EE4708: Data Analytics Laboratory

End Semester Examination

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras

A Mathematical Essay on Linear Regression

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras
me18b114@smail.iitm.ac.in

***Abstract*—This paper gives a brief introduction of multivariate linear regression and the basic mathematical theory behind it. We also look at some metrics to evaluate the performance of linear regression on the data-set. Further, we apply linear regression on a real world data-set containing the cancer incidence and mortality rates in various regions of USA, along with social and economic data from that region, to investigate whether the lower income groups are at a higher risk of cancer deaths. We employ exploratory data analysis methods and linear regression to find trends in our data. Our findings show a clear relationship between the socioeconomic status and the cancer mortality rates.**

On the second attempt, the following additions have been made to the model -
1. Regularization Theory.
2. Scaling the data.
3. Applying Regularization and feature selection with lasso regression.
4. Applying Regularization with elastic net regression.
5. Applying XGBoost Regressor on the data.

I. Introduction

Linear regression is a supervised machine learning algorithm used to train a model to predict the behaviour of our data based on some variables. In linear regression, we find out the relationship between the inputs and the target variable by trying to find a best fit line.

In simple linear regression, we model our data in a linear equation –

$$\mathrm{y}_p = \alpha + \beta_1 \mathrm{X}_1 + \beta_2 \mathrm{X}_2 ..... + \beta_k \mathrm{X}_n + \epsilon \tag{1}$$

We try to minimize the least square error of the various data points from our fit line.

$$Minimize \sum_{i=1}^{n} (y_p - y_i)^2 \tag{2}$$

Based on the cancer incidence, death rate and socio-economic data for various US counties, we have to determine whether low-income groups are at greater risk for being diagnosed and dying from cancer, using linear regression.

The paper will first give a brief explanation of linear regression. Then, we will try to solve the problem and model our data-set by
1. Cleaning the data.
2. Dealing with Categorical variables.
3. Exploratory and visual analysis.
4. Choosing the correct variables.
5. Training the model.
6. Evaluating the model.
7. Interpreting the results.

II. Linear Regression Theory

A. Simple Linear Regression

Linear regression is the process of predicting a quantitative response Y on the basis of a single predictor variable X. It assumes that there is approximately a linear relationship between X and Y. We can write this relationship as –

$$\mathrm{y} = \beta_0 + \beta_1 \mathrm{X}_1 + \epsilon \tag{3}$$

The core idea is to obtain a line that best fits the data. The best fit line is the one for which total prediction error (all data points) are as small as possible.

B. Estimating the coefficients

In a linear regression problem, we have n paired data points of (x, y) containing collected data. We try to find the best fit of a line through these points in such a way that the distance of the line from the points in minimized. The most common approach involves minimizing the least squares criterion.

For error of point k_{th}

$$\epsilon_k = (y_k - y_i)^2 \tag{4}$$

On minimizing the sum of errors . we get the following result -

$$\beta_1 = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sum (X_i - \bar{X})^2} \tag{5}$$

$$\beta_0 = \bar{Y} - \beta_1 \bar{X} \tag{6}$$

C. *Evaluating the model*

In model evaluation, we measure the extent to which the model fits the data. The quality of a linear regression fit is typically assessed using two related quantities: the residual standard error (RSE) and the R22 statistic.

1) ***RMSE****:* To calculate the root mean squared error, we first calculate the residual sum of squares that is the sum of squares difference between the ith observed response value and the ith response value that is predicted by our linear model and then we divide it by the number of observations and then take it's square root.
It is one of the most commonly used metric for linear regression.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - y)^2}{n}} \tag{7}$$

2) $\boldsymbol{R^2}$: The R2 statistic provides an alternative measure of fit. It takes the form of a proportion—the proportion of variance explained—and so it always takes on a value between 0 and 1, and is independent of the scale of Y

$$R^2 = 1 - \frac{RSS}{TSS} \tag{8}$$

where

$$TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2 \tag{9}$$

D. *Multivariate Linear Regression*

In practice, we often have more than one predictor. We extend the simple linear regression model so that it can directly accommodate multiple predictors. A multivariate regression is an extension of multiple regression with one dependent variable and multiple independent variables. Based on the number of independent variables, we try to predict the output.

$$\mathrm{y} = \beta_0 + \beta_1 \mathrm{X}_1 + \beta_2 \mathrm{X}_2 ..... + \beta_k \mathrm{X}_n + \epsilon \tag{10}$$

To estimate the coefficients of multivariate regression, we use the same least square error method to minimize the following equation -

$$\sum_{i=1}^{n}(\mathrm{y}_i - \beta_0 + \beta_1 \mathrm{X}_1 + \beta_2 \mathrm{X}_2 ..... + \beta_k \mathrm{X}_n)^2 \tag{11}$$

III. THE PROBLEM

A NGO is advocating for better health outcomes for low-income populations in the United States. To help with lobbying and fundraising, we need to examine and analyse whether low-income groups are at greater risk for being diagnosed and dying from cancer.

A. *The Variables*

The data given to us has these significant variables -
FIPS - Federal Information Processing Standard. Each area has a unique code.
Incidence rates - Cases per 100,000 population per year which are age-adjusted to the 2000 US standard population.
Mortality Rate - The measure of number of deaths.
All Poverty - Population For whom income in the past 12 months is below poverty level.
M Poverty - Male Population For whom income in the past 12 months is below poverty level.
F Poverty - Female Population For whom income in the past 12 months is below poverty level.
Med Income - Median household income in the past 12 months.
Med Income X - Median household income in the past 12 months for the X ethnicity (X - White, Black, Asian, Hispanic etc.).
All With - Population covered by health insurance.
All Without - Population not covered by health insurance.
M With, M Without - Male population covered and not covered by health insurance respectively.
F With, F Without - Female population covered and not covered by health insurance respectively.
Avg Ann Deaths - Average lung cancer mortalities.
Avg Ann Incidence - Average lung cancer incidence rate.

Population Data - We acquired a data-set from data.world for population estimate in counties in 2015, the year our dataset was collected We will need this data to normalize our variables.

Finally, we merge our socio-economic ,population and mortality rate data to get a merged dataframe with all suitable variables.

B. *Cleaning the Data*

We check for null values in all columns and find that the a significant percentage of Median Income data for various ethnicities is missing. So, we decide to drop these columns and only retain the overall median income data.

We check for non-numeric values in our data and notice that the following variables contain strings.
Incidence Rate - 499 strings including *, -
Mortality Rate and Avg. Ann Deaths – 325 strings including *,-

Since Mortality Rate is the target variable, we will drop all the string columns present in it. For Mortality Rate and Avg. Ann Deaths, we first remove the strings and impute the missing values with median values.

C. *Categorical Variables*

We have the recent trend variable which we have to convert to numerical values using dummy variables.

We have the following strings in recent trend so we can use 0 and 1 in place of them –
Falling – 197
Rising – 39

D. *Normalizing by population*

We create new columns with Population Normalization for Poverty and Health Insurance Data. We use the Population estimates we acquired.

$$Variable.PN = \frac{Variable}{PopulationEstimate} * 10^5 \qquad (12)$$

E. *Visualising and selecting data*

1) ***Income and Poverty*** *:* We have four variables relating to poverty data – All Poverty, M and F Poverty and Median Income.
We draw a pair plot and correlation heat map between these variables. (Fig.1 and 2)

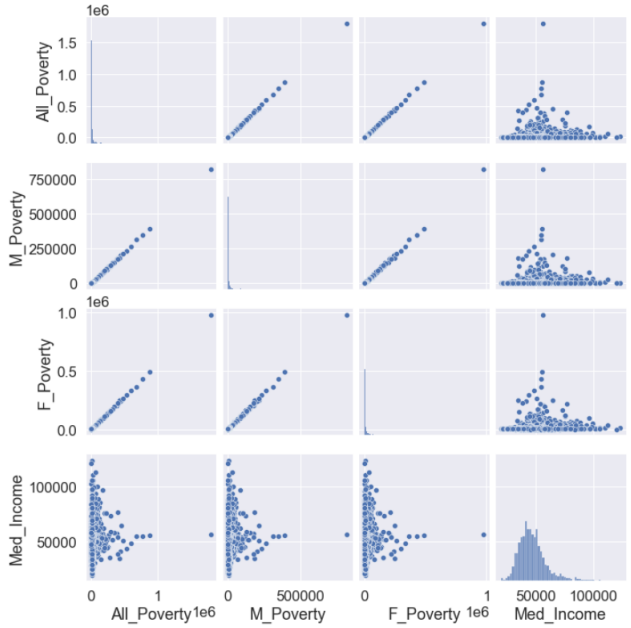

Fig. 1. Poverty Variables Pairplot

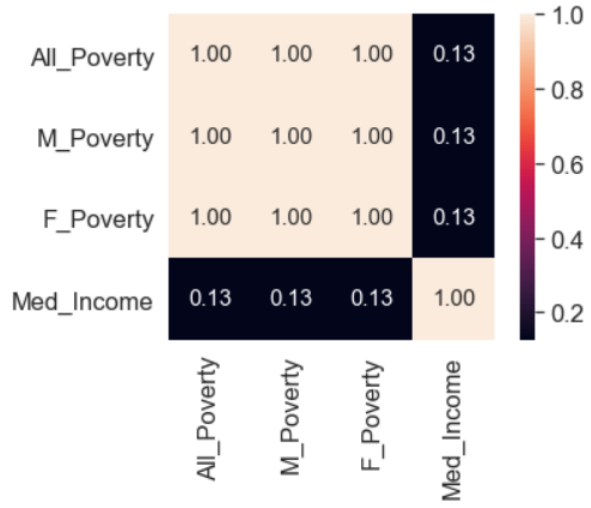

Fig. 2. Poverty Variables Heatmap

We can drop M Poverty and F Poverty as we see very high correlation between the gender related Poverty data. We retain the All Poverty column.

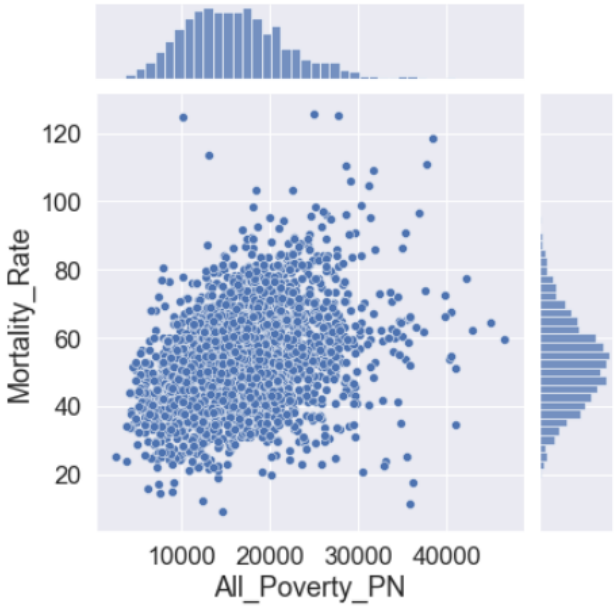

Fig. 3. Mortality Rate vs All Poverty

Through the scatterplot between Mortality Rate and All Poverty (Fig.3), we can see a clear positive relationship between them.

Through the scatterplot between Mortality Rate and Median Income (Fig. 4), we can see a clear negative relationship between them.

2) ***Health Insurance****:* We have 6 variables relating to poverty data – All With and Without, M and F With and

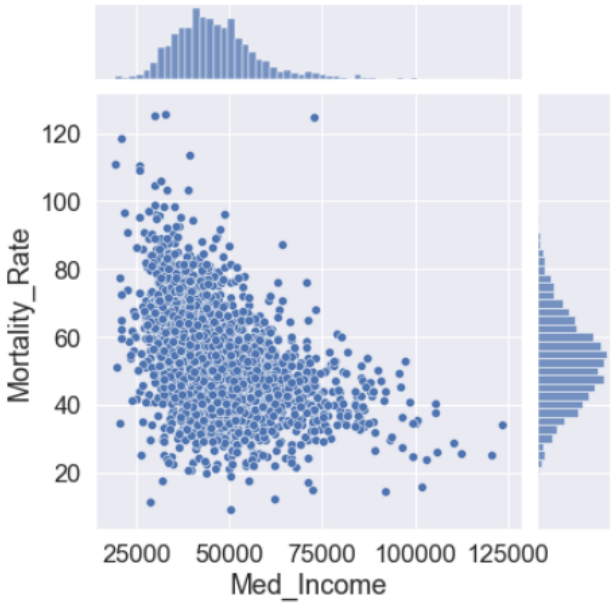

Fig. 4. Mortality vs Med Income

Without.

We draw a pair plot and correlation heatmap between these variables. (Fig. 5 and 6)

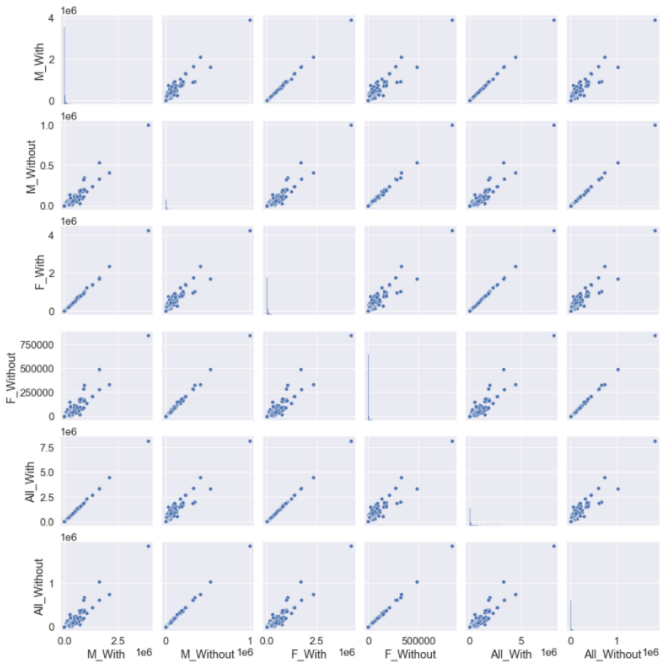

Fig. 5. Health Insurance Pairplot

Similar to poverty data, we can drop the gender related data as we see very high correlation between them. We retain the All With and All Without column.

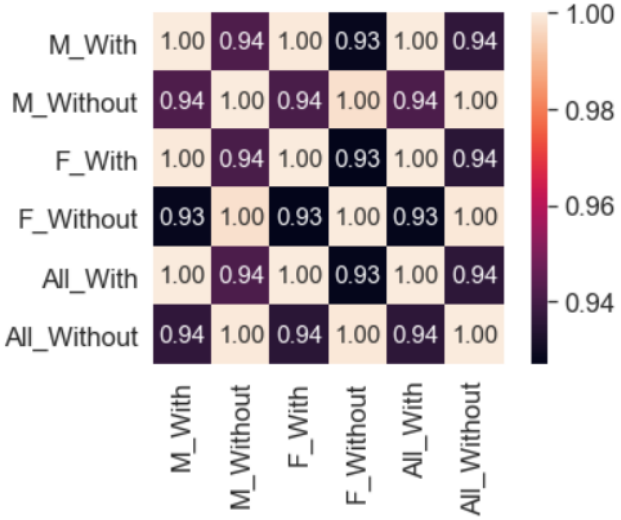

Fig. 6. Health Insurance Variables Heatmap

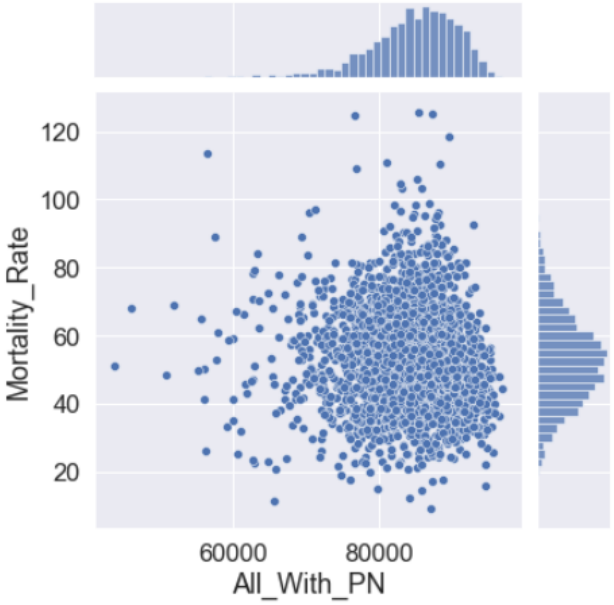

Fig. 7. Mortality vs All With Health Insurance

Through the scatterplot between Mortality Rate and Health Insurance Data (Fig. 7 and 8), we witness some pattern between mortality and health insurance data and therefore it is advisable to include them in our model.

*3) **Death and Incidence Rate**:* These variables are closely interrelated so we should keep only one of these variables. Our hypothesis is also confirmed by the correlation heat map and pair plot. (Fig. 9 and 10). We keep only incidence rate for our model.

