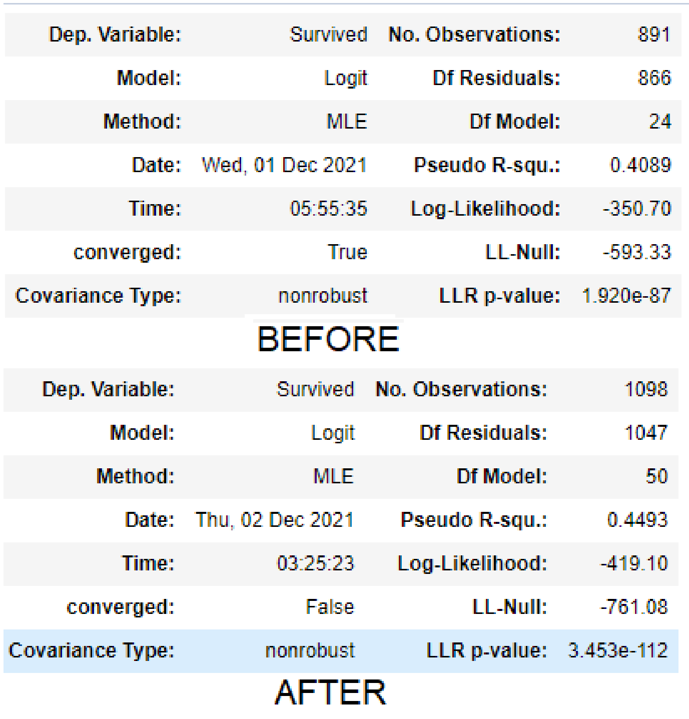
Fig. 22. Ass 1: Residual Scatter before vs after

better f1 score of 0.64 as compared to 0.61 initially.

E. 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 an imbalanced dataset with target percentages as 3.8, 4, 22.2 and 70 percent. As a pre-processing step we encode the categorical variables using the ordinal encoder which does not one-hot encode the variables but assigns different numerical tags for the different categories.

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 fo 0.971 which is not an improvement over the previous score but an improvement over the initial vanilla model.



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. 23. Ass 2: Model stats before vs after

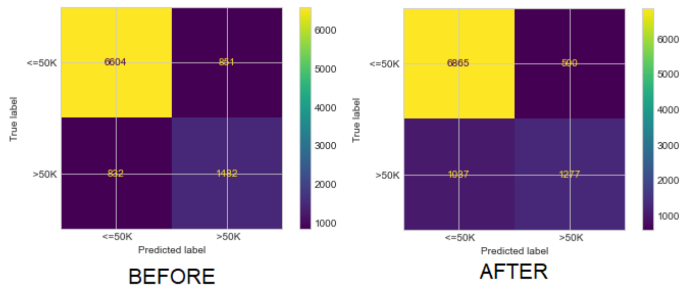
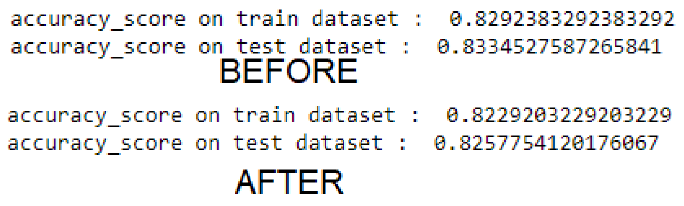


Fig. 24. Ass3: Confusion Matrix Before vs After

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 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.

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



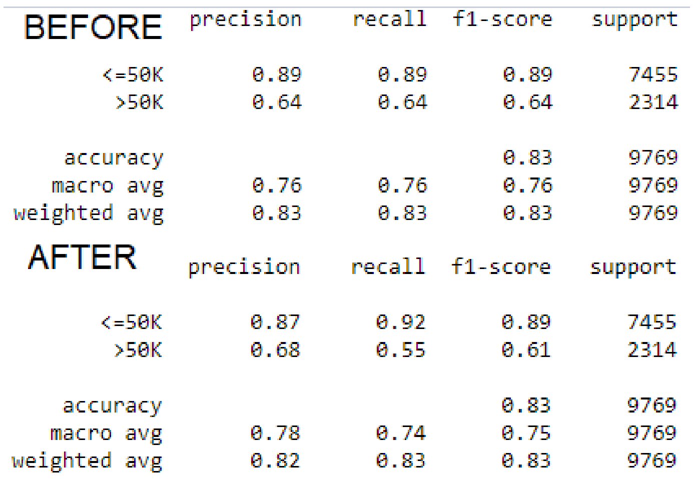
```
accuracy_score on train dataset :  0.8292383292383292
accuracy_score on test dataset :  0.8334527587265841
```

