Data Analytics Lab: Final Exam Improvements in the Assignments

Arjav Singh
Metallurgical and Materials Engineering
Indian Institute of Technology Madras
Chennai, India
mm20b007@smail.iitm.ac.in

***Abstract*—This document presents a second attempt on the previous assignments, focusing on improvising them using better techniques and methods.**

I. Introduction

A. Linear Regression

Linear regression is a method employed to uncover the linear connection between a target variable and one or more predictors. It assumes that the model being learned follows a linear pattern based on certain parameters. Three primary types of linear regression exist: Simple and Multiple, where one involves a single independent variable and the other involves multiple independent variables. The third type, Polynomial regression, extends from multiple linear regression by incorporating higher powers of input features. In this context, one variable is the predictor, while the other acts as the response.

Linear regression seeks a statistical relationship rather than a deterministic one between variables. A deterministic relationship implies that one variable can precisely predict or express the other. For instance, using degrees in temperature allows an accurate prediction of Fahrenheit. On the other hand, a statistical relationship, such as that between height and weight, does not precisely determine the connection between two variables.

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 (na¨ıve) 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. Na¨ıve 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. Na¨ıve 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. Na¨ıve 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. Metrics for model evaluation

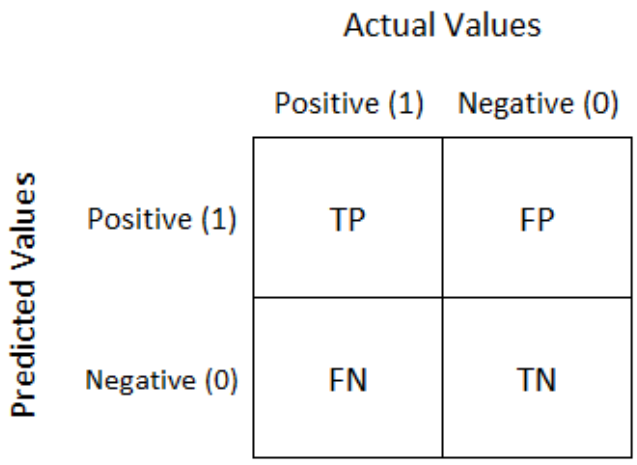


Fig. 1. Confusion Matrix.

1) **Confusion Matrix**: It is used to summarize the performance of a classification algorithm on a set of test data for which the true values are previously known. Sometimes it is also called an error matrix. Terminologies of the Confusion matrix (Figure 1) are:
   - **True Positive**: TP means the model predicted yes, and the actual answer is also yes.
   - **True negative**: TN means the model predicted no, and the actual answer is also no.
   - **False positive**: FP means the model predicted yes, but the actual answer is no.
   - **False negative**: FN means the model predicted no, but the actual answer is yes.

   The rates calculated using the Confusion Matrix are:
   a) **Accuracy**: (TP+TN/Total) tells about overall how classifier Is correct.
   b) **True positive rate**: TP/(actual yes) it says about how much time yes is predicted correctly. It is also called "sensitivity" or "recall."
   c) **False positive rate**: FP/(actual number) says how much time yes is predicted when the actual answer is no.
   d) **True negative rate**: TN/(actual number) says how much time no is predicted correctly, and the actual answer is also no. It is also known as "specificity."
   e) **Misclassification rate**: (FP+FN)/(Total) It is also known as the error rate and tells about how often our model is wrong.
   f) **Precision**: (TP/ (predicted yes)) If it predicts yes, then how often is it correct.
   g) **Prevalence**: (actual yes /total) how often yes condition actually occurs.
   h) **F1-score**: f1 score is defined as the weighted harmonic mean of precision and recall. The best achievable F1 score is 1.0, while the worst is 0.0. The F1 score serves as the harmonic mean of precision and recall. Consequently, the F1-score consistently yields lower values than accuracy measures since it incorporates precision and recall in its computation. When evaluating classifier models, it is advisable to employ the weighted average of the F1 score instead of relying solely on global accuracy.

2) **ROC curve (Receiver Operating Characteristic)**: The Receiver Operating Characteristic (ROC) curve is a useful tool for assessing a model's performance by examining the trade-offs between its True Positive (TP) rate, also known as sensitivity, and its False Negative (FN) rate, which is the complement of specificity. This curve visually represents these two parameters.
   The Area Under the Curve (AUC) metric to summarize the ROC curve concisely. The AUC quantifies the area under the ROC curve. In simpler terms, it measures how well the model can distinguish between positive and negative cases. A higher AUC indicates better classifier performance.
   In essence, AUC categorizes model performance as follows:
   - If AUC = 1, the classifier correctly distinguishes between all the Positive and Negative class points.
   - If 0.5¡ AUC ¡ 1, the classifier will distinguish the positive class value from the negative one because it finds more TP and TN than FP and FN.
   - If AUC = 0.5, the classifier cannot distinguish between positive and negative values.
   - If AUC =0, the classifier predicts all positive as negative and negative as positive.

II. PROBLEM

A. Task in Hand

We have to revisit the assignments and propose better solutions for them with the help of new methods and techniques.

B. Assignment 1: Linear Regression Updates

The assignment revolves around analyzing cancer data aiming to model two continuous variables: average incidence and deaths. Metrics such as Poverty, Income, and Insurance are used as indicators of socioeconomic status. Dependent variables for analysis include Incidence Rate, Average Incidence, Mortality Rate, and Average Deaths across different areas in the United States identified by the FIPS number.

1) Pre-Processing:

We analyze the impact of social status on median income, the sole available social data. A line plot shows varying incomes across communities in different states, indicating higher incomes among Asians and lower incomes among Black communities. If we can demonstrate a substantial influence of median income on Avg Ann Incidence or Deaths, we can consider social status as a valid factor.

Checking for null values, columns related to median income for social groups like Native American, Asian, Black, and Hispanic exhibit substantial missing data (ranging from 20% to 55%), prompting their deletion. Additionally, columns such as Incidence Rate, Avg Ann Incidence, recent trend, mortality rate, and Avg Ann deaths contain non-numeric values. As independent columns lack normalization by population and population data is unavailable, it's preferable to train our model using Avg Ann Incidence and Avg Ann deaths instead of normalized data.

Regarding missing data in Avg Ann Deaths, 325 datapoints contain an asterisk, representing suppressed data due to confidentiality (fewer than 16 cases reported). We plan to address this issue after evaluating other columns. For Avg Ann Incidence, non-numeric data types include '3 or fewer', ' ', which is relatively insignificant compared to the total data.

Feature extraction generates new columns indicating rising and falling recent trends, replacing the recent trend column. Remaining null data requires treatment before model training. Handling methods like removing rows with null values, imputing using state-wise median, or employing a model capable of handling missing data are considered. The latter method is chosen.

2) Visualization:

Scatter plots for Avg Ann Incidence vs. Avg Ann Deaths (Figure 2) show high correlation, suggesting testing on one would yield similar results for the other. Pair plots reveal high correlation among poverty columns and median income. To address multicollinearity, M poverty, F poverty, M with, M without, F with, F without columns are dropped.

3) Statistical Linear Regression Modeling & Updates:

A statistical linear regression model is built with treated parameters as random variables, yielding an adjusted R-squared score of 0.862. Multicollinearity is checked using VIF, identifying a column with a VIF of around 15, necessitating its removal. After dropping this feature, the model achieves greater stability with an adjusted R-squared value of 0.86.

QQ plots comparing the old and new (outlier-free) data reveal a better fit for the t-distribution in the new data, especially at the extremes. The resulting data is outlier-free, making the model more robust. Residual scatter plots show better alignment with the zero line, indicating a better fit of the regression line to the processed data, resulting in lower errors (residuals).