This also indicates high correlation between Incidence Rate and Mortality Rate. We can use either of these variables in

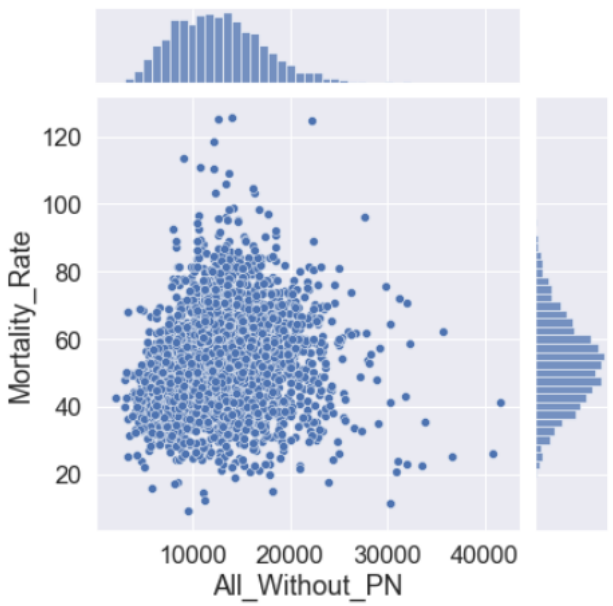

Fig. 8. Mortality vs All Without Health Insurance

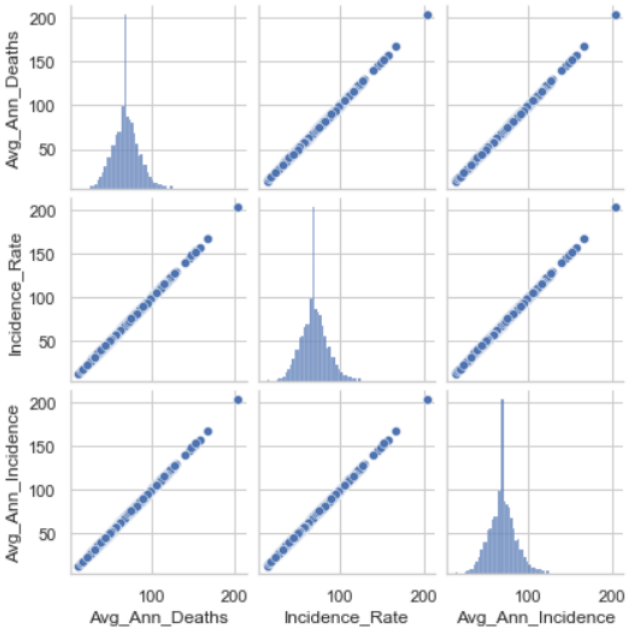

Fig. 9. Incidence Variables Pairplot

our regression model.

There is a definite positive correlation with mortality rate, as visible in scatterplot. (Fig. 11)

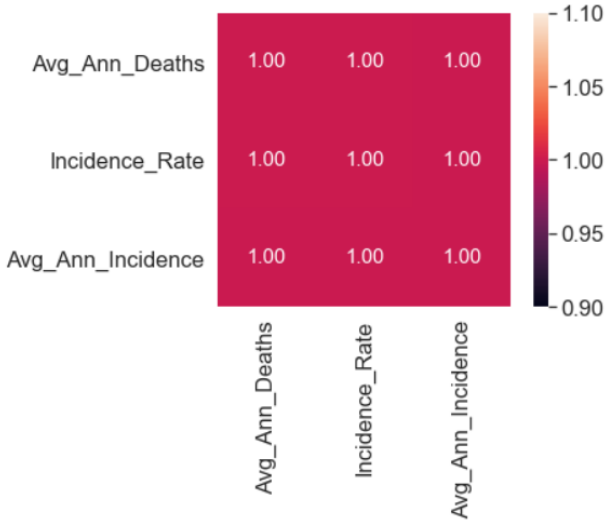

Fig. 10. Incidence Variables Heatmap

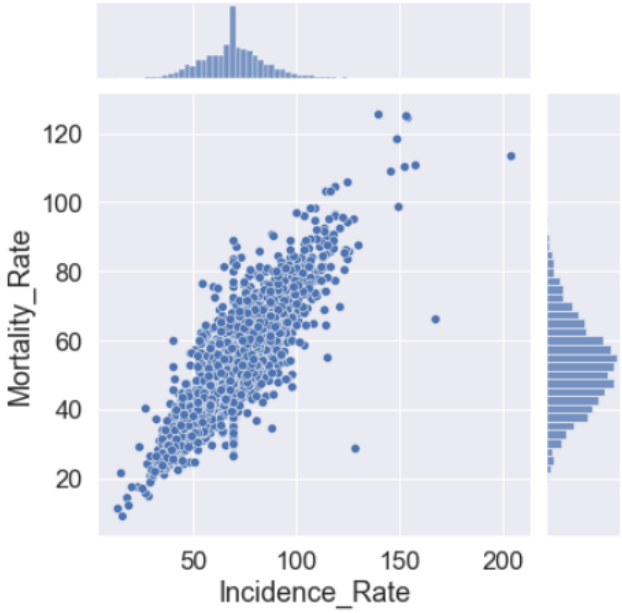

Fig. 11. Mortality Rate vs Incidence Rate

F. Visualizing incidence rate with poverty and income variables

Incidence rate represents the number of cancer cases reported in a particular area. We can also visualize incidence rate's relation to poverty and income data and check for any trends.

Through the below given plot between Incidence Rate and Median Income (Fig.12), we can see a slight negative trend, which confirms our suspicion that lower income groups are more vulnerable to cancer.

Similarly, through the plot between Incidence Rate and Poverty stats (Fig. 13), we can see a positive trend which

shows that in areas with higher number of people below the poverty line, we see higher incidence of cancer cases

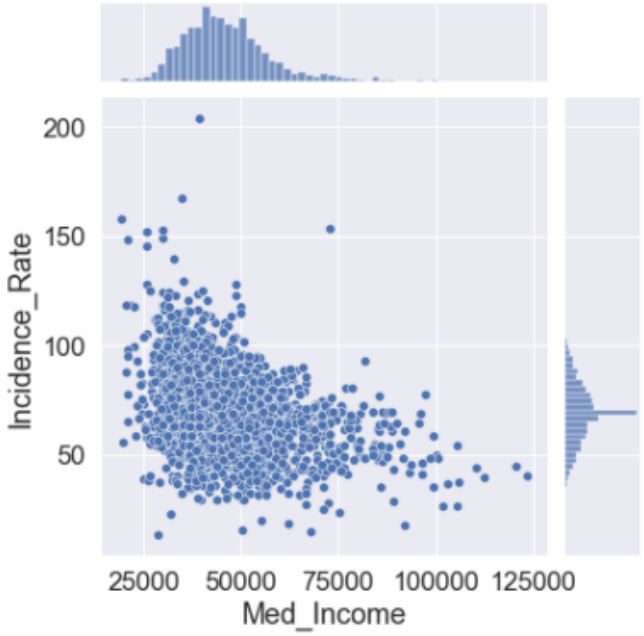

Fig. 12. Incidence Rate vs Median Income

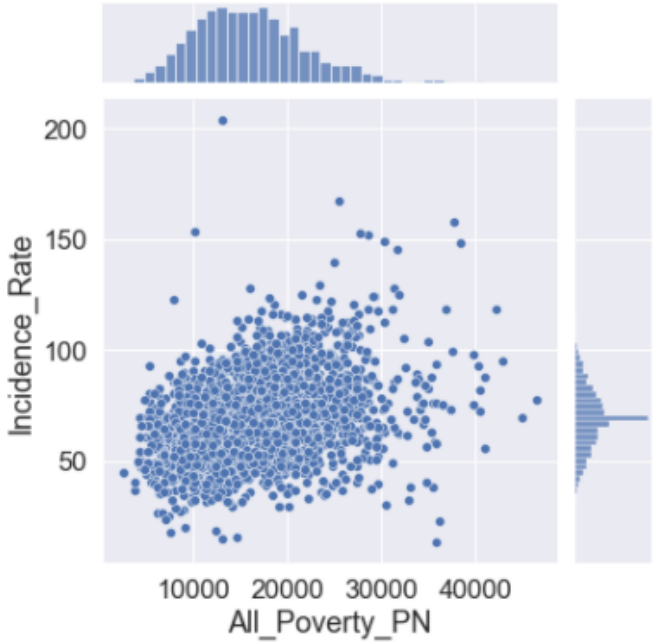

Fig. 13. Incidence Rate vs Poverty

G. *Final Variables for our model*

We have these final variables with us - 'All Poverty PN', 'Med Income', 'All With PN', 'All Without PN', 'Incidence Rate','rising' and 'falling'.

We will draw final correlation heat map to ensure no correlated variables.

We find no correlated variables and therefore, we are ready to go ahead with the model. (Fig. 12)

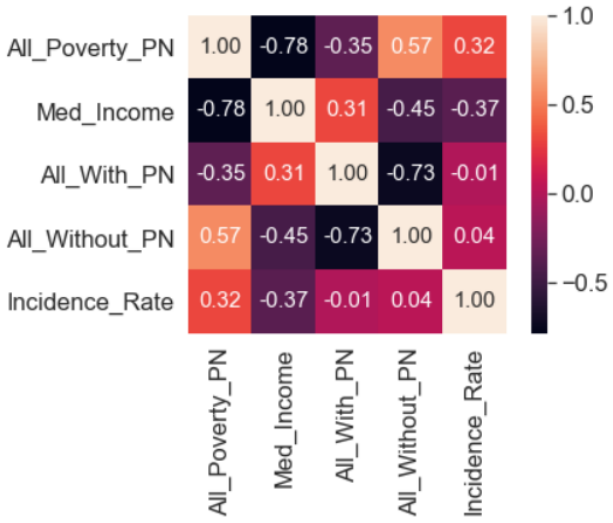

Fig. 14. Final Variables Pairplot

IV. The Model

A. *Fitting our Data*

We have the following X and y for our final model.

X= All Poverty PN , Med Income, All With PN, All Without PN, rising , falling.
y = Mortality Rate.

Using scikit-learn library's LinearRegression() function on our X and y variables, we develop a multivariate linear regression model -

$$\begin{aligned}\text{Mortality.Rate}_i = \beta_0 &+ \beta_1 \text{All.Poverty.PN}_i \\ &+ \beta_2 \text{Med.Income}_i \\ &+ \beta_3 \text{All.With.PN}_i \\ &+ \beta_4 \text{All.Without.PN}_i \\ &+ \beta_5 \text{rising}_i \\ &+ \beta_6 \text{falling}_i + \epsilon\end{aligned}$$

B. *Intercepts and Coefficients*

After fitting our data, we get the following values - Intercept -

$$\beta_0 = 6.144721764929308 \tag{13}$$

The variable-wise coefficients are tabulated below

	Coefficient
All_Poverty_PN	0.000057
Med_Income	-0.000116
All_With_PN	0.000028
All_Without_PN	0.000240
Incidence_Rate	0.656103
rising	-0.986183
falling	0.682052

Fig. 15. The Coefficients of Model

C. *Interpretations of the coefficients in the model*

*1) **Poverty and Income Variables**: -*

Med Income = -0.000116

We have a strong negative relationship. This can be interpreted as on 1000 dollar decrease in median income of the region, mortality rate goes up by 1.16 . Therefore we can report the lower income groups are having a higher mortality rate and in danger.

All Poverty PN = 0.000057

We have a positive coefficient which can be interpreted as if the number of people below the poverty line in an area increases, the risk of cancer death is higher.

*2) **Health Insurance Data**: -*

All Without = 0.00024

We have a positive coefficient which means that if the area has higher number of people without health insurance, the risk of cancer death is higher. Having an insurance can affect the treatment a person gets and therefore we witness a higher coefficient for All Without (0.000240) than All With (0.000028). This adds weight to the fact that lower income groups who can't afford health care are at a greater risk.

*3) **Incidence Rate**: -*

Incidence Rate = 0.656103

As we saw before, the incidence and mortality rate are highly correlated as logically, more instances of cancer cases means more deaths.

D. *Model Evaluation and Metrics*

The Root Mean Squared Error (RMSE) of the best model (XGBoost) is 2.561417394994466

The R2 score for the best model comes out to be 0.9667146856907032

V. Improvements in the model

A. *Ridge Regression or L2 regularization*

Regularization is a technique used to reduce the errors by fitting the function appropriately on the given training set and avoid over fitting. The regularization term, or penalty, imposes a cost on the optimization function.

In ridge regression, the cost function is altered by adding a penalty equivalent to square of the magnitude of the coefficients. The cost function -

$$(Y - X\beta)^T(Y - X\beta) + \lambda\beta^T\beta$$

Ridge regression puts constraint on the coefficients (w). The penalty term (lambda) regularizes the coefficients such that if the coefficients take large values the optimization function is penalized. So, ridge regression shrinks the coefficients and it helps to reduce the model complexity and multi-collinearity.

B. *Lasso Regression or L1 regularization*

In lasso regression, instead of taking the square of the coefficients, magnitudes are taken into account.This type of regularization (L1) can lead to zero coefficients i.e. some of the features are completely neglected for the evaluation of output. So Lasso regression not only helps in reducing over-fitting but it can help us in feature selection. The cost function -

$$(Y - X\beta)^T(Y - X\beta) + \lambda|\beta|_1 \tag{14}$$

C. *Elastic Net*

Elastic net is a popular type of regularized linear regression that combines two popular penalties, specifically the L1 and L2 penalty functions. The cost function is -

$$(Y - X\beta)^T(Y - X\beta) + \lambda_1|\beta|_1 + \lambda_2|\beta|_2{}^2 \tag{15}$$

D. *XGBoost*

XGBoost stands for eXtreme Gradient Boosting.

It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework.

XGBoost is an ensemble tree method that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture.

XGBoost uses the following optimization techniques -

- **Parallelization** - XGBoost approaches the process of sequential tree building using parallelized implementation. This is possible due to the interchangeable nature of loops used for building base learners; the outer loop that enumerates the leaf nodes of a tree, and the second inner loop that calculates the features.

- **Tree Pruning** - The stopping criterion for tree splitting within GBM framework is greedy in nature and depends on the negative loss criterion at the point of split. XGBoost uses 'max depth' parameter as specified instead of criterion first, and starts pruning trees backward.

- **Hardware Optimization** - This algorithm has been designed to make efficient use of hardware resources. This is accomplished by cache awareness by allocating internal buffers in each thread to store gradient statistics.

E. Scaling the data

Scaling refers to the process of putting values in the same range or scale such that no one variable dominates the others. Many Machine Learning algorithms use distance between two data points and if the features have varying magnitudes.

We have used scikit-learn's StandardScaler method to scale our data. StandardScaler removes the mean and scales each feature to unit variance. This operation is performed feature-wise in an independent way.

F. Applying Regularization

*1) **Lasso or L1 regularization**:* On applying feature selection with Lasso regression, we get All Poverty PN, All With PN, All Without PN, rising , falling's coefficient as 0, implying that they are not significant features in our model. The R2 score achieved by lasso regression is 0.7216650590301876

	Coefficient
All_Poverty_PN	0.000000
Med_Income	-1.550058
All_With_PN	-0.000000
All_Without_PN	0.000000
Incidence_Rate	11.532910
rising	-0.000000
falling	0.000000

Fig. 16. Lasso Coefficients

*2) **Elastic Net regularization**:* The elastic net model doesn't give good results with a low R2 score of 0.624.

G. Applying XGBoost

XGBoost is one of the most powerful regression algorithms available and it works the best for our data. It gives a R2 score of 0.96, outperforming all other models.

VI. Conclusions

According to our findings, the incidence rate and mortality rate are linked to the socioeconomic background. As we expected, the incidence rate and mortality rate are highly correlated as more cases mean more death instances. We have both visual and mathematical evidence to support our claim.

Through mathematical evidence based on our linear regression model with mortality rate as our target variable, we can report-

- A negative coefficient for median income, indicating that lower income groups have higher mortality risk.

- A positive coefficient for poverty statistic, indicating that areas with higher number of people below poverty line have higher mortality rate.

- A higher positive coefficient for people without health insurance than those with health insurance, indicating that people who cannot afford insurance are at higher risk.

After Improvements-

- Using lasso regression, we find that Median Income and Incidence Rate are the most significant features of the model.
- XGBoost regressor gives the best R2 score, 0.966, for our model.

Through visual evidence ,the data indicates a negative exponential link between death /incidence rate and median income. This can be explained by the fact that initially as the income decreases, the death rate goes up, but after a certain threshold the curve flattens down as lowering the income further won't affect the treatment and lifestyle.

The positive link between death/ incidence rate and people below poverty line also has a similar explanation.

Therefore we can conclude that lower income groups have higher incidence and mortality rates and they are definitely at an higher risk of cancer.

References

[1] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013) An Introduction to Statistical Learning with applications in R, www.StatLearning.com, Springer-Verlag, New York

[2] Swaminathan, S. (2019, January 18). Linear regression - detailed view. Medium. Retrieved September 16, 2021, from https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86.

[3] Bureau, U. S. C. (2020, February 13). Acs poverty data tables. The United States Census Bureau. Retrieved September 16, 2021, from https://www.census.gov/topics/income-poverty/poverty/data/tables/acs.html.

[4] US population Estimates 2015 - dataset BY NRIPPNER. data.world. (2017, January 27). Retrieved September 17, 2021, from https://data.world/nrippner/us-population-estimates-2015/.

A Mathematical Essay on Logistic Regression

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras
me18b114@smail.iitm.ac.in

***Abstract*—This paper gives a brief introduction of multivariate logistic regression and the basic mathematical theory behind it. We also look at some metrics to evaluate the performance of logistic regression models. Further, we apply logistic regression on a real world data-set containing the socio-economic details of the passengers who were on the RMS Titanic ship, which sank after colliding with an iceberg, resulting in the death of 1502 out of 2224 passengers. Through exploratory data analysis and predictive logistic regression model, we investigate trends in data to predict which passengers were more likely to survive based on their age, fares, passenger class, embarkation point etc. Then we evaluate our model using confusion matrix and F1 score. Our findings reveal some interesting trends.**

On the second attempt, the following additions have been made to the model -
1. Random Search and XGBoost theory.
2. Scaling the data.
3. Dealing with outliers.
4. Balancing the target class with SMOTE.
5. Applying XGBoost on the data.
6. Hyper parameter tuning with Random Search.