BEFORE

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

AFTER

Fig. 25. Ass3: Accuracy Before vs After

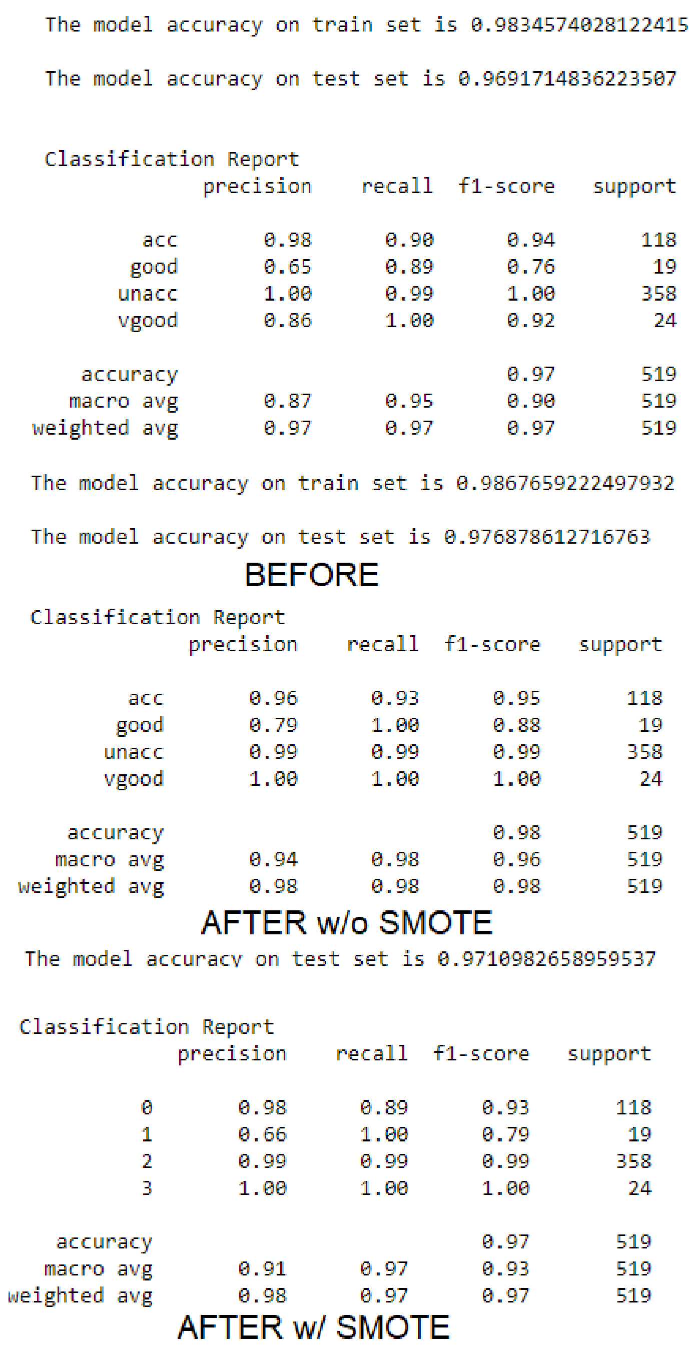


```
BEFORE  precision    recall  f1-score   support

       <=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. 26. Ass3: Classification Report Before vs After

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.