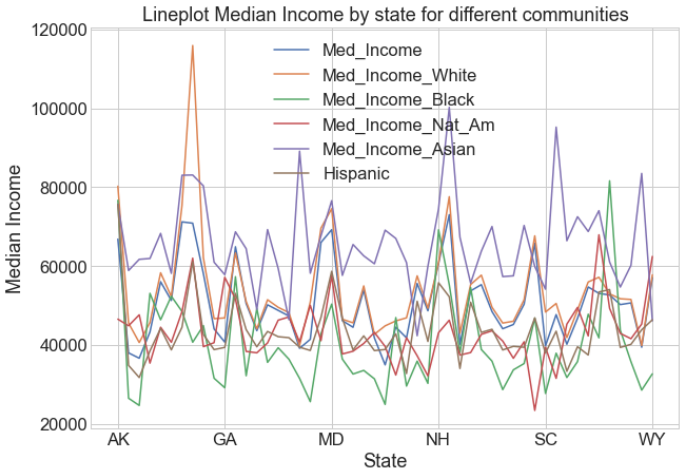


Fig. 2. Assignment 1: Line-plot median income by state

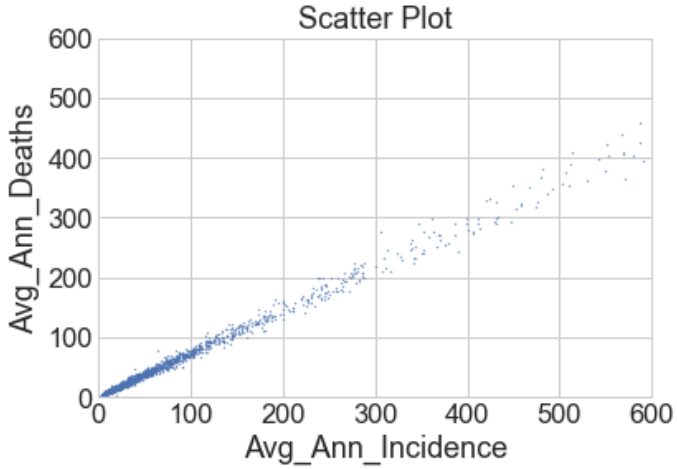


Fig. 3. Assignment 1: Scatter-Plot

C. Assignment 2: Logistic Regression

We are presented with a problem involving the biggest shipwreck of the history, the shipwreck of 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,

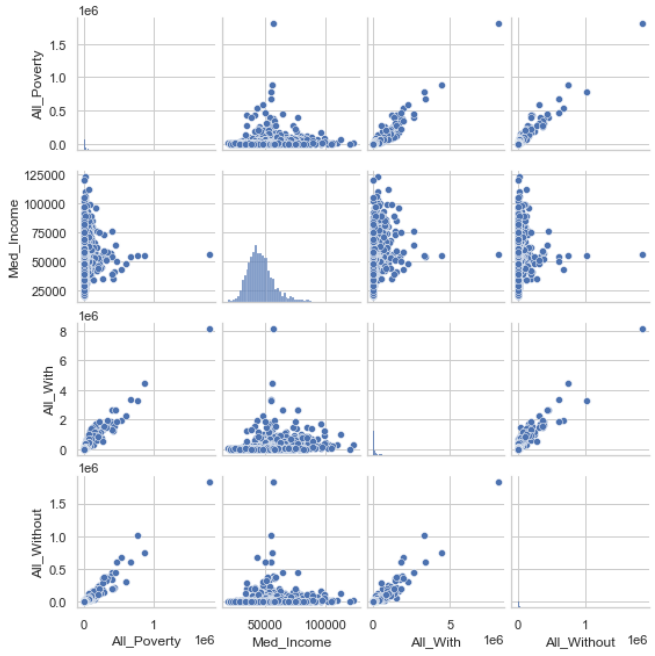


Fig. 22. Assignment 1: Pair-plot

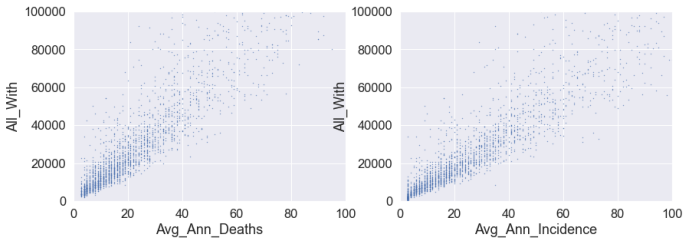


Fig. 23. Assignment 1: Scatter-plot

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.

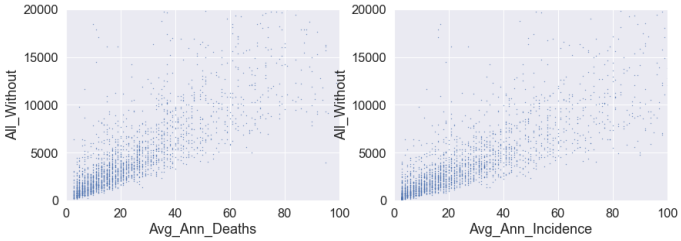


Fig. 24. Assignment 1: Scatter-plot

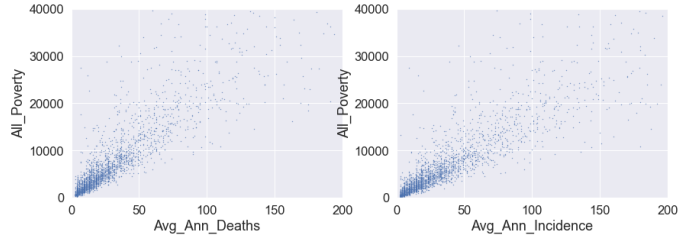


Fig. 25. Assignment 1: Scatter-plot

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.