I. Introduction

There are multiple types of algorithm methods used in machine learning. One such popular and commonly used machine learning method is logistic regression. It is a supervised machine learning algorithm used to train a model to predict the behaviour of our data based on given parameters. It is used to estimate the relationship between a dependent (target) variable and one or more independent variables.

Logistic regression is a classification algorithm which models the probabilities with two possible outcomes. In this algorithm, the Logit function is used for calculating the log odds ratio, which in turn gives the likelihood of that particular outcome. A general equation for logistic regression can be written as -

$$\ell = \log_b \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n \qquad (1)$$

where,
p is the probability of event happening.
$\beta_0, \beta_1, ...\beta_n$ are the coefficients of the model.
To calculate these coefficients, we use Maximum Likelihood Estimation with log-likelihood as the cost function.

In this paper, we apply logistic regression to the data-set containing socio-economic data of all the passengers on the RMS titanic ship which sank on April 15, 1912. We are given various details of the passenger like their age, ticket fare, passenger class, embarkation point, sex etc. We are required to find out if it was just luck or is there is any relation between the above mentioned factors and the survival of the passenger.

The paper will first give a brief explanation of logistic regression. Then, we will try to solve the Titanic problem and model our data-set by
1. Cleaning the data.
2. Feature engineering for categorical variables.
3. Exploratory and visual analysis.
4. Checking for Correlation.
5. Dealing with outliers.
6. Scaling the Data.
7. Dealing with Imbalance.
8. Training the model.
9. Evaluating the model.
10. Interpreting the results.

II. Logistic Regression Theory

A. *Classification*

Classification is a process of categorizing a given set of data into classes. Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category, label or class.
Many real world problems require classification rather than regression. In the classification setting we have a set of training observations (x1, y1), . . . , (xn, yn) that we can use to build a classifier.

Linear regression is not useful in problems like if we are trying to predict the medical condition of a patient in the hospital and we have 3 possible diagnoses- stroke, drug overdose, and epileptic seizure based on the symptoms of the patient. We can label these diseases at 1,2 and 3. In this case, even if we create a metric to accurately convert the symptoms to numeric data, regression won't be able to classify the data

as it cannot give the output as 1,2 or 3. Alternatively, we can create a threshold that if predicted value is between 0.5 to 1.5, we will assign it to label 1. This brings us to the basic idea behind classification algorithms which can give qualitative response.

B. *Logistic Regression*

*1) **The Logit Function**:* A Logit function, commonly known as the log-odds function, depicts probability values ranging from 0 to 1, as well as negative infinity to infinite. The function is the inverse of the sigmoid function, except instead of the X-axis, it restricts values between 0 and 1. The Logit function is most often utilised in analysing probabilities since it occurs in the 0 to 1 range.

$$\text{logit}(p) = \sigma^{-1}(p) = \ln\left(\frac{p}{1-p}\right) \quad \text{for} \quad p \in (0,1) \tag{2}$$

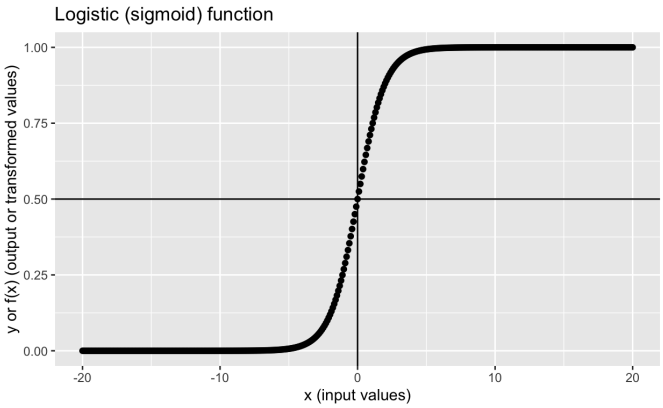

Fig. 1. Logit function in the domain of 0 to 1

*2) **Using Logistic Regression**:* Logistic regression is a technique used when the dependent variable is categorical. Logistic regression employs binomial probability theory in which there are only two values to predict: that probability (p) is 1 rather than 0, i.e. the event belongs to one group rather than the other.

Consider a model with two predictors, 1x1 and x2, and one binary (Bernoulli) response variable, Y, which we call p=P(Y=1). We assume that the predictor variables and the log-odds (logit) of the occurrence that Y=1 have a linear relationship. The following mathematical form may be constructed for this linear connection

$$\ell = \log_b \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{3}$$

where l is the log-odds, b is the base of the logarithm, and β_i are parameters of the model

We can calculate odds and probability as

$$\frac{p}{1-p} = b^{\beta_0+\beta_1 x_1} \tag{4}$$

$$p = \frac{b^{\beta_0+\beta_1 x_1}}{b^{\beta_0+\beta_1 x_1}+1} = \frac{1}{1+b^{-(\beta_0+\beta_1 x_1)}} \tag{5}$$

*3) **Maximum Likelihood Estimation**:* In logistic regression, we use Maximum Likelihood Estimation(MLE). The main aim of MLE is to find the value of our parameters for which the likelihood function is maximized. The likelihood function is nothing but a joint pdf of our sample observations and joint distribution is the multiplication of the conditional probability for observing each example given the distribution parameters.

Let a generalized linear model function parameterized by θ

$$h_\theta(X) = \frac{1}{1+e^{-\theta^T X}} = \Pr(Y=1 \mid X;\theta) \tag{6}$$

The likelihood function for logistic regression is

$$L(\theta \mid y;x) = \Pr(Y \mid X;\theta) \tag{7}$$

$$= \prod_i \Pr(y_i \mid x_i;\theta) \tag{8}$$

$$= \prod_i h_\theta(x_i)^{y_i}(1-h_\theta(x_i))^{(1-y_i)} \tag{9}$$

*4) **Fitting the model**:* We use the log-likelihood as the cost function in case of logistic regression. It is defined as -

$$n^{-1}\log L(\theta \mid y;x) = n^{-1}\sum_{i=1}^{n}\log \Pr(y_i \mid x_i;\theta) \tag{10}$$

To find the coefficients of the our model, we use optimization methods like gradient descent or Iteratively reweighted least squares (IRLS). In IRLS method, we maximise log-likelihood of a Bernoulli distributed process. We have the following data -

Parameters -

$$\mathbf{w}^T = [\beta_0, \beta_1, \beta_2, \ldots] \tag{11}$$

Explanatory Variables -

$$\mathbf{x}(i) = [1, x_1(i), x_2(i), \ldots]^T \tag{12}$$

Expected value of the Bernoulli distribution -

$$\mu(i) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}(i)}} \tag{13}$$

We can iteratively use the following algorithm -

$$\mathbf{w}_{k+1} = \left(\mathbf{X}^T\mathbf{S}_k\mathbf{X}\right)^{-1}\mathbf{X}^T\left(\mathbf{S}_k\mathbf{X}\mathbf{w}_k + \mathbf{y} - \boldsymbol{\mu}_k\right) \tag{14}$$

where $\mathbf{S} = \text{diag}(\mu(i)(1-\mu(i)))$ and $\boldsymbol{\mu} = [\mu(1), \mu(2), \ldots]$ and

$$\mathbf{X} = \begin{bmatrix} 1 & x_1(1) & x_2(1) & \dots \\ 1 & x_1(2) & x_2(2) & \dots \\ \vdots & \vdots & \vdots & \end{bmatrix}$$
$$\mathbf{y}(i) = [y(1), y(2), \dots]^T$$

Through this method, we can find the coefficients for our model and fit our data-set.

C. *Evaluating the model*

In model evaluation, we measure the extent to which the model fits the data.

*1) **Confusion Matrix**:* A confusion matrix is a table that shows the performance of a machine learning model. It displays how many of the model's predictions were right and how many were incorrect and which classes did the model succeed in and which did it fail in. It offers us an understanding of how the model works. If a model fails for a certain class, we may investigate it, figure out why, and try to improve the model.

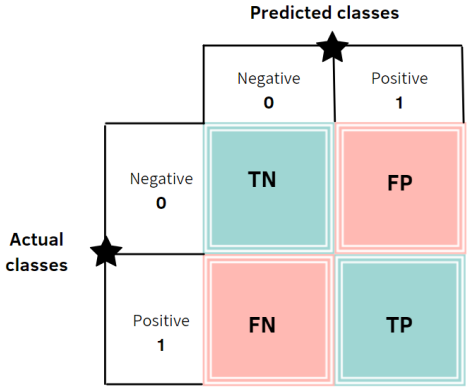

Fig. 2. Sample Confusion Matrix

There are various metrics based on confusion matrix. Some of them are listed below

*2) **Precision and Recall**:* Precision is the ratio between the True Positives and all the Positives. For our model, it will be the measure of persons who we correctly identified as survived out of all the people who survived.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \tag{15}$$

Recall is the measure of our model correctly identifying True Positives. For all the patients who actually survived, recall tells us how many we correctly predicted.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{16}$$

*3) **F1 Score**:* F1 score is the measure which combines both precision and recall. It is calculated as the harmonic mean of precision and recall -

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}$$

D. *Multivariate Logistic Regression*

In practice, we often have more than one predictor. We extend the simple logistic regression model so that it can directly accommodate multiple predictors. The ordinary expression can be modified as -

$$\log \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n \tag{18}$$

$$p = \frac{1}{1 + b^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n)}} \tag{19}$$

To estimate the coefficients of multivariate logistic regression, we use the same Maximum Likelihood Estimation methods.

III. The Problem

The sinking of the Titanic is one of the most infamous shipwrecks in history. 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. While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others. In this challenge, we have to build a predictive model that answers the question: "what sorts of people were more likely to survive?" using passenger data (ie name, age, gender, socio-economic class, etc).

A. *The Variables*

Survival - Survival in the crash, 0 = No, 1 = Yes
Pclass - Ticket class of the Passenger, 1 = 1st, 2 = 2nd, 3 = 3rd class.
Sex - Male or Female
Age - Age in years
Sibsp - Number of siblings / spouses aboard the Titanic.
Parch - Number of parents / children aboard the Titanic.
Ticket - Ticket number.
Fare - Passenger fare.
Cabin - Cabin number.
Embarked - Port of Embarkation out of Cherbourg, Queenstown, and Southampton.

B. *Cleaning the Data*

We check for null values in all columns.(Fig. 3)

We find that the a significant percentage of cabin data missing. So we drop that column. The missing data on age can be imputed.

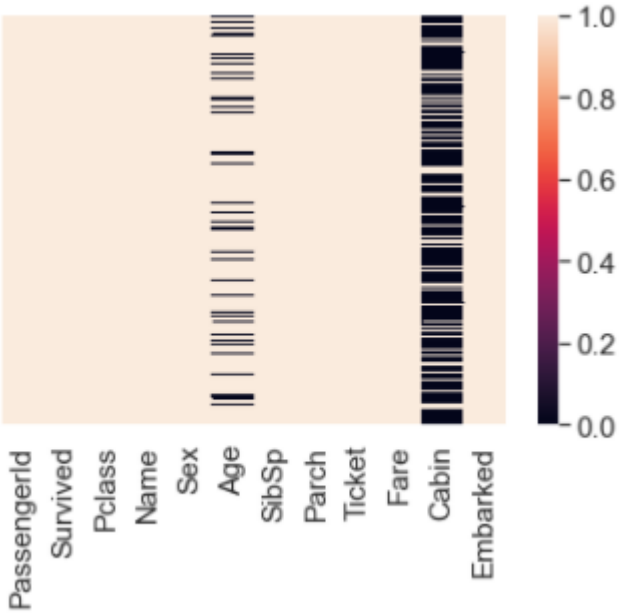

Fig. 3. Null values heatmap

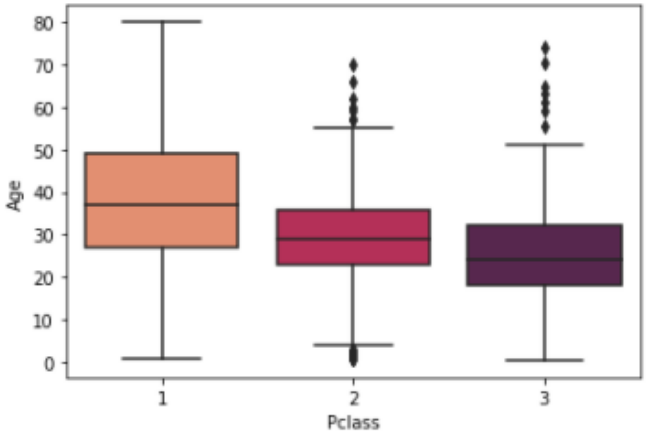

Fig. 4. Age grouped by Pclass

C. *Cleaning and Visualizing the numeric variables*

*1) **Age**:* The age distribution has clear trends with Pclass, as visible from the boxplot.

The median values class-wise are-
Class 1 - 37
Class 2 - 29
Class 3 - 24

We impute these values for the missing data.

To visualize the trend, we make a lineplot between Age and the proportion of passengers who survived of that particular age. (Fig. 5 and 6.)

*2) **Fare**:* The fare is well-distributed, with the mean fare being 32.204.

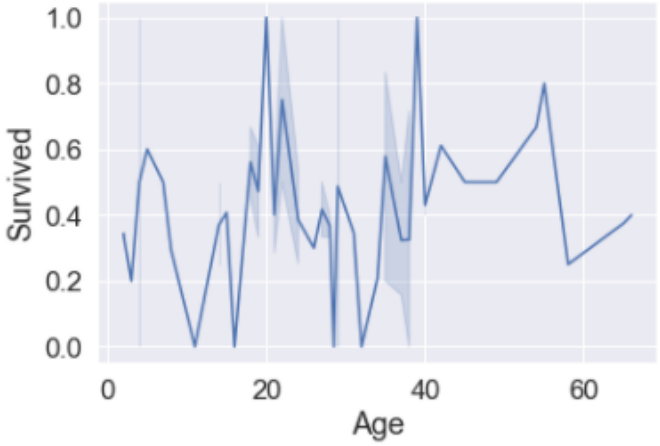

Fig. 5. Survival vs Age

As expected, the fare varies with passenger class, with means being -
Class 1 - 84.15
Class 2 - 20.66
Class 3 - 13.67

Dealing with Outliers - Some fare values are more than 500, which are obvious outliers. So we drop these columns.

*3) **PClass**:* We will check whether there is any relation between the proportion of people survived and their Passenger class by drawing a countplot grouped by class.

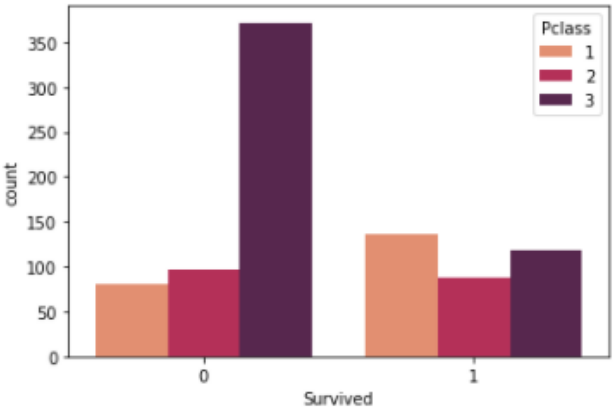

Fig. 6. Survival grouped by Passenger Class

The 1st class survived more, as evident from mean of survived values grouped by Pclass -

The median values class-wise are-
Class 1 - 0.63
Class 2 - 0.47
Class 3 - 0.24

*4) **Siblings and Parents**:* We can combine the siblings and parents columns to figure out if a passenger is travelling alone or with relatives. Then we plot the data to compare

survival based on co-passengers.

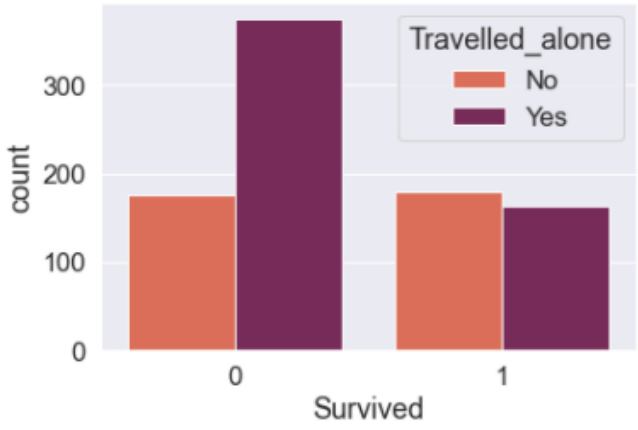

Fig. 7. Survival grouped by number of relatives onboard

It seems that passengers who travelled alone had less survival chance.(Fig. 7)

D. *Feature Engineering in Categorical Variables*

In this section, we will visualize and create dummies for the categorical variables.

1) ***Sex*** *:* We check whether survival depends upon gender or not, by making a countplot.