```
 The model accuracy on train set is 0.9834574028122415

 The model accuracy on test set is 0.9691714836223507


 Classification Report
               precision    recall  f1-score   support

         acc       0.98      0.90      0.94       118
        good       0.65      0.89      0.76        19
       unacc       1.00      0.99      1.00       358
       vgood       0.86      1.00      0.92        24

    accuracy                           0.97       519
   macro avg       0.87      0.95      0.90       519
weighted avg       0.97      0.97      0.97       519
```

```
The model accuracy on train set is 0.9867659222497932

The model accuracy on test set is 0.976878612716763

Classification Report
               precision    recall  f1-score   support

         acc       0.96      0.93      0.95       118
        good       0.79      1.00      0.88        19
       unacc       0.99      0.99      0.99       358
       vgood       1.00      1.00      1.00        24

    accuracy                           0.98       519
   macro avg       0.94      0.98      0.96       519
weighted avg       0.98      0.98      0.98       519
```

```
The model accuracy on test set is 0.9710982658959537


Classification Report
               precision    recall  f1-score   support

           0       0.98      0.89      0.93       118
           1       0.66      1.00      0.79        19
           2       0.99      0.99      0.99       358
           3       1.00      1.00      1.00        24

    accuracy                           0.97       519
   macro avg       0.91      0.97      0.93       519
weighted avg       0.98      0.97      0.97       519
```

Fig. 27. Ass4: Classification Report Before vs After

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

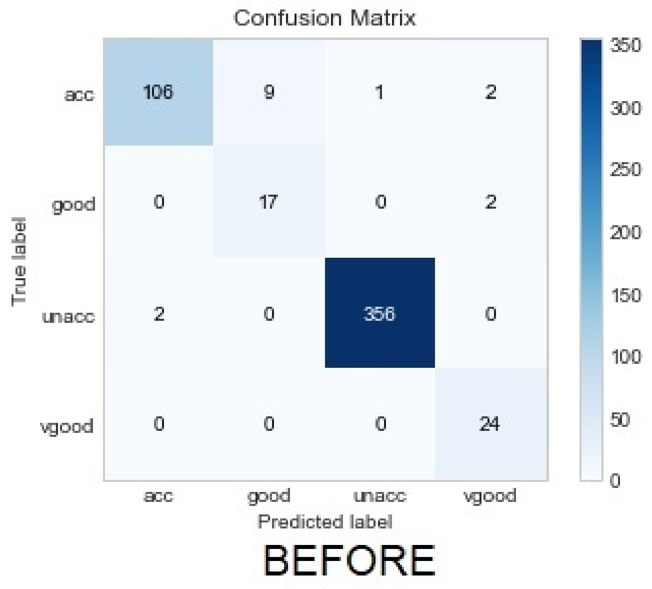
[11] https://www.analyticsvidhya.com/blog/2021/10/machine-learning-for-stock-market-prediction-with-step-by-step-implementation/

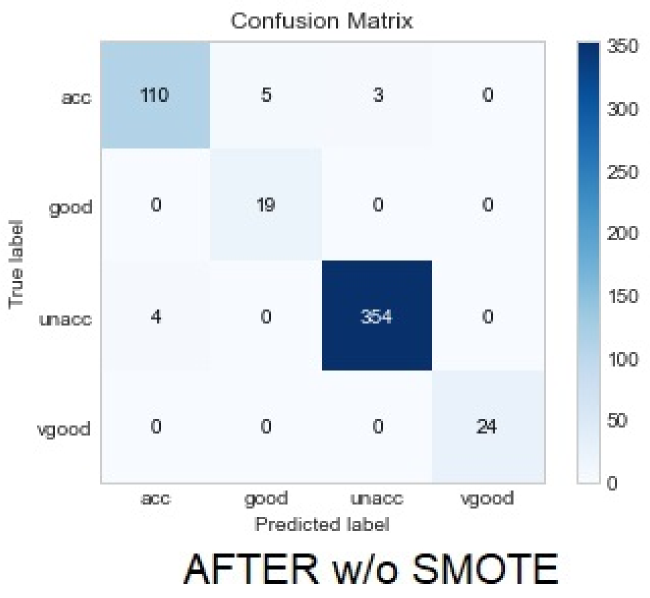
[12] https://towardsdatascience.com/why-1-5-in-iqr-method-of-outlier-detection-5d07fdc82097

[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





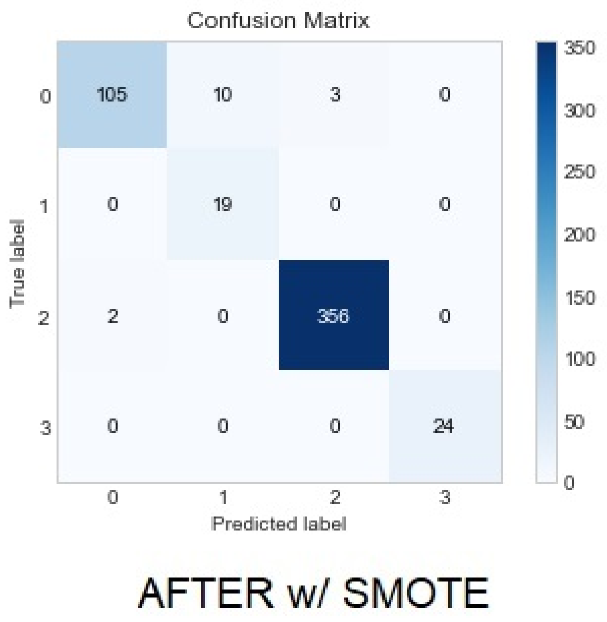
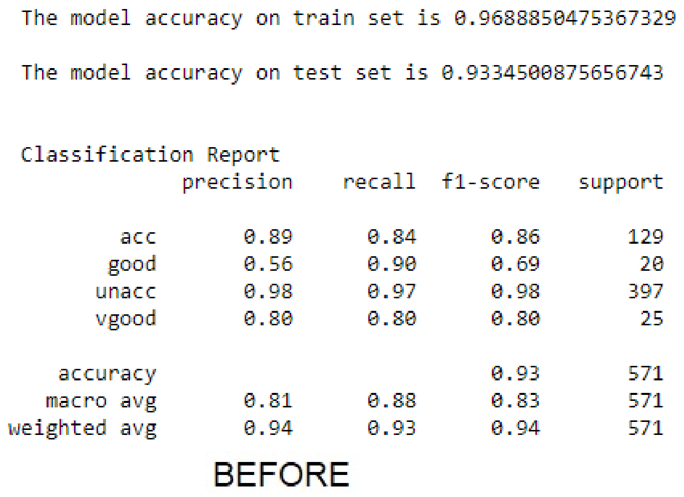
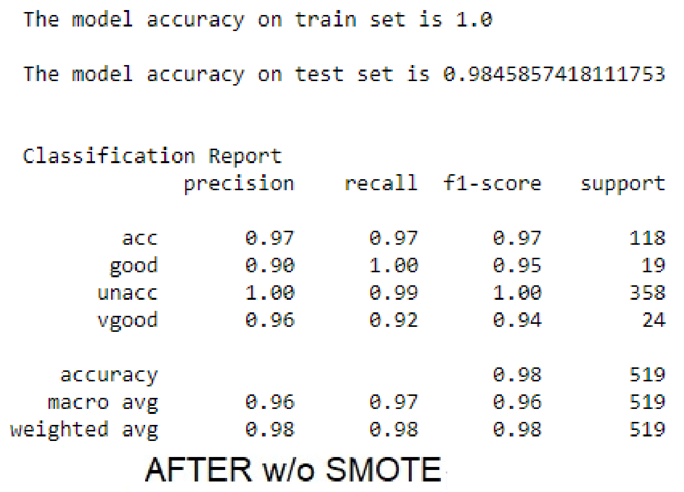


Fig. 28. Ass4: Confusion Matrix Before vs After