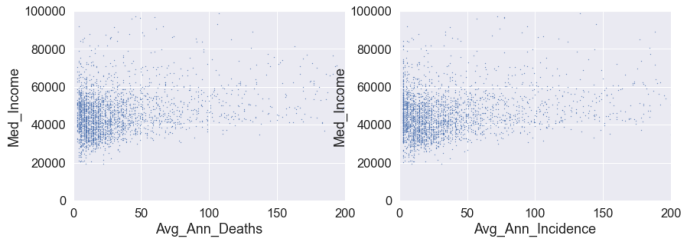


Fig. 26. Assignment 1: Scatter-plot

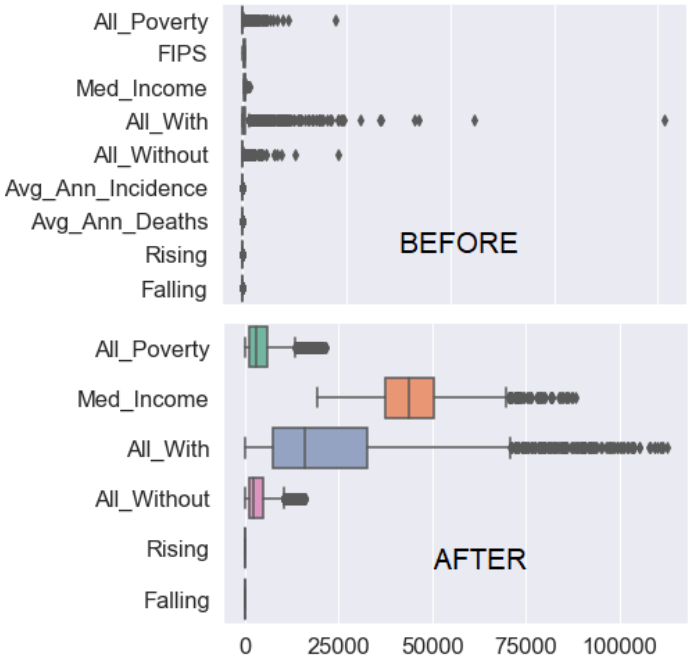


Fig. 27. Ass 1: Boxplot before vs after

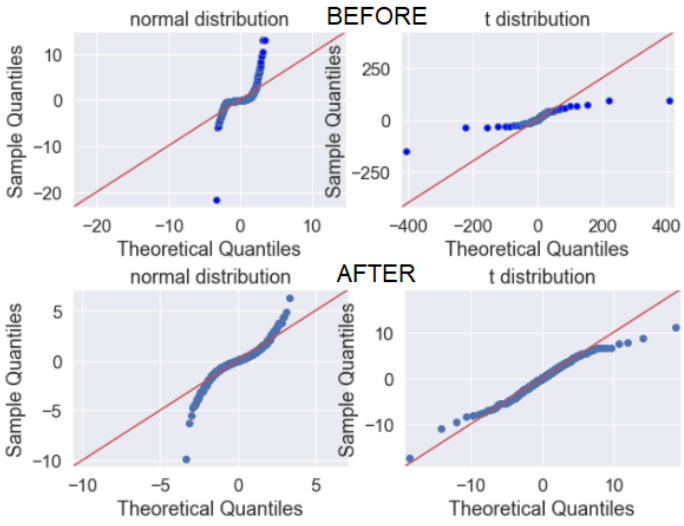


Fig. 28. Ass 1: QQ plot before vs after

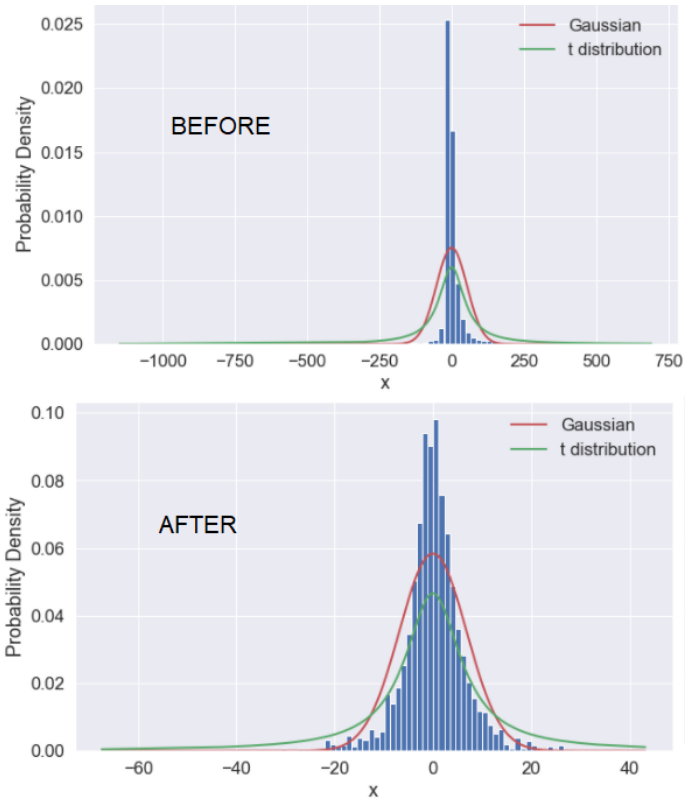


Fig. 29. Ass 1: Residual Plot before vs after

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

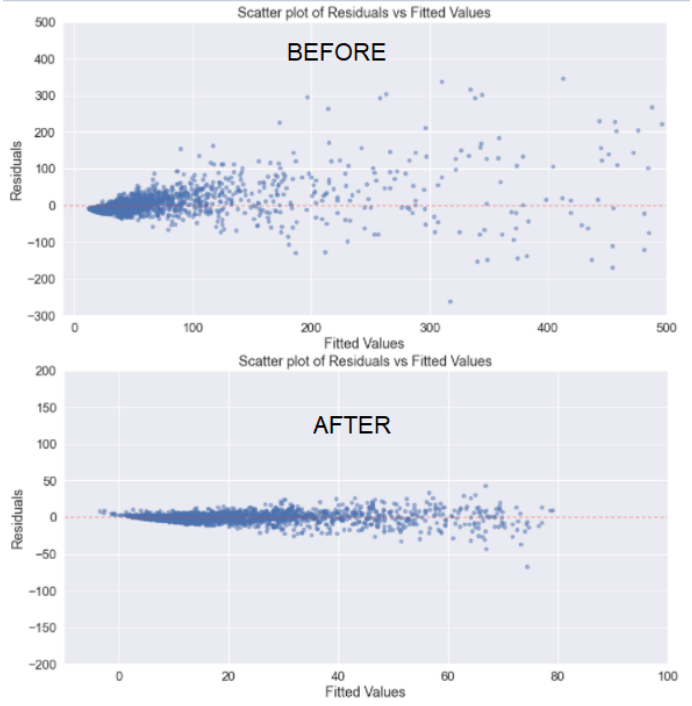


Fig. 30. Ass 1: Residual Scatter before vs after

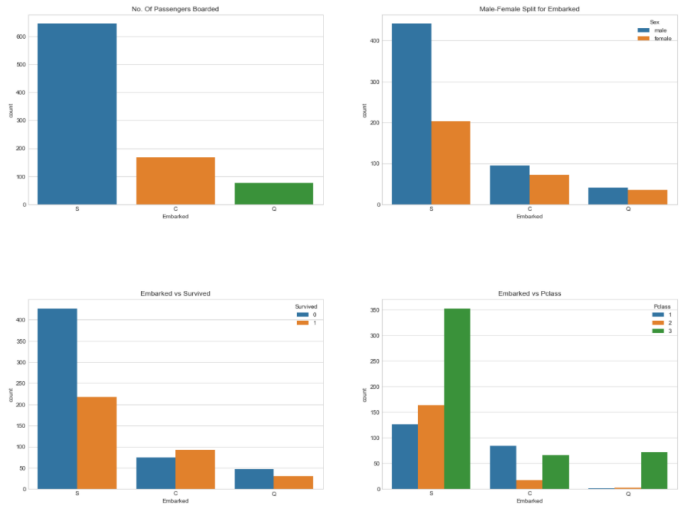


Fig. 31. Ass 2: Embarked

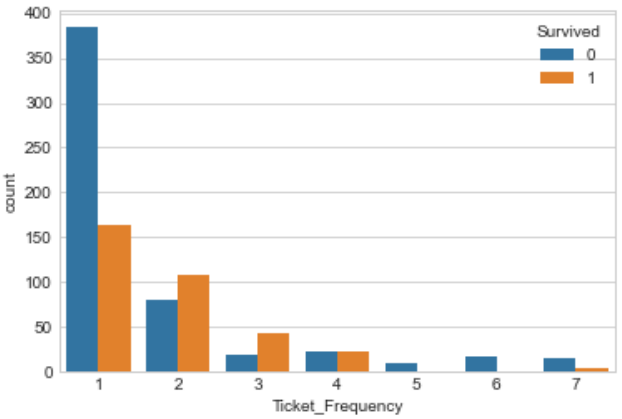


Fig. 32. Ass 2: Count-plot for Ticket Frequency

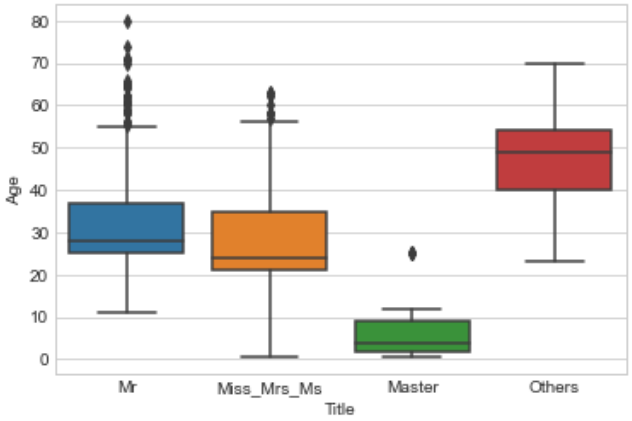


Fig. 33. Ass 2: Boxplot of various titles

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