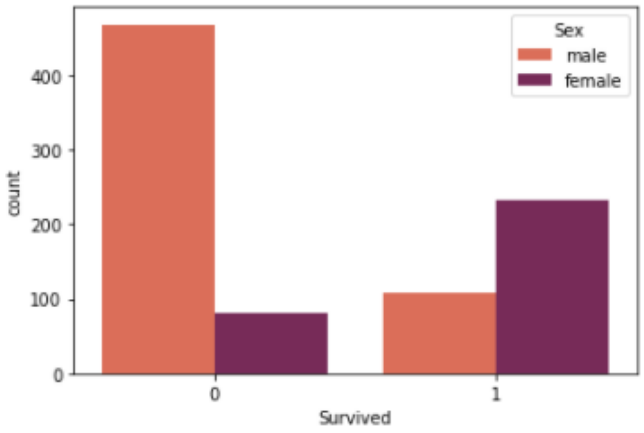

Fig. 8. Survival grouped by Sex

The mean of survived variable is -
Female - 0.742
Male - 0. 188

Clearly, females had a better chance of survival. We convert the categorical features into numeric by creating male and female columns.
(Fig. 8)

2) ***Embarked****:* There are 3 embarkation points- Cherbourg, Queenstown, and Southampton. To check whether the survival depends on embarkation point, we create the following countplot.
(Fig. 9)

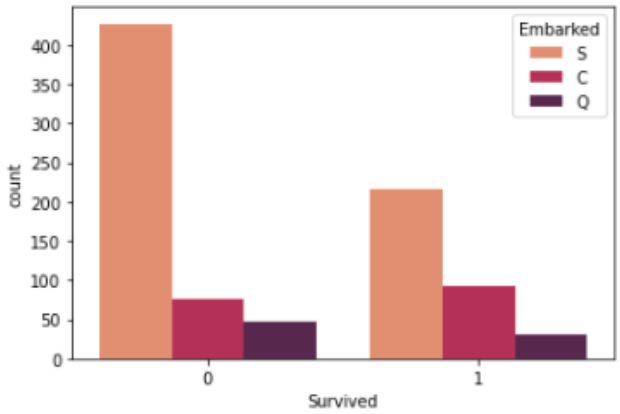

Fig. 9. Embarked Countplot

The embarkation pointwise mean of survived variable is -
Southampton - 0.553
Cherbourg - 0.389
Queenston - 0.336

Then we create dummy columns to account for these trends in our model.

3) ***Name and Ticket****:* These variables can be dropped.

E. *Checking Correlation*

In our final set of variables, we will check if there is any instance of high correlation among them, by making a heatmap.

No high correlation is witnessed among our variables.(Fig. 10)

IV. The Model

A. *Train Test Split*

We have the following X and y for our final model, which we divide our data into training and testing data..
X= PClass, Age, SibSp, Parch, Fare, Male, Female, Embarked, Relatives
Y = Survived

B. *Training the Model*

Using scikit-learn's LogisticRegression() function on our X and y variables, we develop a multivariate logistic regression model. The coefficients -

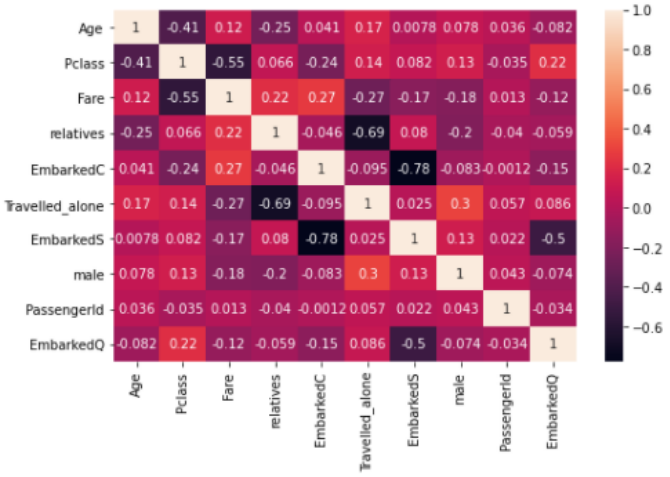

Fig. 10. Correlation Heatmap

	Coefficient
Pclass	-0.896801
Age	-0.555191
Fare	0.177938
relatives	-0.625110
Travelled_alone	-0.276369
male	-0.622823
female	0.622823
EmbarkedS	-0.244168
EmbarkedC	-0.192774
EmbarkedQ	0.010602

Fig. 11. Coefficients

C. *Predictions and Metrics*

*1) **Training Data**:* The confusion matrix for our training set in shown below (Fig.12)

Based on the matrix, we can calculate the following -

*2) **Testing Data**:* The confusion matrix for our testing set in shown below (Fig.14)

Based on the matrix, we can calculate the following -

V. Improvements in the Model

A. *Random Search CV*

Machine learning models have hyperparameters.

Hyperparameters are points of choice or configuration that allow a machine learning model to be customized for a specific

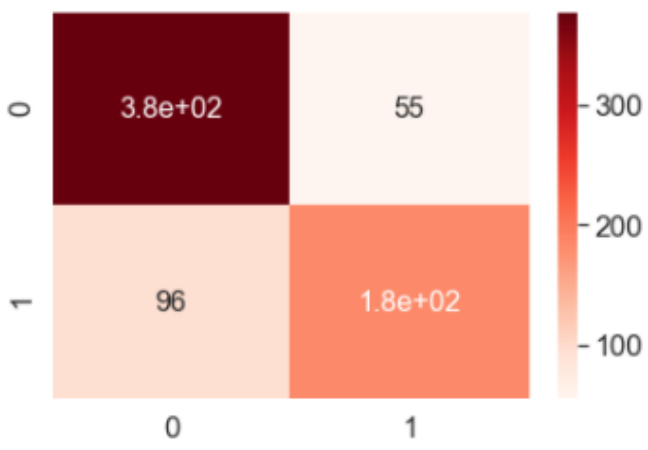

Fig. 12. Confusion Matrix for training data

	precision	recall	f1-score	support
0	0.81	0.87	0.84	432
1	0.77	0.69	0.73	280
accuracy			0.80	712
macro avg	0.79	0.78	0.78	712
weighted avg	0.79	0.80	0.79	712

Fig. 13. Scores for training data

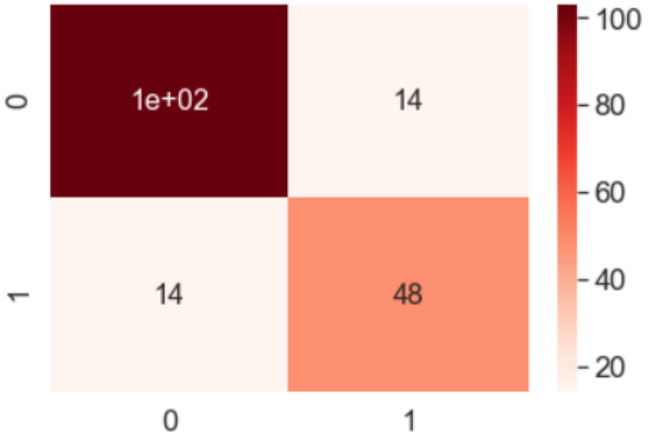

Fig. 14. Confusion Matrix for testing data

	precision	recall	f1-score	support
0	0.88	0.88	0.88	117
1	0.77	0.77	0.77	62
accuracy			0.84	179
macro avg	0.83	0.83	0.83	179
weighted avg	0.84	0.84	0.84	179

Fig. 15. Scores for testing data

task or dataset.In a model, we have to objectively search different values for model hyperparameters and choose a subset that results in a model that achieves the best performance on a given dataset. This is called hyperparameter optimization.

Random search is a technique where random combinations of the hyperparameters are used to find the best solution for the built model. We use the the scikit-learn's RandomizedSearchCV() function for our logistic regression.

B. XGBoost

XGBoost stands for eXtreme Gradient Boosting.
It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework.

XGBoost is an ensemble tree method that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture.

XGBoost uses the following optimization techniques -

- **Parallelization** - XGBoost approaches the process of sequential tree building using parallelized implementation. This is possible due to the interchangeable nature of loops used for building base learners; the outer loop that enumerates the leaf nodes of a tree, and the second inner loop that calculates the features.

- **Tree Pruning** - The stopping criterion for tree splitting within GBM framework is greedy in nature and depends on the negative loss criterion at the point of split. XGBoost uses 'max depth' parameter as specified instead of criterion first, and starts pruning trees backward.

- **Hardware Optimization** - This algorithm has been designed to make efficient use of hardware resources. This is accomplished by cache awareness by allocating internal buffers in each thread to store gradient statistics.

C. Scaling the Data

Scaling refers to the process of putting values in the same range or scale such that no one variable dominates the others. Many Machine Learning algorithms use distance between two data points and if the features have varying magnitudes.

We have used scikit-learn's StandardScaler method to scale our data. StandardScaler removes the mean and scales each feature to unit variance. This operation is performed feature-wise in an independent way.

D. SMOTE for balancing the data

SMOTE stands for Synthetic Minority Oversampling Technique .It is used to address imbalanced datasets to oversample the minority class. The simplest approach involves duplicating examples in the minority class, although these examples don't add any new information to the model. Instead, new examples can be synthesized from the existing examples. This is a type of data augmentation for the minority class and is referred to as the Synthetic Minority Oversampling Technique, or SMOTE for short. SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space, and drawing a new sample at a point along that line. Specifically, a random example from the minority class is first chosen. Then k of the nearest neighbors for that example is found (typically k=5). A randomly selected neighbor is chosen and a synthetic example is created at a randomly selected point between the two examples in feature space.

E. Random Search for Hyper Parameter Tuning

Using scikit-learn's RandomizedSearchCV function on the following params -
'penalty': ['l1','l2'], 'C': [0.001,0.01,0.1,1,10,100,1000],

The best model comes out with C=1000.

F. Applying XGBoost

XGBoost is one of the most powerful algorithms available and it works the best for our data. It gives a f1 score of 0.9755 on the training set and 0.8258 on testing set. This implies that XGBoost is overfitting the data.

VI. Conclusions

From our observations, we can conclude the following -

- **Passenger class** - A higher proportion of people who survived were in class 1. The class 3 passengers survived the least.
- **Sex** - More number of female passengers survived in the crash than the male counterparts.
- **Embarkation Point** - Out of the 3 embarkation points, Cherbourg, Queenstown, and Southampton, we find that the passengers from Southampton survived in higher proportion, followed by Cherbourg and then Queenstown.
- **Travelling alone** - According to the observations, passengers who travelled alone had lower chances of survival.
- **Age** - From the data, we observed that elder passengers bought class 1 tickets which increased their survival.
- **Fare** - Similar to age, we can report that higher the fare, higher is the passenger class and higher is survival chance.

After improvements

- Using RandomizedSearchCV, the most optimal value of hyper parameter C is 1000.

- XGBoost Classifier overfits the data with a training score of 0.97 and a significantly lower testing score of 0.83

Thus, survival was not entirely based on luck, there were various factors that contributed to it, as shown by both the qualitative and the quantitative analysis.

REFERENCES

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, An introduction to statistical learning: With applications in r. New York, NY: Springer, 2021.

[2] "Logistic regression," Wikipedia, 21-Sep-2021. [Online]. Available: https://en.wikipedia.org/wiki/Logisticregression. [Accessed: 27-Sep-2021].

[3] J. Brownlee, "What is a confusion matrix in machine learning," Machine Learning Mastery, 14-Aug-2020. [Online]. Available: https://machinelearningmastery.com/confusion-matrix-machine-learning/. [Accessed: 27-Sep-2021].

Assignment 3: Naive Bayes Classifier

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras
me18b114@smail.iitm.ac.in

***Abstract*—This paper gives a brief introduction of naive bayes classifier and the basic mathematical theory behind it. We also look at some metrics to evaluate the performance of our model. Further, we apply the classifier on a real world data-set containing the socio-economic details of the people of USA, collected by US Census Bureau in 1996, to predict whether a person has greater or lesser than 50k income. Through exploratory data analysis and predictive naive bayes classifier model, we investigate trends in data to predict the persons who have higher income. Then we evaluate our model using confusion matrix and F1 score. Our findings reveal some interesting trends.**

On the second attempt, the following additions have been made to the model -
1. Random Search and XGBoost theory.
2. Scaling the data.
3. Balancing the target class with SMOTE.
4. Applying XGBoost on the data.
5. Hyper parameter tuning with Random Search.

I. Introduction

There are multiple types of algorithm methods used in machine learning. One such popular and commonly used machine learning method is Naive Bayes Classifier. It is a supervised learning algorithm used to train a model to predict the behaviour of our data based on given parameters.

Naive Bayes Classifier is a classification technique based on Bayes' Theorem which classifies the data into various classes based on prior probabilities and the likelihood, with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Mathematically,

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x} \mid C_k)}{p(\mathbf{x})} \tag{1}$$

where,
C is the class.
x represents the features of the model.
K is the number of classes.

In this paper, we apply Naive Bayes Classifier to data which was extracted from the 1994 Census bureau database by Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics). The key task is to determine whether a person makes over 50K a year, adult.csv contains the dataset required to solve the task..

The paper will first give a brief explanation of logistic regression. Then, we will try to solve the Income prediction and model our data-set by
1. Cleaning the data.
2. Exploratory and visual analysis.
3. One-Hot Encoding.
4. Checking for Correlation.
5. Scaling the Data.
6. Training the model.
7. Evaluating the model.
8. Interpreting the results.

II. Naive Bayes Classifier Theory

A. *Classification*

Classification is a process of categorizing a given set of data into classes. Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category, label or class.

Many real world problems require classification rather than regression. In the classification setting we have a set of training observations (x1, y1), . . . , (xn, yn) that we can use to build a classifier.

Regression is not useful in problems like if we are trying to predict the medical condition of a patient in the hospital and we have 3 possible diagnoses- stroke, drug overdose, and epileptic seizure based on the symptoms of the patient. We can label these diseases at 1,2 and 3. In this case, even if we create a metric to accurately convert the symptoms to numeric data, regression won't be able to classify the data as it cannot give the output as 1,2 or 3. Alternatively, we can create a threshold that if predicted value is between 0.5 to 1.5, we will assign it to label 1. This brings us to the basic idea behind classification algorithms which can give qualitative response.

B. *Naive Bayes Classifier*

*1) **The Bayes' Theorem**:* Bayes' theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event. It is named after Thomas Bayes, an English statistician and philosopher. It is

mathematically stated as -

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \tag{2}$$

where A and B are events and
• P(A/B) is a conditional probability: the probability of event A occurring given that B is true. It is also called the posterior probability of A given B.
• P(B/A) is the probability of event B occurring given that A is true.
• P(A) and P(B) are probabilities of observing A and B respectively without any given conditions.

*2) **Using the Naive Bayes Classifier**:* Naive Bayes is a classification algorithm based on Bayes theorem. Given k possible outcomes or classes, and vector x representing n features, the model calculates the probability of the entry being is class k. Mathematically, it calculates -

$$p(C_k \mid x_1, \ldots, x_n) \tag{3}$$

It is called Naive classifer because it makes a naive assumption to the Bayes' theorem, which is, independence among the features, i.e $x_1, ..., x_n$ are all independent.

In other words, the probability is calculated as -

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x} \mid C_k)}{p(\mathbf{x})} \tag{4}$$

which translates to -

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}} \tag{5}$$

The denominator is constant and the numerator is equivalent to joint probability of C and x. After applying the naive independence condition, the joint probability boils down to -

$$p(C_k \mid x_1, .., x_n) \propto p(C_k, x_1, \ldots, x_n) \tag{6}$$

$$\propto p(C_k)\ p(x_1 \mid C_k)\ p(x_2 \mid C_k)\ p(x_3 \mid C_k) \tag{7}$$

$$\propto p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k) \tag{8}$$

The constant of proportionality, the scaling factor is P(X)

$$Z = p(\mathbf{x}) = \sum_k p(C_k)\ p(\mathbf{x} \mid C_k) \tag{9}$$

Therefore, the final distribution is -

$$p(C_k \mid x_1, \ldots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k) \tag{10}$$

*3) **Working of the classifier**:* Once we have the distribution for the joint probability for C and x, the classifier used a decision rule called Maximum a Posteriori (MAP). It classifies the entry to the $k_t h$ class which has the maximum probability.

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}}\ p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k) \tag{11}$$

C. Evaluating the model

In model evaluation, we measure the extent to which the model fits the data.

*1) **Confusion Matrix**:* A confusion matrix is a table that shows the performance of a machine learning model. It displays how many of the model's predictions were right and how many were incorrect and which classes did the model succeed in and which did it fail in. It offers us an understanding of how the model works. If a model fails for a certain class, we may investigate it, figure out why, and try to improve the model.

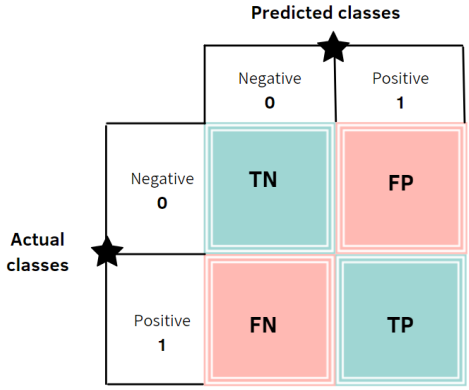

Fig. 1. Sample Confusion Matrix

There are various metrics based on confusion matrix. Some of them are listed below

*2) **Precision and Recall**:* Precision is the ratio between the True Positives and all the Positives. For our model, it will be the measure of persons who we correctly identified having income greater than 50k

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \tag{12}$$

Recall is the measure of our model correctly identifying True Positives. For all the people who have higher than 50k income, recall tells us how many we correctly predicted.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{13}$$

*3) **F1 Score**:* F1 score is the measure which combines both precision and recall. It is calculated as the harmonic mean of precision and recall -

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{14}$$

III. The problem

Given the data extracted by 1994 Census database by (Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics). We have to determine whether a person makes over dollar 50K a year or not based on the below given variables-

A. *The Variables*

Age - Age in years.
Work class - The working class out of Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
fnlwgt - This is the number of people the census believes the entry represents.
Education - Level of education.
Education-num - No. of years of education
Marital-status - Marital status of the person.
Occupation - Occupation of the person.
Race - The race of the person.
Sex - Male or Female.
capital-gain - Profit resulting from an investment or sale of a property.
capital-loss - Loss resulting from an investment or buying of a property.
Native Country - The native country of the person.