```
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
```
BEFORE


```
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/o SMOTE

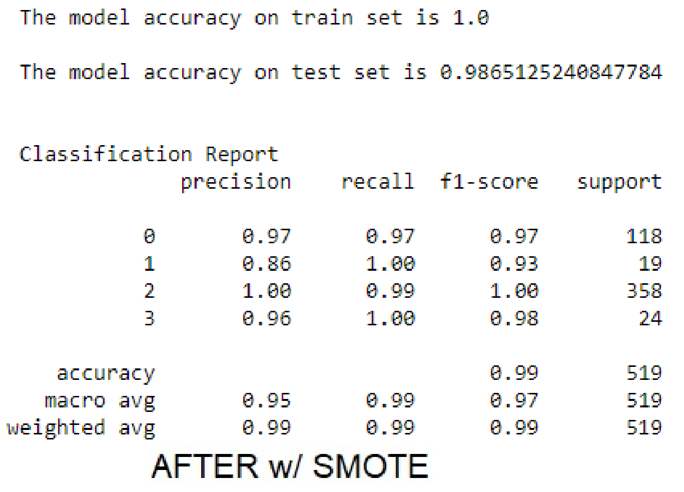
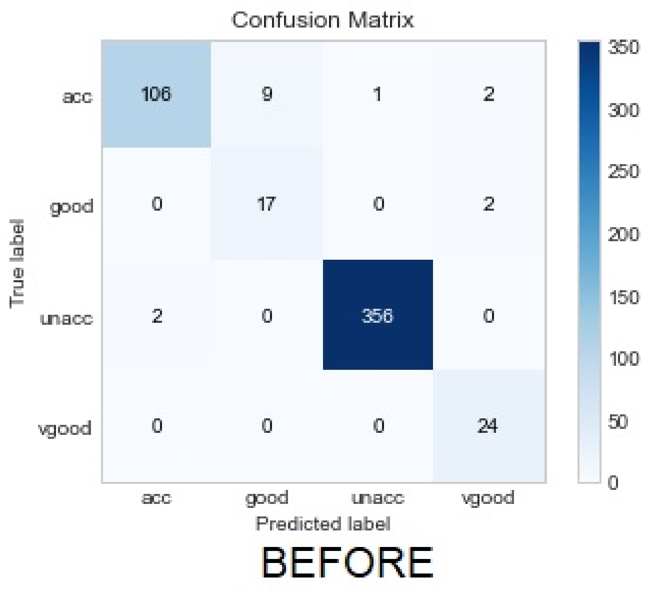
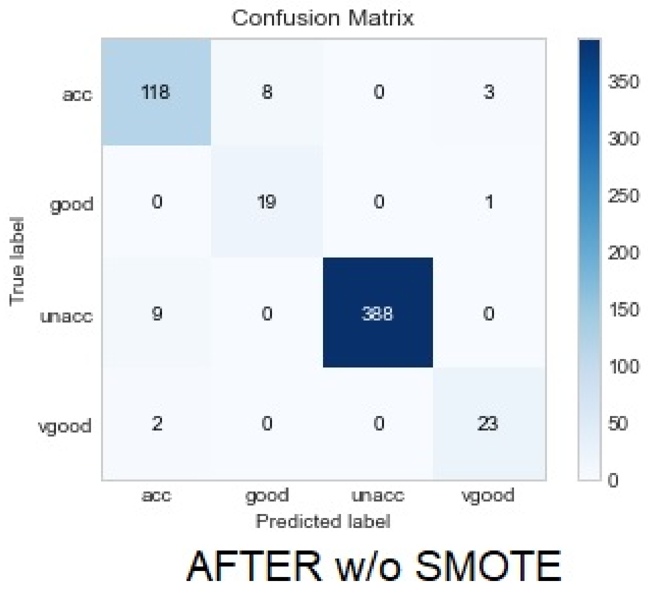

```
The model accuracy on train set is 1.0

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. 29. Ass5: Classification Report Before vs After





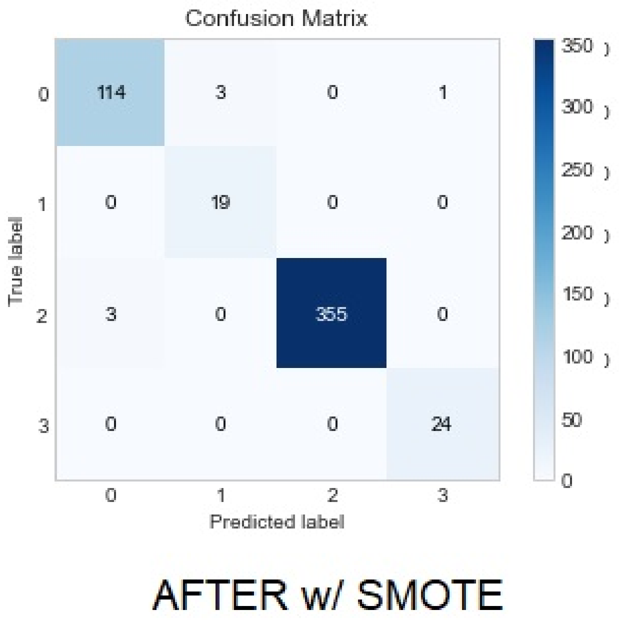


Fig. 30. Ass5: Confusion Matrix Before vs After