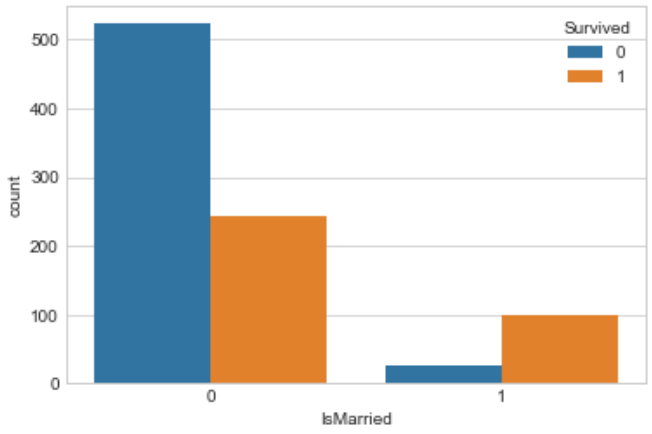


Fig. 34. Ass 2: Countplot - Married or Not

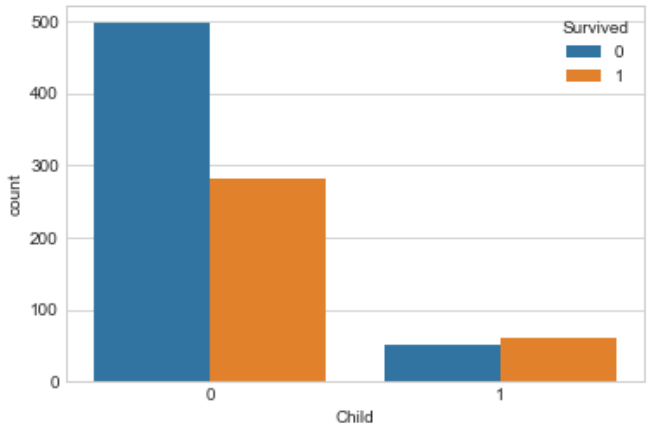


Fig. 35. Ass 2: Countplot - Child or Not

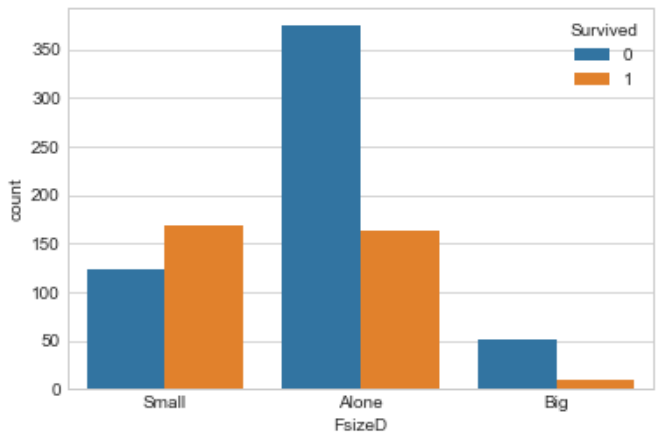


Fig. 36. Ass 2: Countplot for Family Size

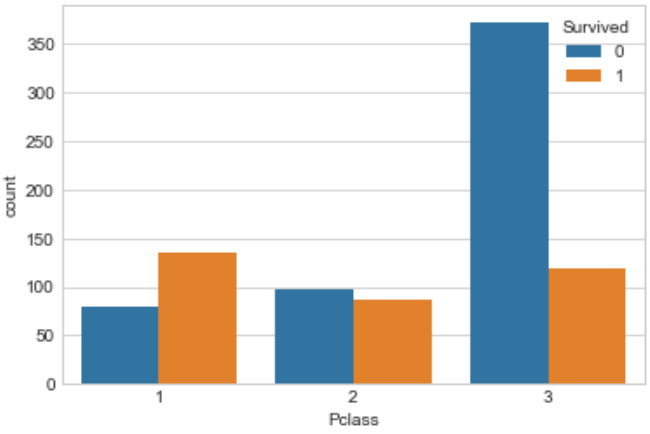


Fig. 37. Ass 2: Countplot for Passenger Class

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

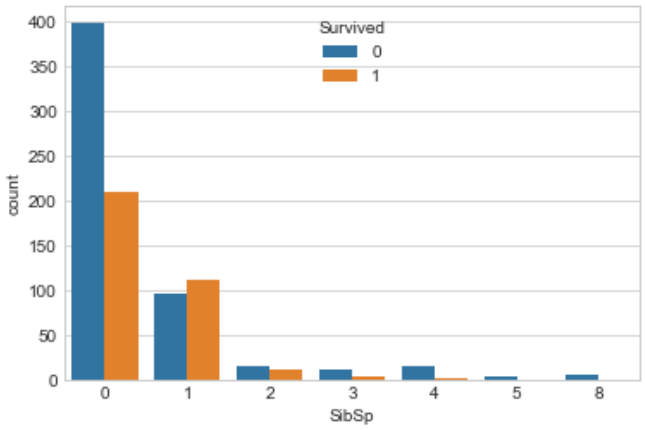


Fig. 38. Ass 2: Countplot - No. of Siblings/Spouse

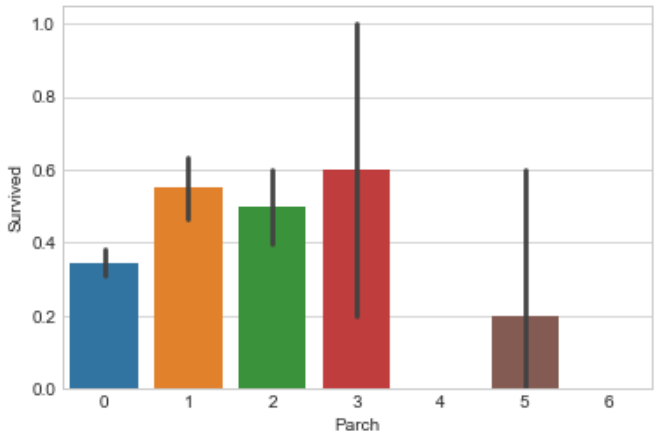


Fig. 39. Ass 2: Probability of Survival vs No. of Parents/Siblings

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

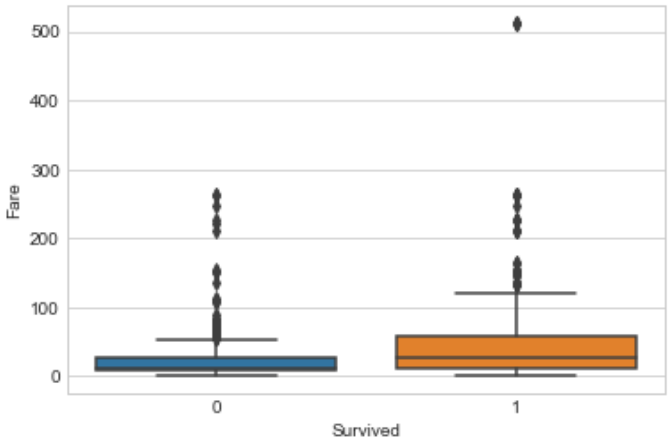


Fig. 40. Ass 2: Boxplot - Fare

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.