B. *Cleaning the Data*

We check for null values in all columns. No data is missing in the dataset.

C. *The target variable - Income*

We have 2 type of entries in the income column - greater and less than 50K, which labels the number of people having annual income in the above ranges. We have to build a Naive Bayesian Classifier to classify the test set. The target variable doesn't have a high degree of imbalance.

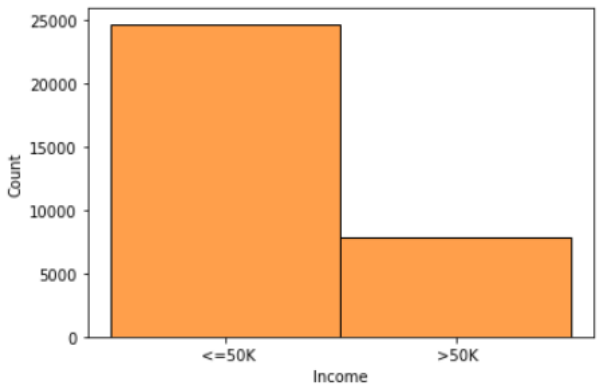

Fig. 2. Income countplot

D. *Cleaning and Visualizing the Numeric Variables*

*1) **Age**:* The age is well distributed with a mean of 38.58 years.
We visualize it using a boxplot.

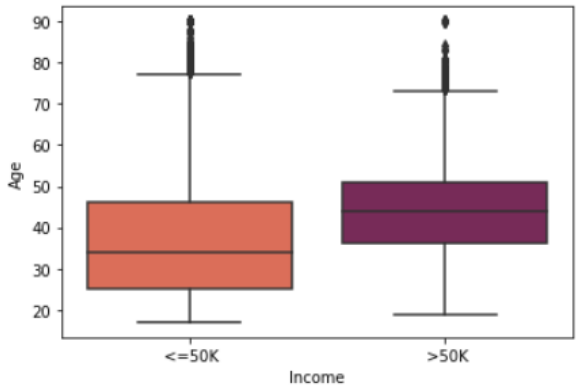

Fig. 3. Age vs Income

The mean age is higher for the people having greater than 50K income, which is quite intuitive as professionals gain experience and reach higher posts with higher age.

*2) **Education num**:* Most of the population have studied for 10 or more years. People who have studied more and gotten higher education tend to have higher income.

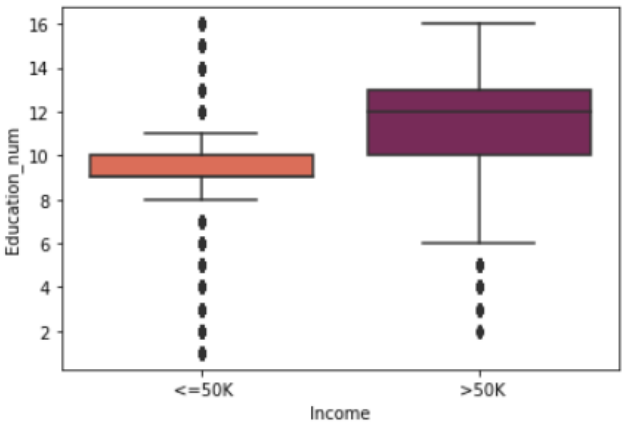

Fig. 4. Age vs Income

*3) **fnlwgt**:* This is a variable included by census committee to indicate the number of people the census believes the entry represents.

This variable does not seem to have any effect on our classifier.

*4) **Hours per week**:* Through the boxplot, we can see that increased number of hours directly leads to higher income.

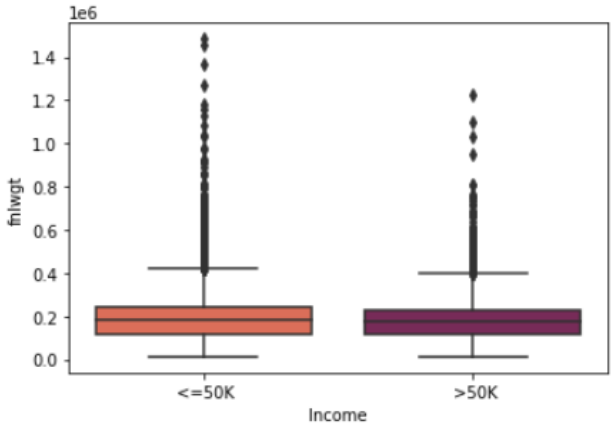

Fig. 5. fnlwgt vs Income

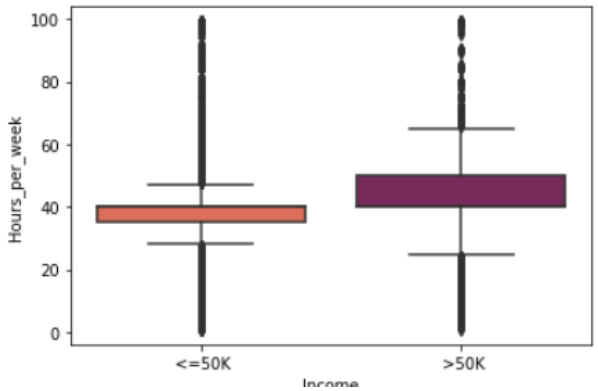

Fig. 6. Hours per week v Income

*5) **Capital gain and Capital loss**:* Some people get profits/losses from transactions of property or investments. This supplements their annual income. In our original dataset, the number of people having greater than 50K income is only 24.08 %.

But if we conditionally select the people who have made gains more than 0, the percent changes to 61.85 %.

E. Cleaning and Visualizing the Categorical Variables

In this section, we will visualize the trends for the categorical variables.

*1) **Work Class** :* This variable represents the type of job the person has.
We find that some entries are '?'. We replace them with the mode value, which is 'Private'.

To visualize how Work Class effects the income, we plot a bar plot between the work class and the fraction of people having income above 50K.

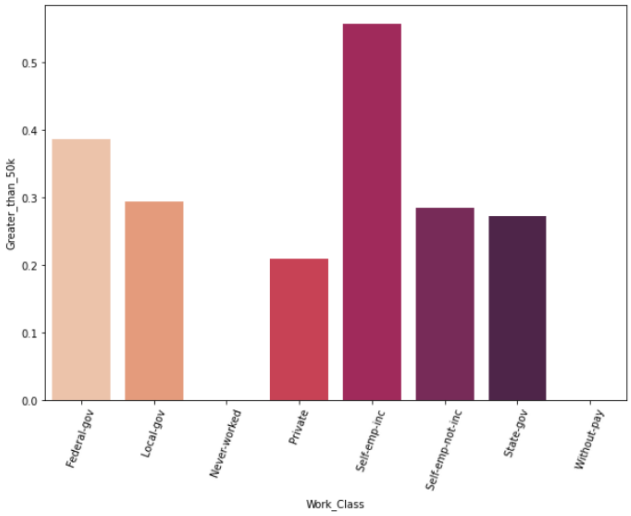

Fig. 7. Fraction of people having income greater than 50K vs Work Class

We see that self employed and federal government employees have higher fractions.

*2) **Education**:* This represents what kind of education the persons have received from Bachelors, HS-grad, Masters etc. To visualize how Education effects the income, we plot a bar plot between the Education and the fraction of people having income above 50K.

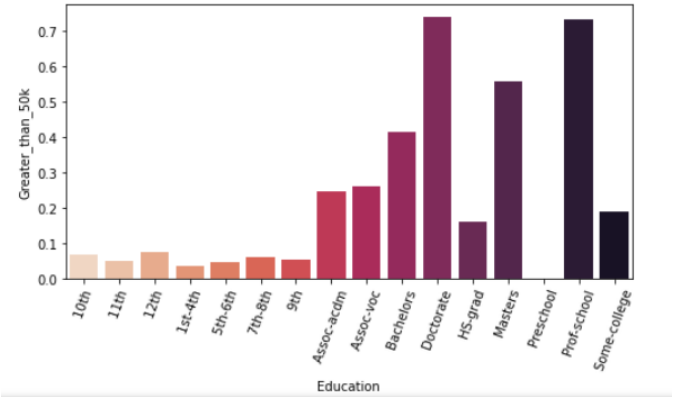

Fig. 8. Fraction of people having income greater than 50K vs Education

We can see that Doctorates and Prof-School graduates are among the top earners while the high school studied people have the lowest income.

*3) **Marital Status**:* This represents the marital status (divorced, married etc.)
To visualize how marital status effects the income, we plot a bar plot between the Marital Status and the fraction of people having income above 50K.

It is visible that married people constitute a larger fraction of people having higher income.

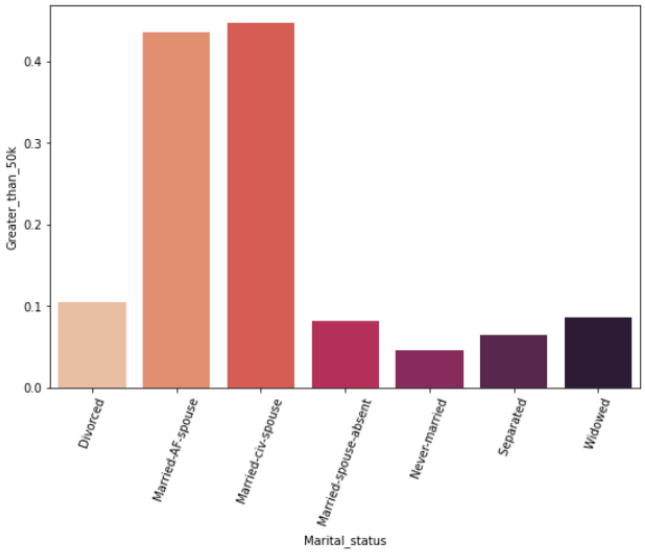

Fig. 9. Fraction of people having income greater than 50K vs Marital status

4) ***Occupation****:* This represents the type of job like sales, IT, clerical etc.
Some columns are missing some data and contain '?'. We replace it with the mode value.
To visualize how the job type effects the income, we plot a bar plot between the Occupation and the fraction of people having income above 50K.

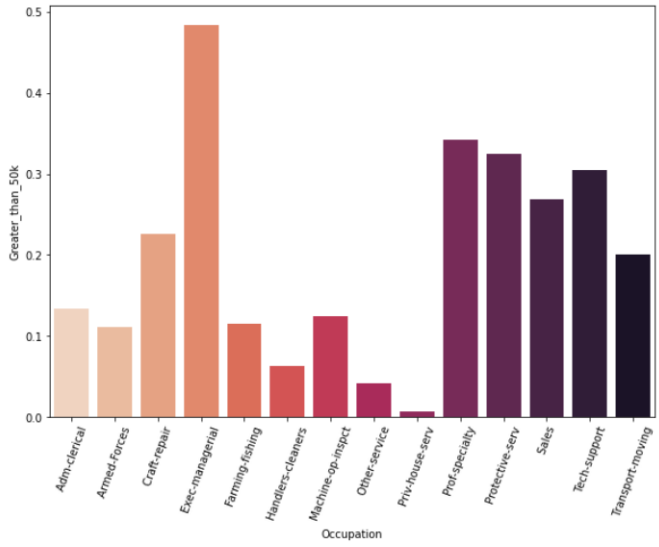

Fig. 10. Fraction of people having income greater than 50K vs Occupation

It is visible that Executive managers tend to have higher income.

5) ***Race****:* This variable represents the Race of the person.
To visualize how race effects the income, we plot a bar plot between the Race and the fraction of people having income above 50K.

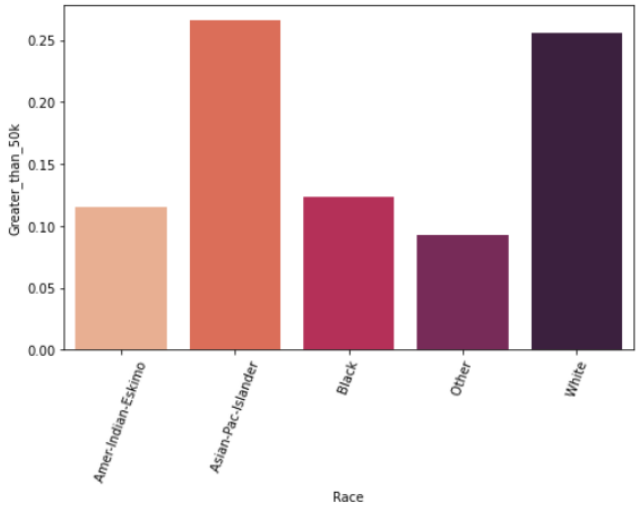

Fig. 11. Fraction of people having income greater than 50K vs Race

It is visible that Asian and White people have higher number of people above 50K income.

6) ***Sex****:* This represents the sex of the person
To visualize how marital status effects the income, we plot a bar plot between the Sex and the fraction of people having income above 50K.

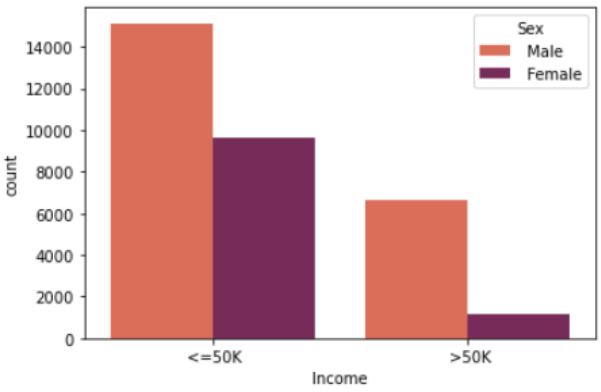

Fig. 12. Fraction of people having income greater than 50K vs Sex

Gender inequality is clearly evident from the graph. Out of all the entries, female employees account for only 10.9 % above 50K while men acocunt for 30.57 %.

F. ***One Hot Encoding***

To account for categorical variables in our model, we use One-Hot Encoder. With this, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns. Each integer value is represented as a binary vector.

G. *Checking Correlation*

Through the below heatmap, we can see no high values of correlation among our numeric variables-

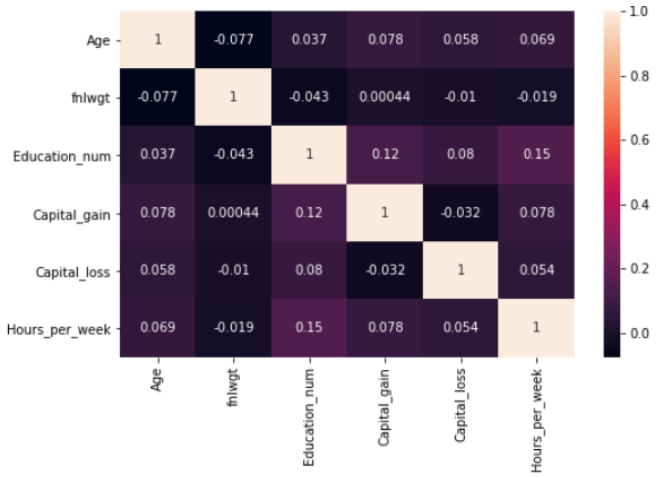

Fig. 13. Correlation Heatmap

IV. The Model

A. *Train Test Split*

We have the following X and y for our final model, which we divide our data into training and testing data..
X= 'Age', 'Work Class', 'fnlwgt', 'Education', 'Education num', 'Marital status', 'Occupation', 'Relationship', 'Race', 'Sex', 'Capital gain', 'Capital loss', 'Hours per week', 'Native country', 'Income' .
Y = Income

B. *Training the Model*

Using scikit-learn's GaussianNB() function on our X and y variables, we develop a Gaussian Naive Bayes classifier model.

C. *Predictions and Metrics*

The confusion matrix for our testing set in shown below
Based on the matrix, we can calculate the following -

The f1 score is 0.8028.

V. Improvements in the Model

A. *Random Search CV*

Machine learning models have hyperparameters.
Hyperparameters are points of choice or configuration that allow a machine learning model to be customized for a specific task or dataset.In a model, we have to objectively search different values for model hyperparameters and choose a subset that results in a model that achieves the best performance on a given dataset. This is called hyperparameter optimization.

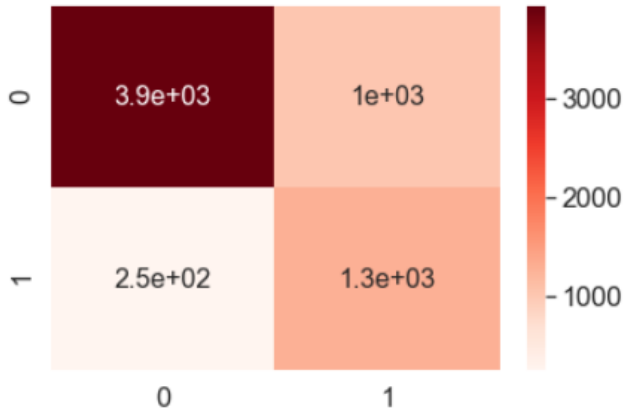

Fig. 14. Confusion Matrix for testing data

	precision	recall	f1-score	support
<=50K	0.94	0.79	0.86	4959
>50K	0.56	0.84	0.67	1553
accuracy			0.80	6512
macro avg	0.75	0.81	0.76	6512
weighted avg	0.85	0.80	0.81	6512

Fig. 15. Scores for testing data

Random search is a technique where random combinations of the hyperparameters are used to find the best solution for the built model. We use the the scikit-learn's Randomized-SearchCV() function for our logistic regression.

B. *XGBoost*

XGBoost stands for eXtreme Gradient Boosting.
It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework.

XGBoost is an ensemble tree method that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture.

XGBoost uses the following optimization techniques -

- **Parallelization** - XGBoost approaches the process of sequential tree building using parallelized implementation. This is possible due to the interchangeable nature of loops used for building base learners; the outer loop that enumerates the leaf nodes of a tree, and the second inner loop that calculates the features.

- **Tree Pruning** - The stopping criterion for tree splitting within GBM framework is greedy in nature and depends

on the negative loss criterion at the point of split. XGBoost uses 'max depth' parameter as specified instead of criterion first, and starts pruning trees backward.

- **Hardware Optimization** - This algorithm has been designed to make efficient use of hardware resources. This is accomplished by cache awareness by allocating internal buffers in each thread to store gradient statistics.

C. Scaling the Data

Scaling refers to the process of putting values in the same range or scale such that no one variable dominates the others. Many Machine Learning algorithms use distance between two data points and if the features have varying magnitudes.

We have used scikit-learn's RobustScaler method to scale our data. RobustScaler transforms the feature vector by subtracting the median and then dividing by the inter-quartile range.

D. SMOTE for balancing the data

SMOTE stands for Synthetic Minority Oversampling Technique .It is used to address imbalanced datasets to oversample the minority class. The simplest approach involves duplicating examples in the minority class, although these examples don't add any new information to the model. Instead, new examples can be synthesized from the existing examples. This is a type of data augmentation for the minority class and is referred to as the Synthetic Minority Oversampling Technique, or SMOTE for short. SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space, and drawing a new sample at a point along that line. Specifically, a random example from the minority class is first chosen. Then k of the nearest neighbors for that example is found (typically k=5). A randomly selected neighbor is chosen and a synthetic example is created at a randomly selected point between the two examples in feature space.

E. Random Search for Hyper Parameter Tuning

Using scikit-learn's RandomizedSearchCV function on the following params -
'var smoothing': np.logspace(0,-10, num=100)

The best model comes out with var smoothing =1.2618568830660183e-10

F. Applying XGBoost

XGBoost is one of the most powerful algorithms available and it works the best for our data. It gives a f1 score of 0.9307 on the training set and 0.7874 on testing set. This implies that XGBoost is overfitting the data.

VI. Conclusions

From our observations, we can conclude the following about the significant features-

- **Age** -With increasing age, people tend to have higher income.
- **Sex** - There is a clear gender disparity in income, with males having higher income.
- **Hours per week** - More hours per week on the job leads to more income.
- **Education** - People with higher education and with doctorates and professional degrees have higher pay.
- **Occupation** - From the data, we observed that self employed people have higher income and among the professional workers, executive managers have the highest fraction of people above 50K income.
- **Race and Native Country** - The American people, especially the whites, have higher income, followed by people from Asia-Pacific countries.
- **Capital gain or loss** -Making capital gains via profits from sale of properties of investments directly supplements the income, making it higher.

After improvements

- Using RandomizedSearchCV, the most optimal value of hyper parameter 'var smoothing' is 1.2618568830660183e-10.
- XGBoost Classifier overfits the data with a training score of 0.93 and a significantly lower testing score.

Thus, as shown by both the qualitative and the quantitative analysis, the income of the people of the United Stated depends upon many social, cultural and educational factors.

References

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, An introduction to statistical learning: With applications in r. New York, NY: Springer, 2021.

[2] "Naive Bayes classifier," Wikipedia, 18-Aug-2021. [Online]. Available: https://en.wikipedia.org/wiki/NaiveBayesclassifier. [Accessed: 12-Oct-2021].

[3] R. Gandhi, "Naive Bayes classifier," Medium, 17-May-2018. [Online]. Available: https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c. [Accessed: 12-Oct-2021].

[4] J. Brownlee, "What is a confusion matrix in machine learning," Machine Learning Mastery, 14-Aug-2020. [Online]. Available: https://machinelearningmastery.com/confusion-matrix-machine-learning/: :text=A

Assignment 4: Decision Trees Classifier

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras
me18b114@smail.iitm.ac.in

***Abstract*—This paper gives a brief introduction of Decision Trees Classifier and the basic mathematical theory behind it. We also look at some metrics to evaluate the performance of our model. Further, we apply the classifier on the Car Evaluation Database, derived from a simple hierarchical decision model. The prediction task is to classify based on its safety. Through exploratory data analysis and decision trees classifier, we investigate trends in data to predict which cars are safer. Then we evaluate our model using confusion matrix and F1 score. Our findings reveal some interesting trends.**

On the second attempt, the following additions have been made to the model -
1. Random Search and XGBoost theory.
2. Explored the hyper parameters of Decision Trees Classifier.
3. Hyper parameter tuning with Random Search.
4. Applying XGBoost on the data.

I. Introduction

There are multiple types of algorithm methods used in machine learning. One such popular and commonly used machine learning method is Decision Trees Classifier. It is a supervised learning algorithms. Decision trees are among the most popular machine learning algorithms given their intelligibility and simplicity.

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

In this paper, we apply Decision Trees Classifier to the Car Evaluation Database, which was derived from a simple hierarchical decision model, to classify a car based on its safety.

The paper will first give a brief explanation of decision trees classifier. Then, we will try to solve the Income prediction and model our data-set by
1. Cleaning the data.
2. Exploratory and visual analysis.
3. Feature Engineering for Categorical Variables.
4. Checking for Correlation.
5. Training the model.
6. Evaluating the model.
7. Interpreting the results.

II. Decision Tree Classifier Theory

A. *Classification*

Classification is a process of categorizing a given set of data into classes. Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category, label or class.

Many real world problems require classification rather than regression. In the classification setting we have a set of training observations (x1, y1), . . . , (xn, yn) that we can use to build a classifier.

Regression is not useful in problems like if we are trying to predict the medical condition of a patient in the hospital and we have 3 possible diagnoses- stroke, drug overdose, and epileptic seizure based on the symptoms of the patient. We can label these diseases at 1,2 and 3. In this case, even if we create a metric to accurately convert the symptoms to numeric data, regression won't be able to classify the data as it cannot give the output as 1,2 or 3. Alternatively, we can create a threshold that if predicted value is between 0.5 to 1.5, we will assign it to label 1. This brings us to the basic idea behind classification algorithms which can give qualitative response.

B. *Decision Tree Classifier*

A tree based classifier involves stratifying or segmenting the predictor space into a number of simple regions. In order to make a prediction for a given observation, we typically use the mean or the mode of the training observations in the region to which it belongs. Since the set of splitting rules used to segment the predictor space can be summarized in a tree, these types of approaches are known as decision tree methods. Tree-based methods are simple and useful for interpretation.

It is a tree-structured classifier, where internal nodes represent the features of a data set, branches represent the decision rules and each leaf node represents the outcome.

Terminology

- **Root Node** -Root node is from where the decision tree starts. It represents the entire data set, which further gets divided into two or more homogeneous sets.
- **Leaf Node** - Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting** - Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch Tree** - A tree formed by splitting the tree.
- **Pruning** -Pruning is the process of removing the unwanted branches from the tree.

C. *Working of Decision Tree Algorithm*

In a decision tree, for predicting the class of the given data set, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real data set) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree.

Attribute Selection - While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. There are two techniques for this -

*1) **Gini Index**:* Gini Impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. The Gini impurity can be computed by summing the probability of an item p_i with label i being chosen times the probability -

$$\sum_{k \neq i} p_k = 1 - p_i \tag{1}$$

, of a mistake in categorizing that item. It reaches its minimum (zero) when all cases in the node fall into a single target category.

To compute Gini impurity for a set of items with J classes, suppose $i \in \{1, 2, ..., J\}$,p_i be the fraction of items labeled with class *i* in the set.

$\mathrm{I}_G(p) = \sum_{i=1}^{J} \left(p_i \sum_{k \neq i} p_k \right)$

= $\sum_{i=1}^{J} p_i(1 - p_i)$

= $\sum_{i=1}^{J} (p_i - p_i^2)$

= $\sum_{i=1}^{J} p_i - \sum_{i=1}^{J} p_i^2$

= 1 - $\sum_{i=1}^{J} p_i^2$

*2) **Information Gain**:* Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute. It calculates how much information a feature provides us about a class. According to the value of information gain, we split the node and build the decision tree. A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

Information Gain = Entropy(S) - [(Weighted Avg)* Entropy(each feature)]

Entropy is defined as -

$$(T) = \mathrm{I}_E\left(p_1, p_2, \ldots, p_J\right) = -\sum_{i=1}^{J} p_i \log_2 p_i \tag{2}$$

where $p_1, p_2, \ldots$ are fractions that add up to 1 and represent the percentage of each class present in the child node that results from a split in the tree.

$$\overbrace{E_A(\mathrm{IG}(T,a))}^{\text{expected information gain}} = \overbrace{(T)}^{\text{entropy (parent)}} - \overbrace{(T \mid A)}^{\text{weighted sum of entropies (children)}}$$
$$= -\sum_{i=1}^{J} p_i \log_2 p_i - \sum_{a} p(a) \sum_{i=1}^{J} -\Pr(i \mid a) \log_2 \Pr(i \mid a)$$

That is, the expected information gain is the mutual information, meaning that on average, the reduction in the entropy of T is the mutual information.

D. *Advantages of Decision Trees Classifier*

- The logic of a decision tree is easy to follow and transparent. It mimics the human thought process by successively asking questions that adapt based on previous answers, until enough knowledge is gained to make a final prediction.
- While utilizing a decision tree algorithm, it is not essential to standardize or normalize the data that has been collected. It can handle both continuous and categorical variables.
- The decision tree follows a non-parametric method; meaning, it is distribution-free and does not depend on probability distribution assumptions. It can work on high-dimensional data with excellent accuracy.

E. *Evaluating the model*

In model evaluation, we measure the extent to which the model fits the data.

*1) **Confusion Matrix**:* A confusion matrix is a table that shows the performance of a machine learning model. It displays how many of the model's predictions were right and how many were incorrect and which classes did the model succeed in and which did it fail in. It offers us an understanding of how the model works. If a model fails for a certain class, we may investigate it, figure out why, and try to improve the model.

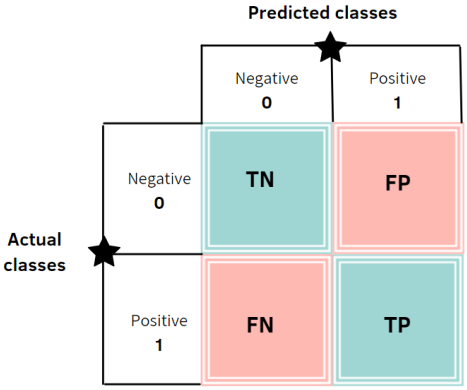

Fig. 1. Sample Confusion Matrix

There are various metrics based on confusion matrix. Some of them are listed below

*2) **Precision and Recall**:* Precision is the ratio between the True Positives and all the Positives. For our model, it will be the measure of number of cars for which we correctly identified its safety category.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \tag{3}$$

Recall is the measure of our model correctly identifying True Positives. For all the cars in the given safety label, recall tells us how many we correctly predicted.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{4}$$

*3) **F1 Score**:* F1 score is the measure which combines both precision and recall. It is calculated as the harmonic mean of precision and recall -

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

III. The Problem

The Car Evaluation Database was derived from a simple hierarchical decision model. The prediction task is to classify a car based on its safety

A. *The Variables*

Buying - The price of the car.
Maintenance - The amount of maintenance required.
Doors - The number of doors.
Persons - The capacity of the car.
Lug boot - The size of the luggage boot in the car.
Safety - The estimated safety of the car.

B. *The Target Variable*

We have to predict the overall safety of the car. In the given data, there is an high imbalance and a lot of cars have unacceptable safety, denoted by 'unacc'. A countplot is shown below -

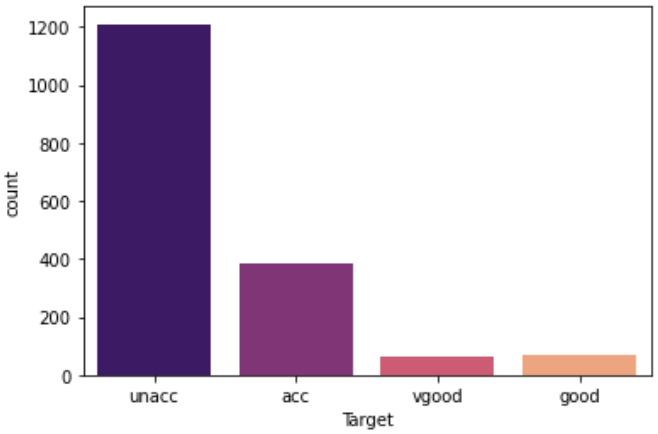

Fig. 2. Target Countplot

C. *Numerical Variables*

*1) **Doors**:* This variable indicates the number of doors in the car. To visualise the relationship of safety and number of doors, we plot the below given graph-

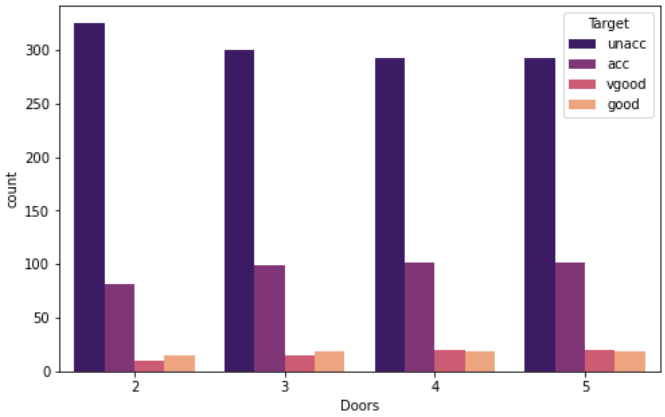

Fig. 3. Doors vs Safety

From the plot, we can see that there is not a significant difference between cars having 4 or more doors, but the cars having 2-3 doors are less safer.

*2) **Persons**:* This variable indicates capacity of the car. To visualise the relationship of safety and number of persons the car can hold, we plot the below given graph-

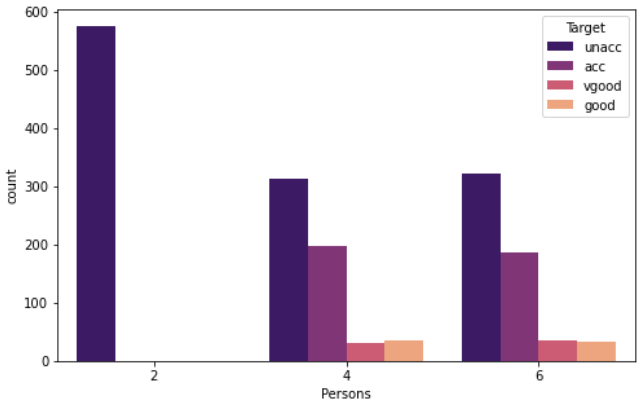

Fig. 4. Number of Persons vs Safety

From the plot, we can see that the cars having higher capacity are on the safer side.

D. Categorical Variables

*1) **Buying**:* This variable indicates the price of the car. It classifies them as vhigh, high, med, and low. To visualise the relationship of safety and buying price, we plot the below given graph-

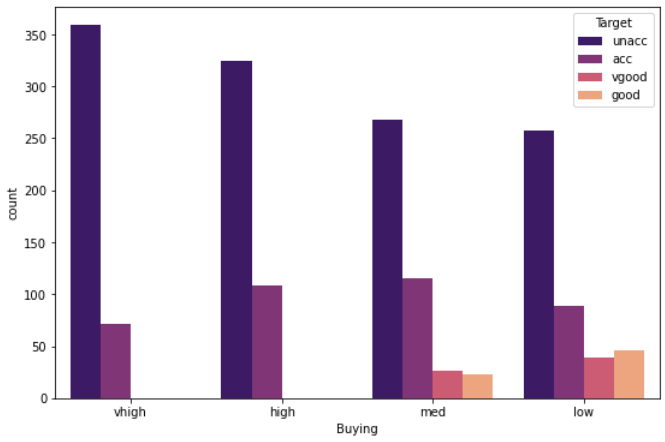

Fig. 5. Buying variable countplot

There seems no significant relationship between safety and buying price.

*2) **Maintenance**:* This variable indicates the amount of maintenance required for the car. It classifies them as vhigh, high, med, and low. To visualise the relationship of safety and required maintenance, we plot the below given graph-

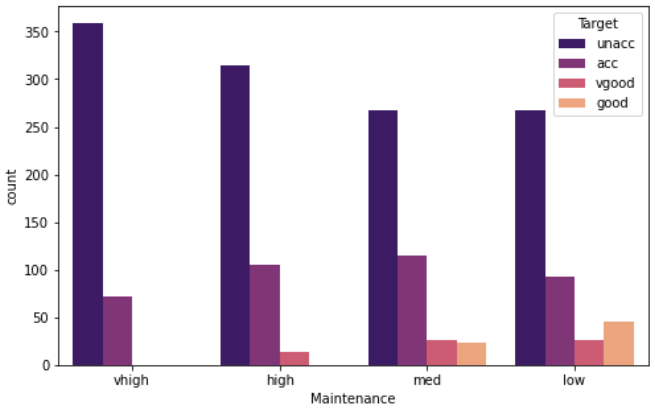

Fig. 6. Maintenance variable countplot

The plot indicates that lower maintenance cars are relatively safer.