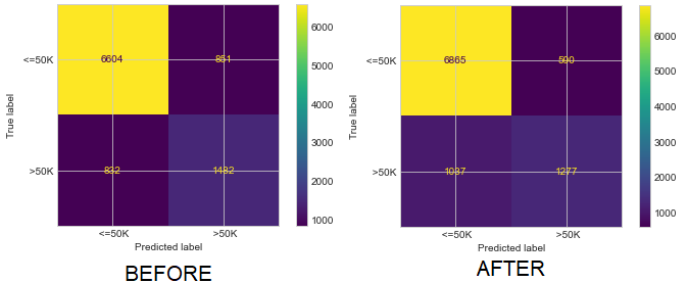


Fig. 42. Ass3: Confusion Matrix Before vs After

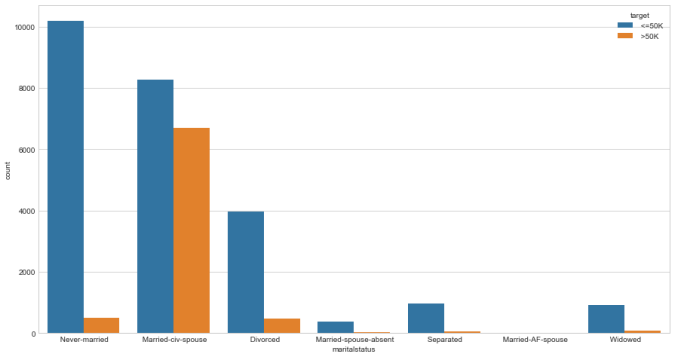


Fig. 43. Ass3: Count-plot Marital Status

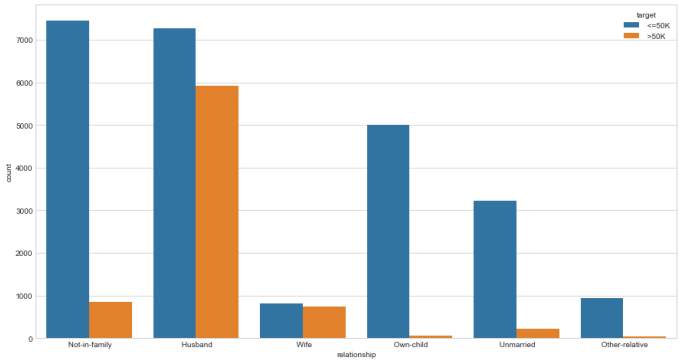


Fig. 44. Ass3: Count-Plot Relationship

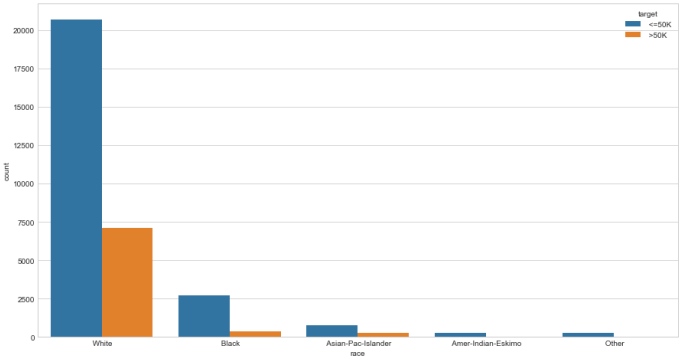


Fig. 45. Ass3: Countplot Race

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

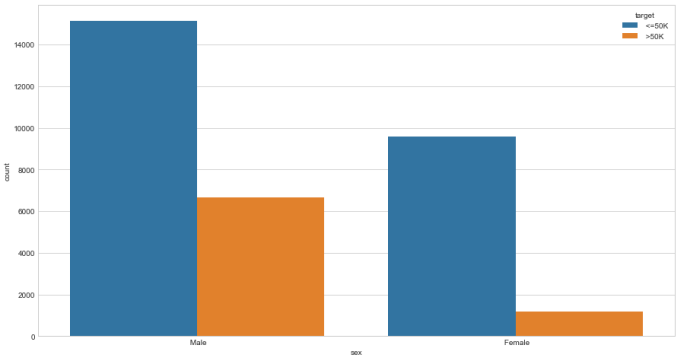


Fig. 46. Ass3: Countplot - Sex

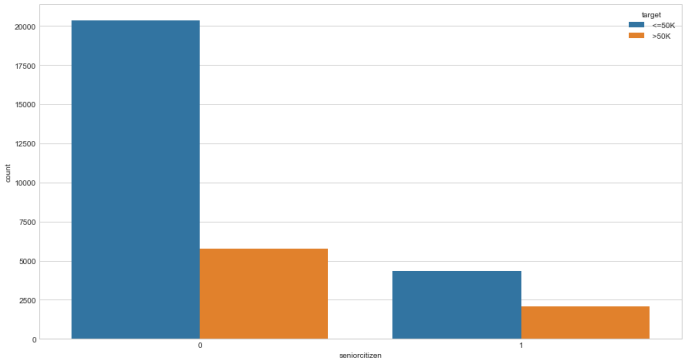


Fig. 48. Ass3: Countplot Senior Citizen

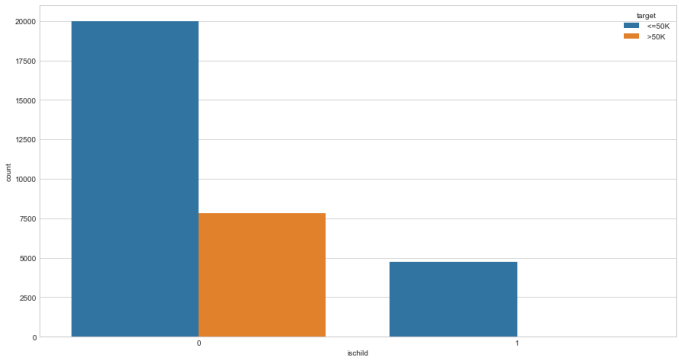


Fig. 47. Ass3: Countplot Children

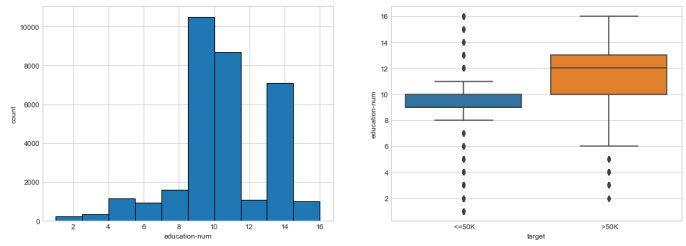


Fig. 49. Ass3: Education Number distribution

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,

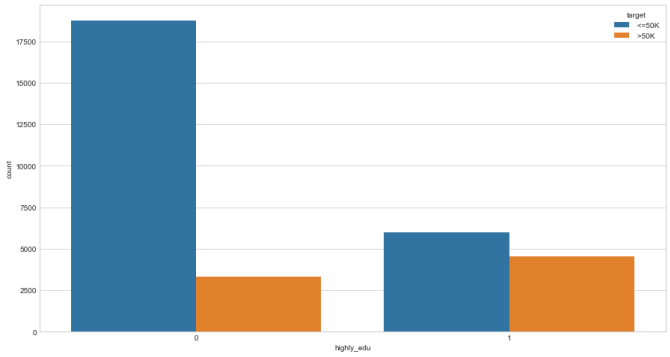


Fig. 50. Ass3: Countplot Highly Educated

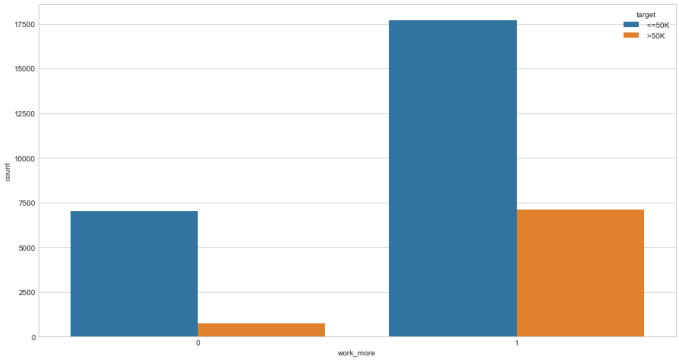


Fig. 52. Ass3: Count-Plot Working More

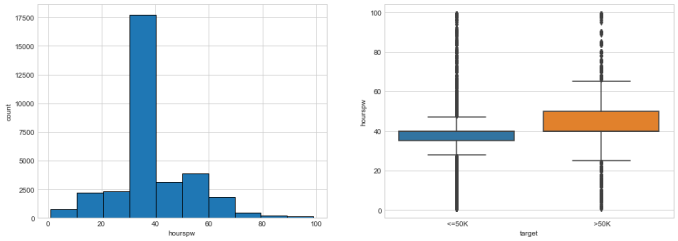


Fig. 51. Ass3: Hours per week distribution

```
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

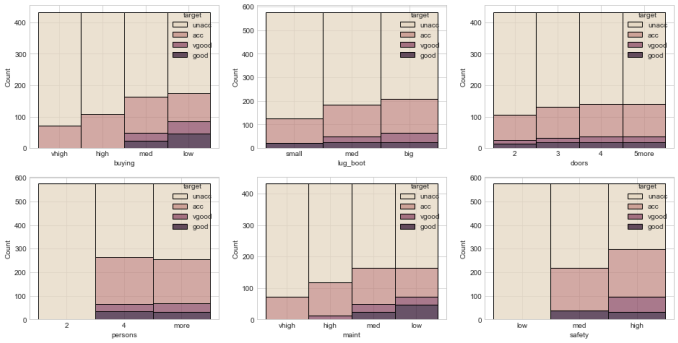


Fig. 55. Ass4&5: Univariate Analysis