*3) **Lug boot**:* This variable indicates the size of the luggage boot of the car. It classifies the size as small, med and big. To visualise the relationship of safety and required maintenance, we plot the below given graph-

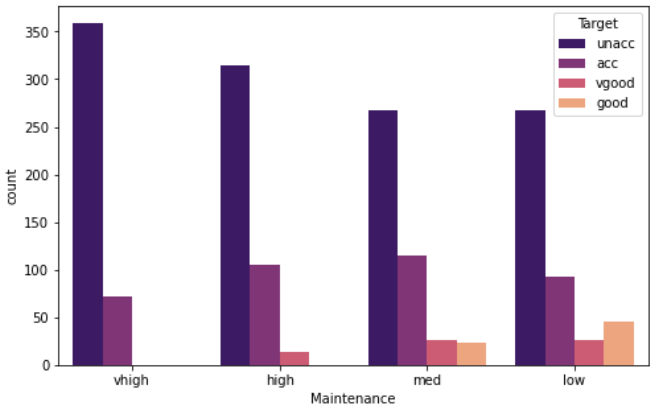

Fig. 7. Lug Boot variable countplot

The plot indicates a weak positive relationship between larger luggage boot and higher safety.

*4) **Safety**:* This variable indicates the estimated safety of the car. It classifies the size as low, med and big. To visualise the relationship of actual safety and estimated safety, we plot the below given graph-

As expected, the plot indicates that the estimations are largely correct and cars with higher safety rating are relatively

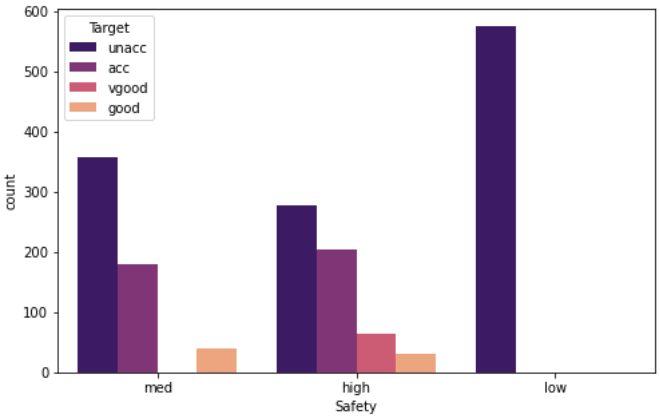

Fig. 8. Safety variable countplot

safer.

E. *Checking Correlation*

Through the below heatmap, we can see no high values of correlation among our numeric variables-

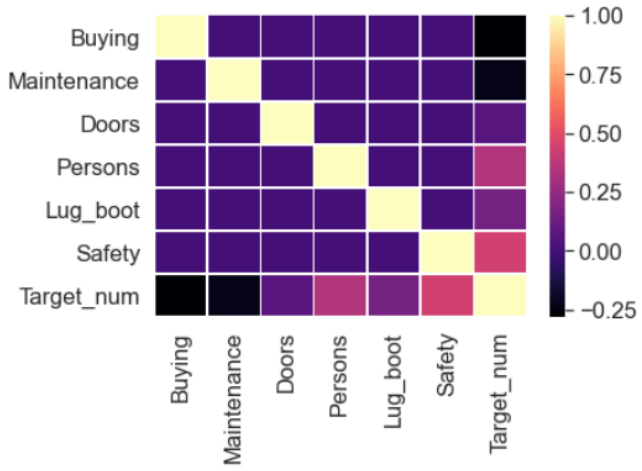

Fig. 9. Correlation Heatmap

We can see there is positive correlation of safety with number of persons and the estimated safety.

IV. The Model

A. *Train Test Split*

We have the following X and y for our final model, which we divide our data into training and testing data..
X = 'Buying', 'Maintenance', 'Doors', 'Persons', 'Lug boot', 'Safety' Y = Target

B. *Training the Model and Parameter Tuning*

Using scikit-learn's DecisionTreeClassifier() function on our X and y variables, we develop a Tree model.

We have to decide the number of layers, or the Max depth parameter. So we plot a graph between max depth and cross validation score. We witness an elbow shaped graph, we choose maxdepth as 11.

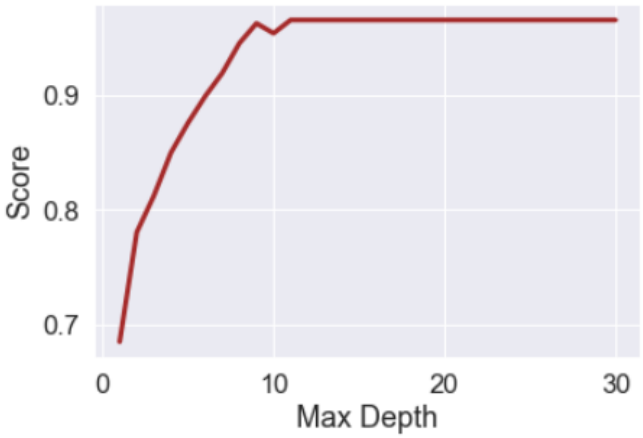

Fig. 10. Max Depth vs Score

C. *Predictions and Metrics*

The confusion matrix for our testing set in shown below

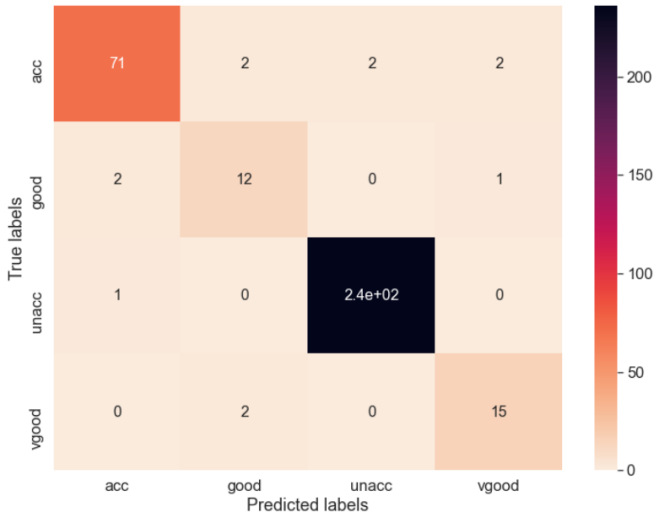

Fig. 11. Confusion Matrix for testing data

Based on the matrix, we can calculate the following (Fig 12) -

The f1 score is 0.97.

D. *Visualizing the tree*

A sample figure tree for our data is shown (Fig. 13)

	precision	recall	f1-score	support
acc	0.96	0.92	0.94	77
good	0.75	0.80	0.77	15
unacc	0.99	1.00	0.99	237
vgood	0.83	0.88	0.86	17
accuracy			0.97	346
macro avg	0.88	0.90	0.89	346
weighted avg	0.97	0.97	0.97	346

Fig. 12. Scores for testing data

V. Improvements in the model

A. *Random Search CV*

Hyperparameters are points of choice or configuration that allow a machine learning model to be customized for a specific task or dataset.In a model, we have to objectively search different values for model hyperparameters and choose a subset that results in a model that achieves the best performance on a given dataset. This is called hyperparameter optimization.

Random search is a technique where random combinations of the hyperparameters are used to find the best solution for the built model. We use the the scikit-learn's RandomizedSearchCV() function for our logistic regression.

The hyper parameters for decision trees are -

- **max depth** - It is the length of the longest path from the tree root to a leaf. The root node is considered to have a depth of 0
- **min samples leaf** - It specifies the minimum number of samples required to be at a leaf node.
- **criterion** - This parameter determines how the impurity of a split will be measured. The default value is "gini" but you can also use "entropy" as a metric for impurity.

B. *XGBoost*

XGBoost stands for eXtreme Gradient Boosting.
It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework.

XGBoost is an ensemble tree method that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture.

XGBoost uses the following optimization techniques -

- **Parallelization** - XGBoost approaches the process of sequential tree building using parallelized implementation. This is possible due to the interchangeable nature of loops used for building base learners; the outer loop that enumerates the leaf nodes of a tree, and the second inner loop that calculates the features.
- **Tree Pruning** - The stopping criterion for tree splitting within GBM framework is greedy in nature and depends on the negative loss criterion at the point of split. XGBoost uses 'max depth' parameter as specified instead of criterion first, and starts pruning trees backward.
- **Hardware Optimization** - This algorithm has been designed to make efficient use of hardware resources. This is accomplished by cache awareness by allocating internal buffers in each thread to store gradient statistics.

C. *Random Search for Hyper Parameter Tuning*

Using scikit-learn's RandomizedSearchCV function on the following params -
'max depth': [2, 3, 5, 10,11,12,13,14.15, 20],
'min samples leaf': [1,2,3, 5, 10, 20, 50, 100],
'criterion': ["gini", "entropy"]

The best values comes out to be -
criterion='entropy' and max depth=20

D. *Applying XGBoost*

XGBoost is one of the most powerful algorithms available and it works the best for our data. It gives a f1 score of 1 on the training set and 0.9826 on testing set. XGBoost clearly outperforms decision trees.

VI. Conclusions

We were able to achieve a F1 score of 0.97 with a max depth of 11. From our observations, we can conclude the following about the significant features-

- **Doors and Passengers** - Cars with higher number of doors and higher passenger capacity are safer than the smaller cars .
- **Price** - The price of the car does not affect its safety.
- **Luggage Boot Size** - Cars with bigger luggage boot tend to have better safety.
- **Estimated safety** - Quite intuitively, cars with higher estimated safety are more safer.
- **Maintenance** - Cars which require low maintenance are more safe than their high maintenance counterparts.

After improvements

- Using RandomizedSearchCV, the most optimal value of hyper parameters are criterion='entropy', max depth=20.
- XGBoost Classifier provides the best score of 0.9826.

Thus, as shown by both the qualitative and the quantitative analysis, the safety of the car depends on various factors.

REFERENCES

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, An introduction to statistical learning: With applications in r. New York, NY: Springer, 2021.

[2] "Machine learning decision tree classification algorithm - javatpoint," www.javatpoint.com. [Online]. Available: https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm. [Accessed: 29-Oct-2021].

[3] "1.10. decision trees," scikit. [Online]. Available: https://scikit-learn.org/stable/modules/tree.html

[4] "Decision tree learning," Wikipedia, 16-Oct-2021. [Online]. Available: https://en.wikipedia.org/wiki/Decisiontreelearning. [Accessed: 29-Oct-2021].

[5] M. Mithrakumar, "How to tune a Decision Tree?," Medium, 12-Nov-2019. [Online]. Available: https://towardsdatascience.com/how-to-tune-a-decision-tree-f03721801680. [Accessed: 03-Dec-2021].

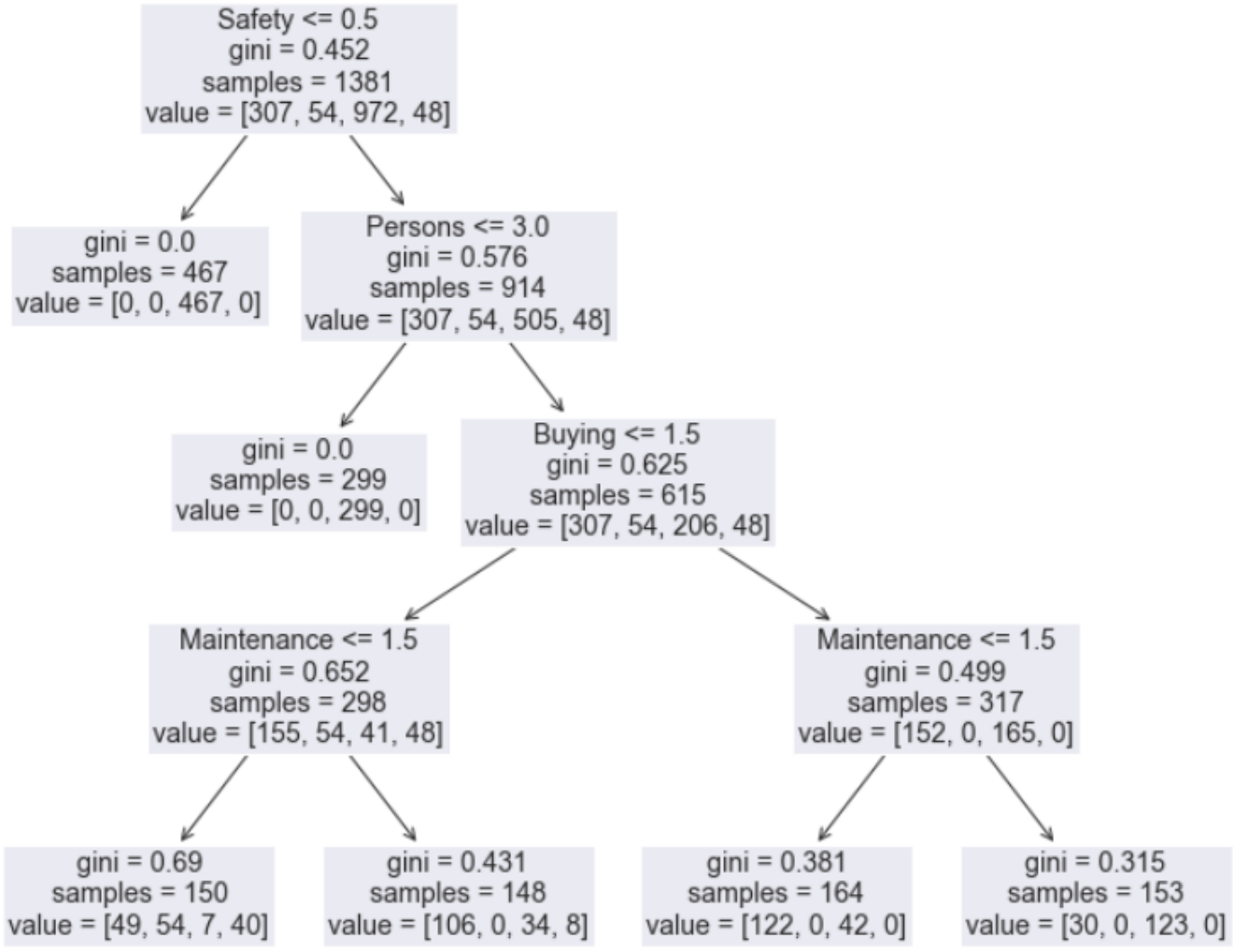

Fig. 13. Decision Tree

Assignment 5: Random Forest Classifier

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras
me18b114@smail.iitm.ac.in

***Abstract*—This paper gives a brief introduction of Random Forest Classifier and the basic mathematical theory behind it. We also look at some metrics to evaluate the performance of our model. Further, we apply the classifier on the Car Evaluation Database, derived from a simple hierarchical decision model. The prediction task is to classify based on its safety. Through exploratory data analysis and decision trees classifier, we investigate trends in data to predict which cars are safer. Then we evaluate our model using confusion matrix and F1 score. Our findings reveal some interesting trends.**

On the second attempt, the following additions have been made to the model -
1. Random Search and XGBoost theory.
2. Explored the hyper parameters of Random Forest Classifier.
3. Hyper parameter tuning with Random Search.
4. Applying XGBoost on the data.

I. Introduction

There are multiple types of algorithm methods used in machine learning. One such popular and commonly used machine learning method is Random Forest Classifier.

Random forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of over fitting to their training set. Random forests generally outperform decision trees.

In this paper, we apply Random Forest Classifier to the Car Evaluation Database, which was derived from a simple hierarchical decision model, to classify a car based on its safety.

The paper will first give a brief explanation of decision trees classifier. Then, we will try to solve the Income prediction and model our data-set by
1. Cleaning the data.
2. Exploratory and visual analysis.
3. Feature Engineering for Categorical Variables.
4. Checking for Correlation.
5. Training the model.
6. Evaluating the model.
7. Interpreting the results.

II. Random Forest Classifier Theory

A. *Classification*

Classification is a process of categorizing a given set of data into classes. Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category, label or class.

Many real world problems require classification rather than regression. In the classification setting we have a set of training observations (x1, y1), . . . , (xn, yn) that we can use to build a classifier.

Regression is not useful in problems like if we are trying to predict the medical condition of a patient in the hospital and we have 3 possible diagnoses- stroke, drug overdose, and epileptic seizure based on the symptoms of the patient. We can label these diseases at 1,2 and 3. In this case, even if we create a metric to accurately convert the symptoms to numeric data, regression won't be able to classify the data as it cannot give the output as 1,2 or 3. Alternatively, we can create a threshold that if predicted value is between 0.5 to 1.5, we will assign it to label 1. This brings us to the basic idea behind classification algorithms which can give qualitative response.

B. *Decision Trees*

A tree based classifier involves stratifying or segmenting the predictor space into a number of simple regions. In order to make a prediction for a given observation, we typically use the mean or the mode of the training observations in the region to which it belongs. Since the set of splitting rules used to segment the predictor space can be summarized in a tree, these types of approaches are known as decision tree methods. Tree-based methods are simple and useful for interpretation.