Fig. 56. Ass4&5: Multivariate Analysis 1

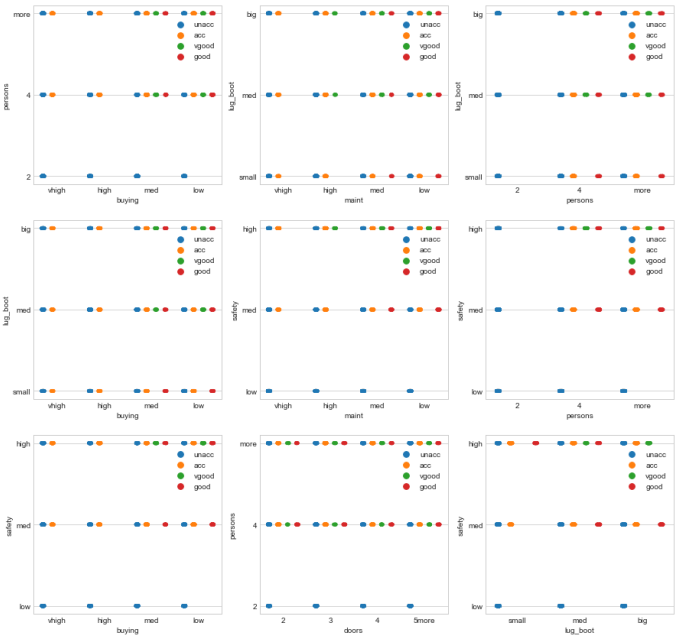


Fig. 57. Ass4&5: Multivariate Analysis 2

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

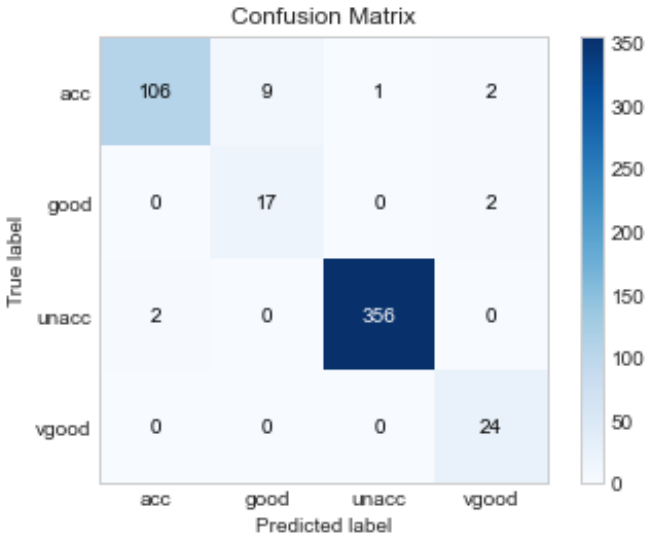


Fig. 58. Ass4&5: Confusion Matrix for Decision Tree

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

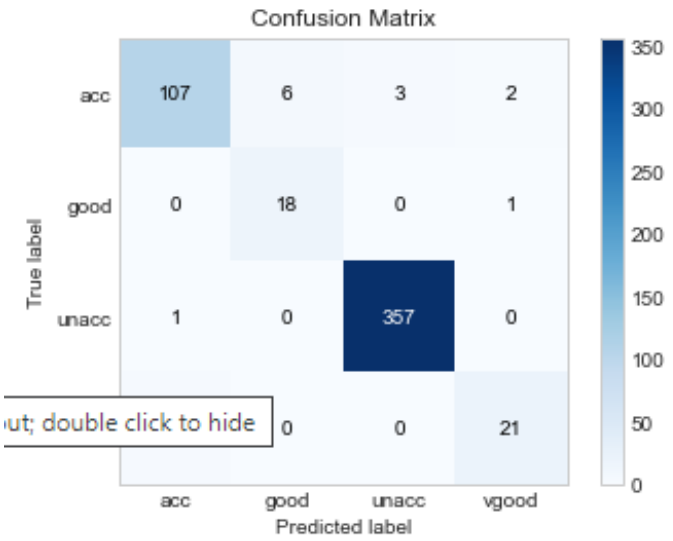


Fig. 60. Ass4&5: Confusion Matrix for Random Forest

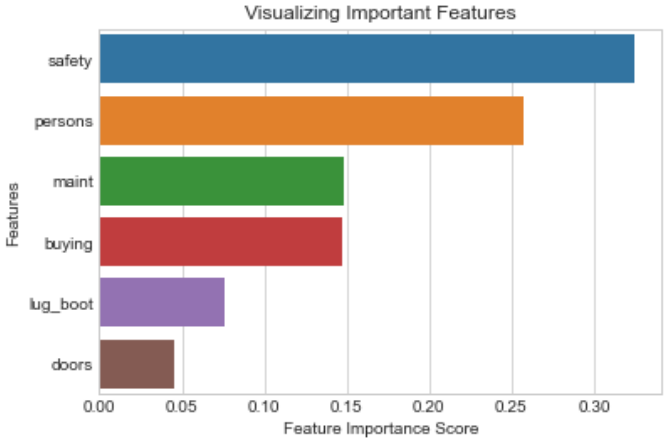


Fig. 61. Ass4&5: Feature Importance chart (higher is better)

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

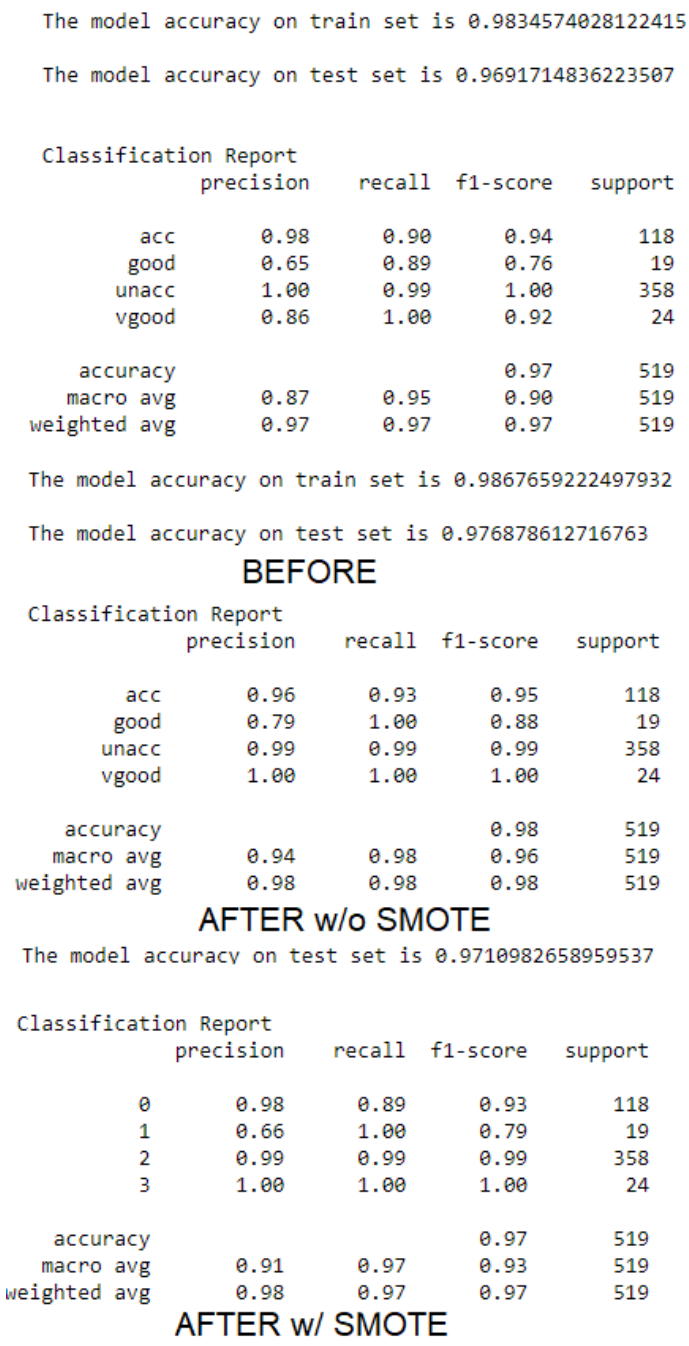


Fig. 62. Ass4: Classification Report Before vs After

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

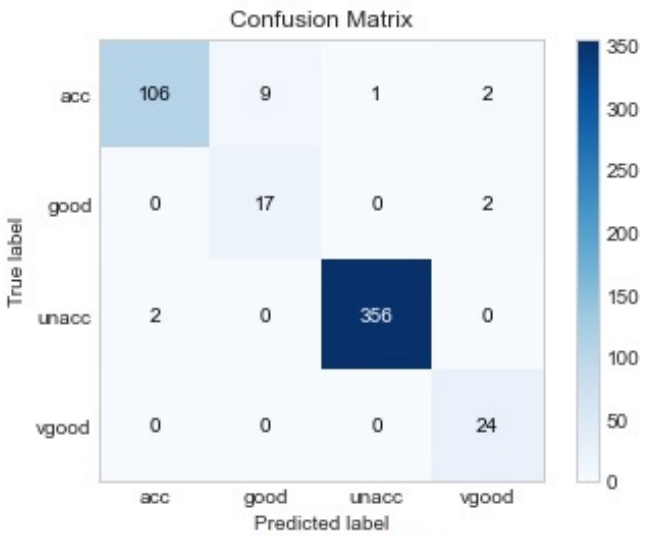
We have divided this paper into two section, one involving a detailed analysis of the stock time series prediction problem and the other part being improving on the already presented problem statements presented during the whole course of this semester. Starting with the stock analysis problem, involving 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

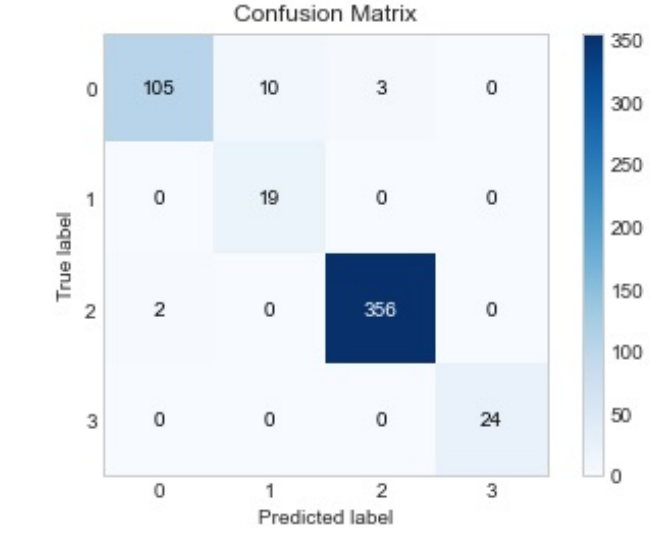


BEFORE

```
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
```



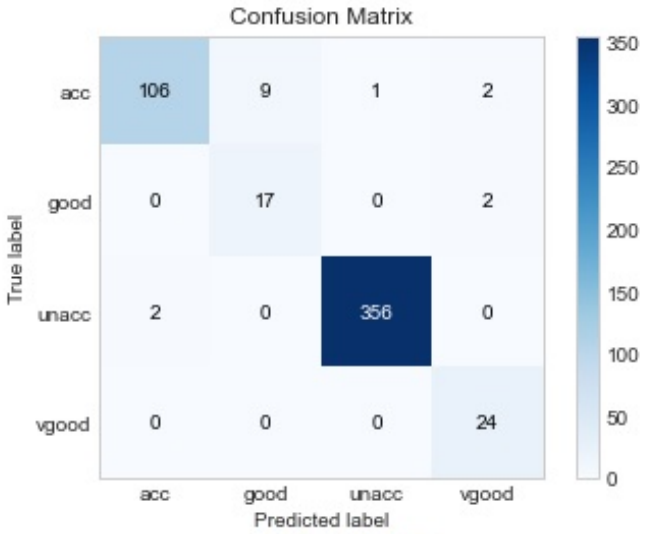
AFTER w/ SMOTE

```
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
```

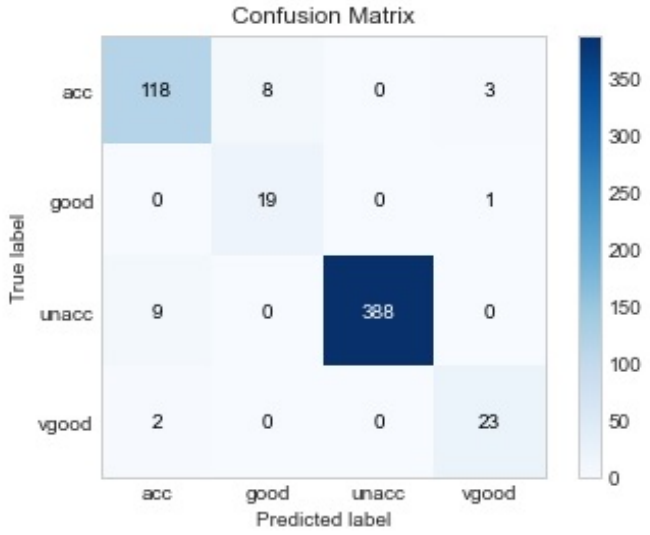
AFTER w/ SMOTE

Fig. 64. Ass5: Classification Report Before vs After

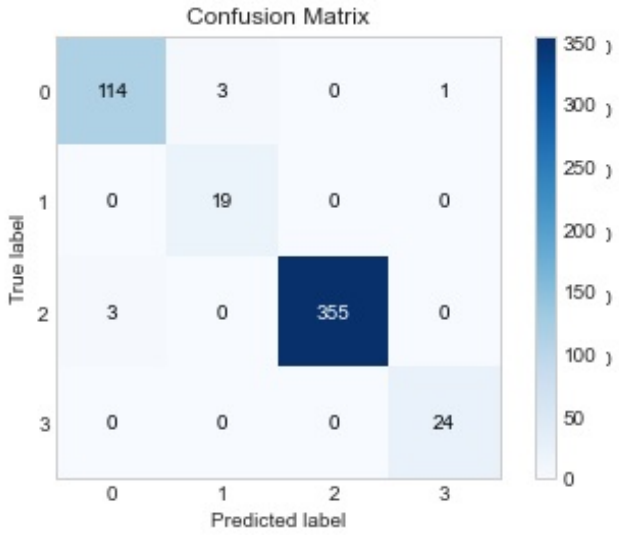
Fig. 63. Ass4: Confusion Matrix Before vs After



BEFORE



AFTER w/o SMOTE



AFTER w/ SMOTE

Fig. 65. Ass5: Confusion Matrix Before vs After

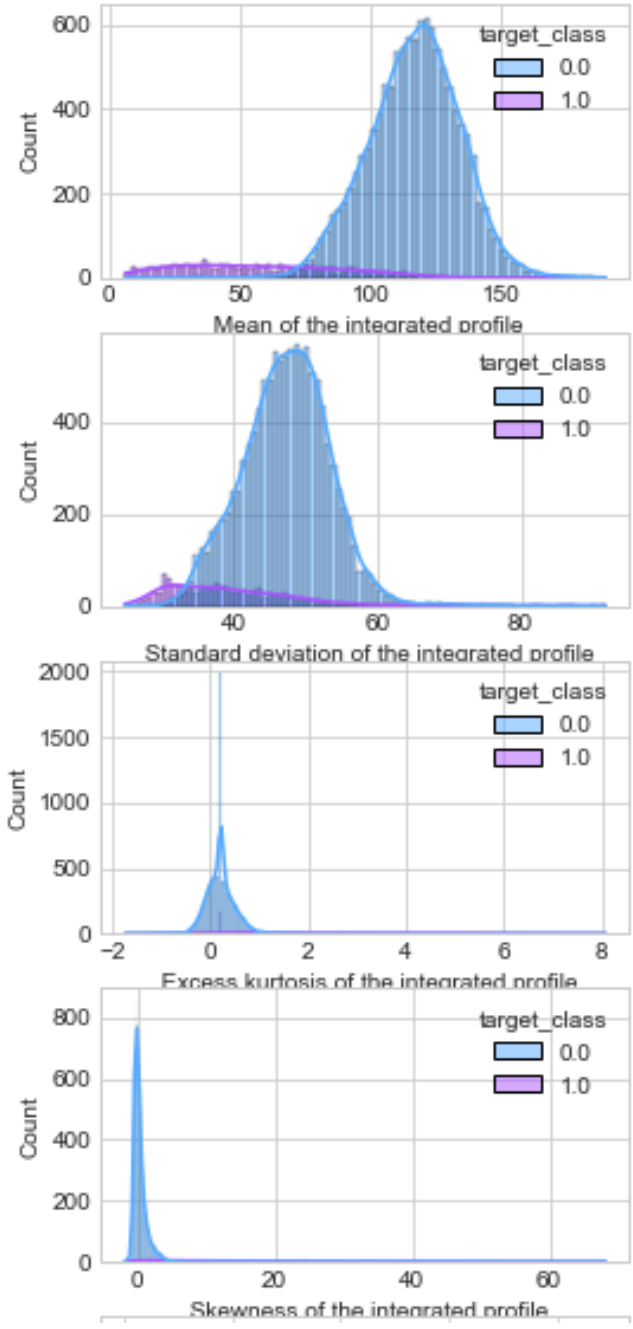


Fig. 66. Ass6: Univariate Analysis

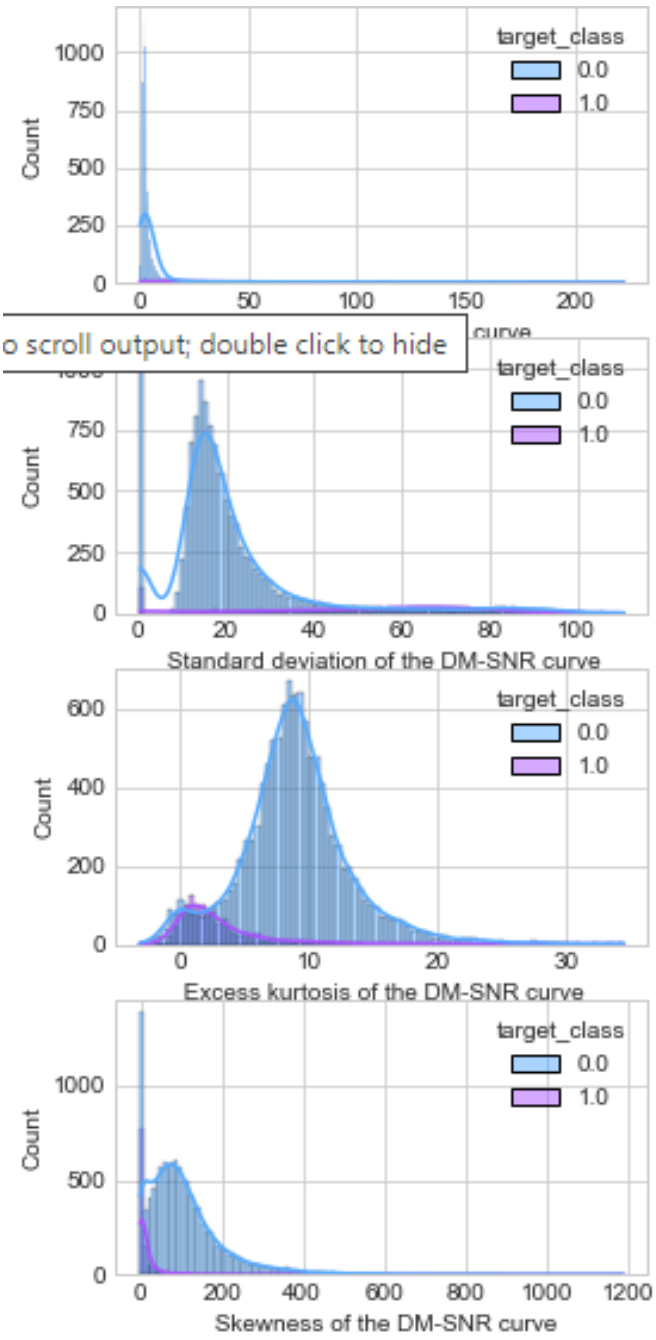


Fig. 67. Ass6: Univariate Analysis

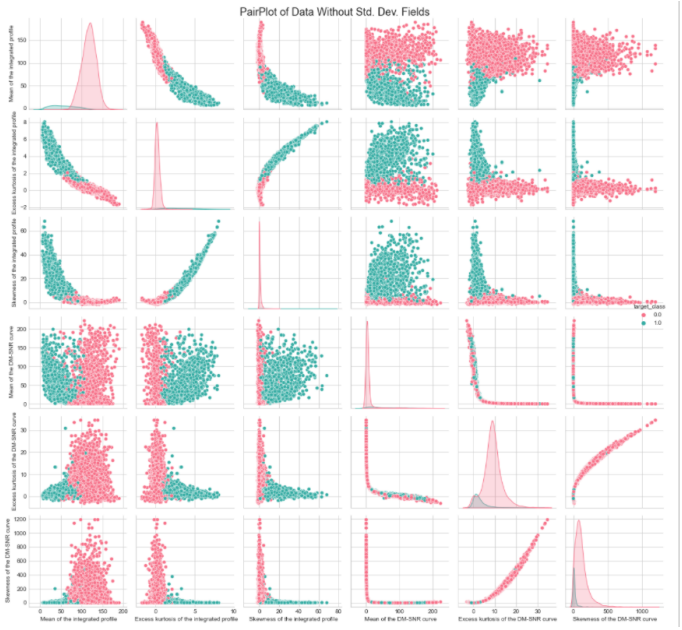


Fig. 68. Ass6: Multivariate Analysis

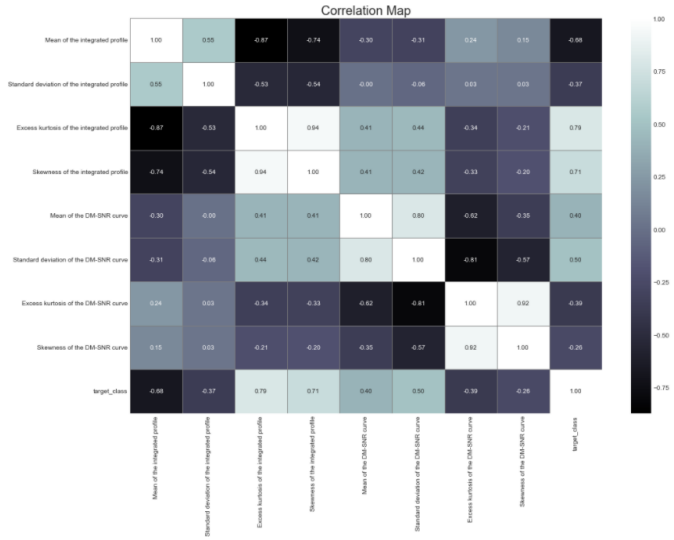


Fig. 69. Ass6: Correlation Map

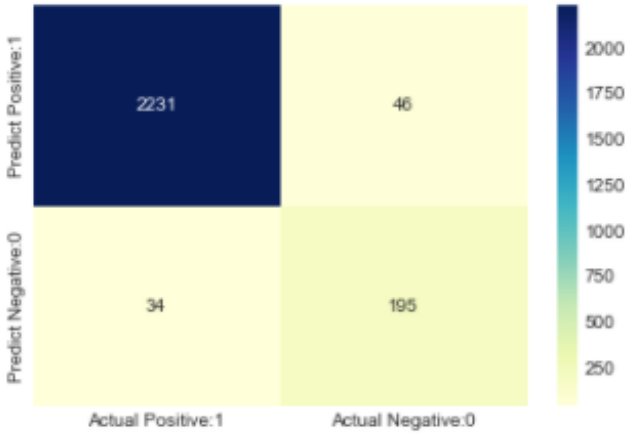


Fig. 70. Ass6: Confusion Matrix

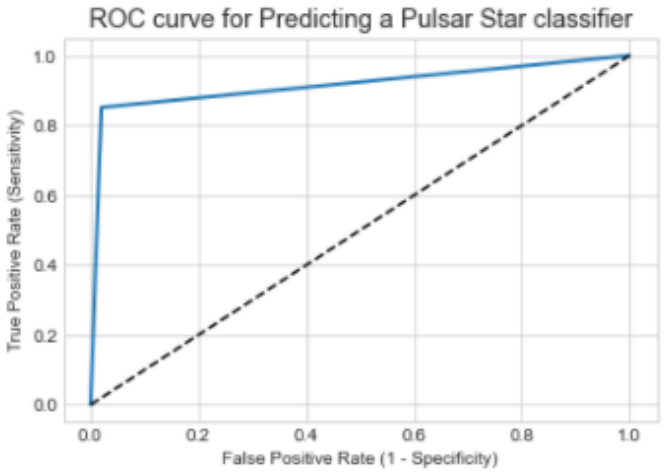


Fig. 71. Ass6: ROC Curve