It is a tree-structured classifier, where internal nodes represent the features of a data set, branches represent the decision rules and each leaf node represents the outcome.

Terminology

- **Root Node** -Root node is from where the decision tree starts. It represents the entire data set, which further gets divided into two or more homogeneous sets.

- **Leaf Node** - Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting** - Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch Tree** - A tree formed by splitting the tree.
- **Pruning** -Pruning is the process of removing the unwanted branches from the tree.

C. *Working of Decision Tree Algorithm*

In a decision tree, for predicting the class of the given data set, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real data set) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree.

Attribute Selection - While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. There are two techniques for this -

*1) **Gini Index**:* Gini Impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. The Gini impurity can be computed by summing the probability of an item p_i with label i being chosen times the probability -

$$\sum_{k \neq i} p_k = 1 - p_i \tag{1}$$

, of a mistake in categorizing that item. It reaches its minimum (zero) when all cases in the node fall into a single target category.

To compute Gini impurity for a set of items with J classes, suppose $i \in \{1, 2, ..., J\}$,p_i be the fraction of items labeled with class *i* in the set.

$\mathrm{I}_G(p) = \sum_{i=1}^{J} \left(p_i \sum_{k \neq i} p_k \right)$

= $\sum_{i=1}^{J} p_i(1 - p_i)$

= $\sum_{i=1}^{J} (p_i - p_i^2)$

= $\sum_{i=1}^{J} p_i - \sum_{i=1}^{J} p_i^2$

= 1 - $\sum_{i=1}^{J} p_i^2$

*2) **Information Gain**:* Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute. It calculates how much information a feature provides us about a class. According to the value of information gain, we split the node and build the decision tree. A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

Information Gain = Entropy(S) - [(Weighted Avg)* Entropy(each feature)]

Entropy is defined as -

$$(T) = \mathrm{I}_E\left(p_1, p_2, \ldots, p_J\right) = -\sum_{i=1}^{J} p_i \log_2 p_i \tag{2}$$

where $p_1, p_2, \ldots$ are fractions that add up to 1 and represent the percentage of each class present in the child node that results from a split in the tree.

$$\overbrace{E_A(\mathrm{IG}(T,a))}^{\text{expected information gain}} = \overbrace{(T)}^{\text{entropy (parent)}} - \overbrace{(T \mid A)}^{\text{weighted sum of entropies (children)}}$$
$$= -\sum_{i=1}^{J} p_i \log_2 p_i - \sum_{a} p(a) \sum_{i=1}^{J} -\Pr(i \mid a) \log_2 \Pr(i \mid a)$$

That is, the expected information gain is the mutual information, meaning that on average, the reduction in the entropy of T is the mutual information.

D. *Bagging*

Bagging, also known as bootstrap aggregation, is the ensemble learning method that is commonly used to reduce variance within a noisy dataset. In bagging, a random sample of data in a training set is selected with replacement—meaning that the individual data points can be chosen more than once.

The training algorithm for random forests applies the general technique of bagging to tree learners. Given a training set $X = X_1, X_2, ...X_n$ with responses $Y = y_1, y_2, ..., y_n$, bagging repeatedly selects a random sample with replacement of the training set and fit trees to these samples.

For b = 1,...,B:
1. Sample, with replacement, n training examples from X, Y; call these X_b, Y_b.
2. Train a classification tree f_b on X_b, Y_b.

After training, predictions for unseen samples x can be made by averaging the predictions from all the individual regression trees on x:

$$\hat{f} = \frac{1}{B} \sum_{b=1}^{B} f_b(x) \tag{3}$$

or by taking the majority vote in the case of classification trees.

The number of samples/trees, B, is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set. An optimal number of trees B can be found using cross-validation, or by observing the out-of-bag error: the mean prediction error on each training sample xi, using only the trees that did not have xi in their bootstrap sample. The training and test error tend to level off after some number of trees have been fit.

E. Random Forest Classifier

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.

Random Forest uses a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. This process is sometimes called feature bagging. The reason for doing this is the correlation of the trees in an ordinary bootstrap sample: if one or a few features are very strong predictors for the response variable (target output), these features will be selected in many of the B trees, causing them to become correlated.

F. Advantages of Random Forest

There are many advantages of random forest over decision trees -

- It reduces overfitting in decision trees and helps to improve the accuracy.
- It is flexible to both classification and regression problems.
- It works well with both categorical and continuous values.
- Normalising of data is not required as it uses a rule-based approach.

G. Evaluating the model

In model evaluation, we measure the extent to which the model fits the data.

*1) **Confusion Matrix**:* A confusion matrix is a table that shows the performance of a machine learning model. It displays how many of the model's predictions were right and how many were incorrect and which classes did the model succeed in and which did it fail in. It offers us an understanding of how the model works. If a model fails for a certain class, we may investigate it, figure out why, and try to improve the model.

There are various metrics based on confusion matrix. Some of them are listed below

*2) **Precision and Recall**:* Precision is the ratio between the True Positives and all the Positives. For our model, it will be the measure of number of cars for which we correctly identified its safety category.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \tag{4}$$

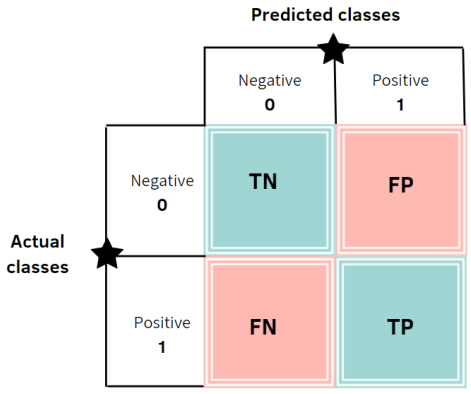

Fig. 1. Sample Confusion Matrix

Recall is the measure of our model correctly identifying True Positives. For all the cars in the given safety label, recall tells us how many we correctly predicted.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{5}$$

*3) **F1 Score**:* F1 score is the measure which combines both precision and recall. It is calculated as the harmonic mean of precision and recall

-

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

III. THE PROBLEM

The Car Evaluation Database was derived from a simple hierarchical decision model. The prediction task is to classify a car based on its safety

A. The Variables

Buying - The price of the car.
Maintenance - The amount of maintenance required.
Doors - The number of doors.
Persons - The capacity of the car.
Lug boot - The size of the luggage boot in the car.
Safety - The estimated safety of the car.

B. The Target Variable

We have to predict the overall safety of the car. In the given data, there is an high imbalance and a lot of cars have unacceptable safety, denoted by 'unacc'. A countplot is shown below -

C. Numerical Variables

*1) **Doors**:* This variable indicates the number of doors in the car. To visualise the relationship of safety and number of doors, we plot the below given graph-

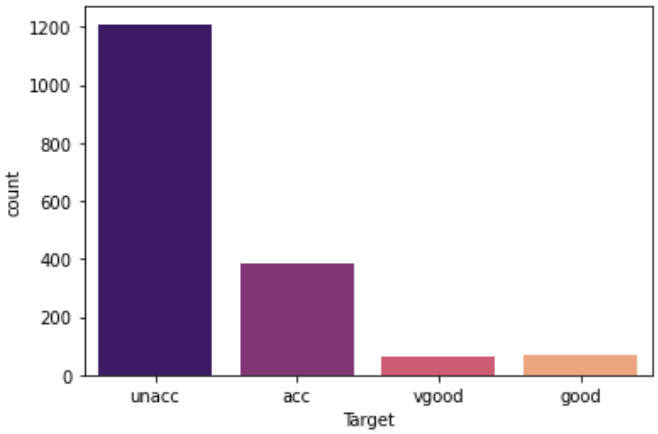

Fig. 2. Target Countplot

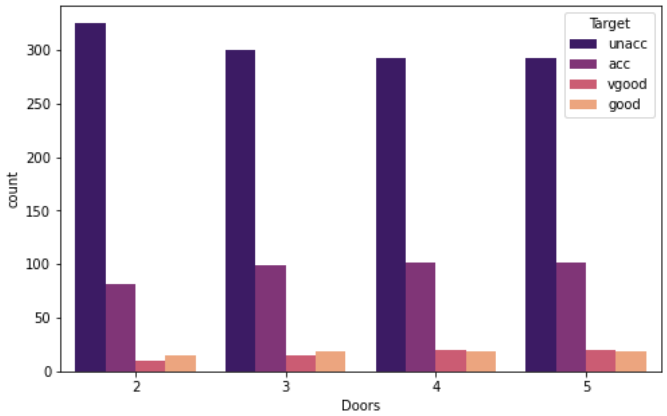

Fig. 3. Doors vs Safety

From the plot, we can see that there is not a significant difference between cars having 4 or more doors, but the cars having 2-3 doors are less safer.

*2) **Persons**:* This variable indicates capacity of the car. To visualise the relationship of safety and number of persons the car can hold, we plot the below given graph-

From the plot, we can see that the cars having higher capacity are on the safer side.

D. Categorical Variables

*1) **Buying**:* This variable indicates the price of the car. It classifies them as vhigh, high, med, and low. To visualise the relationship of safety and buying price, we plot the below given graph-

There seems no significant relationship between safety and buying price.

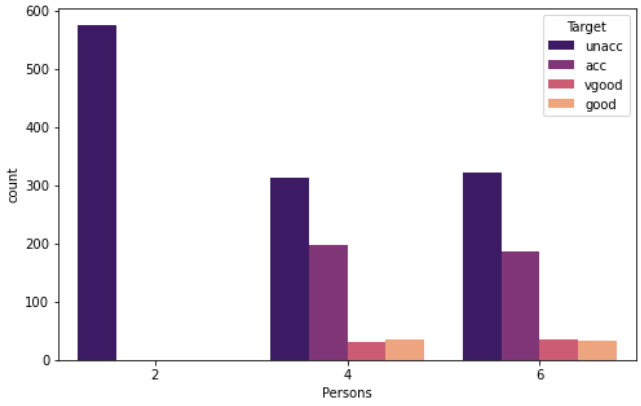

Fig. 4. Number of Persons vs Safety

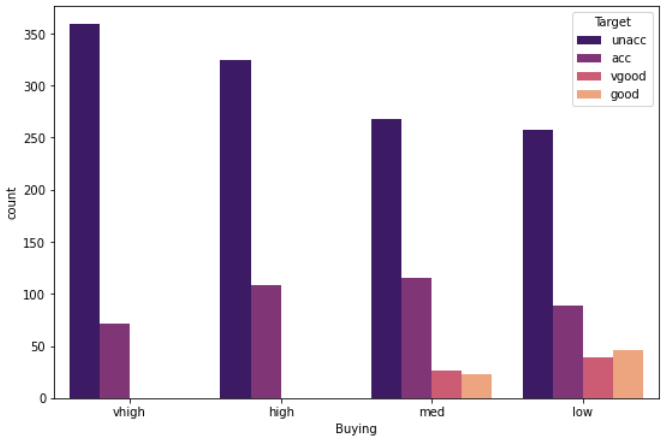

Fig. 5. Buying variable countplot

*2) **Maintenance**:* This variable indicates the amount of maintenance required for the car. It classifies them as vhigh, high, med, and low. To visualise the relationship of safety and required maintenance, we plot the below given graph-

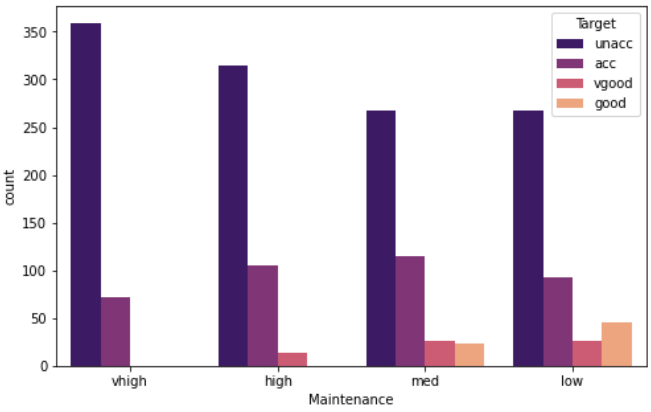

Fig. 6. Maintenance variable countplot

The plot indicates that lower maintenance cars are relatively safer.

*3) **Lug boot**:* This variable indicates the size of the luggage boot of the car. It classifies the size as small, med and big. To visualise the relationship of safety and required maintenance, we plot the below given graph-

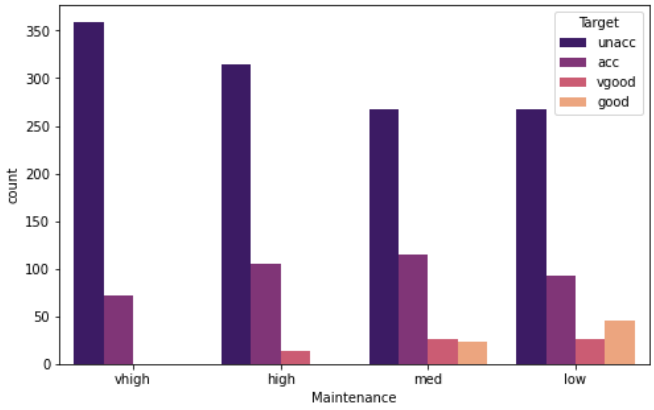

Fig. 7. Lug Boot variable countplot

The plot indicates a weak positive relationship between larger luggage boot and higher safety.

E. Safety

This variable indicates the estimated safety of the car. It classifies the size as low, med and big. To visualise the relationship of actual safety and estimated safety, we plot the below given graph-

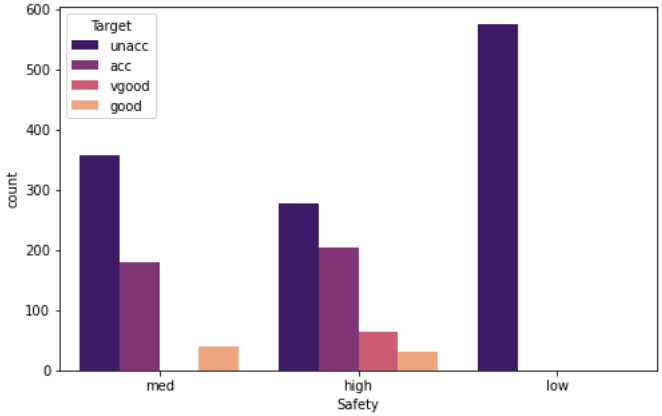

Fig. 8. Safety variable countplot

As expected, the plot indicates that the estimations are largely correct and cars with higher safety rating are relatively safer.

F. Checking Correlation

Through the below heatmap, we can see no high values of correlation among our numeric variables-

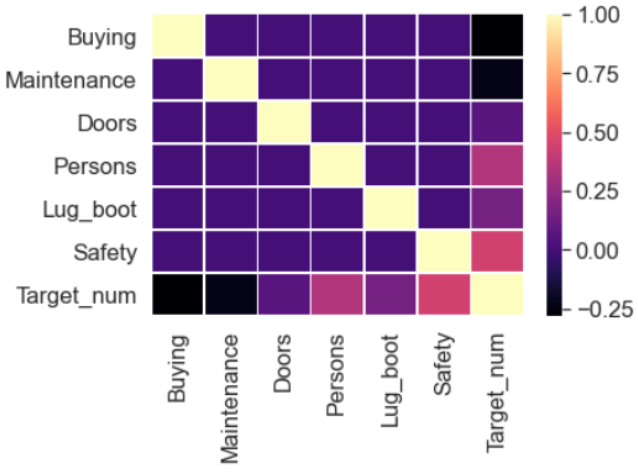

Fig. 9. Correlation Heatmap

We can see there is positive correlation of safety with number of persons and the estimated safety.

IV. The Model

A. Train Test Split

We have the following X and y for our final model, which we divide our data into training and testing data..
X = 'Buying', 'Maintenance', 'Doors', 'Persons', 'Lug boot', 'Safety' Y = Target

B. Training the Model and Parameter Tuning

Using scikit-learn.ensemble's RandomForestClassifier() function on our X and y variables, we develop a Tree model.

We have to decide the number of layers, or the Max depth parameter. So we plot a graph between max depth and cross validation score. We witness an elbow shaped graph, we choose maxdepth as 11.

C. Predictions and Metrics

The confusion matrix for our testing set in shown below

Based on the matrix, we can calculate the following (Fig 12) -

The f1 score is 0.98.

V. Improvements in the Model

A. Random Search CV

Hyperparameters are points of choice or configuration that allow a machine learning model to be customized for a specific

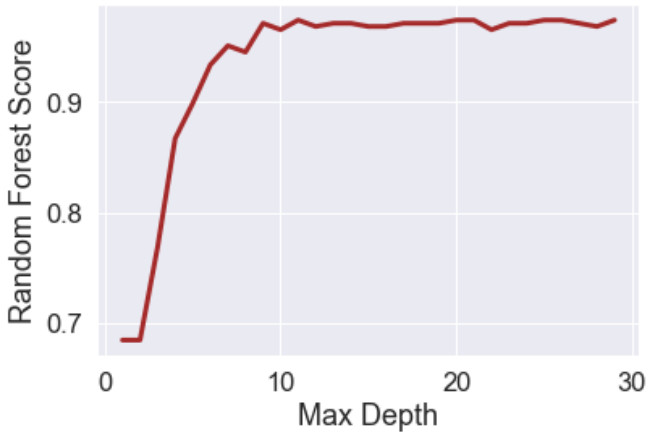

Fig. 10. Max Depth vs Score

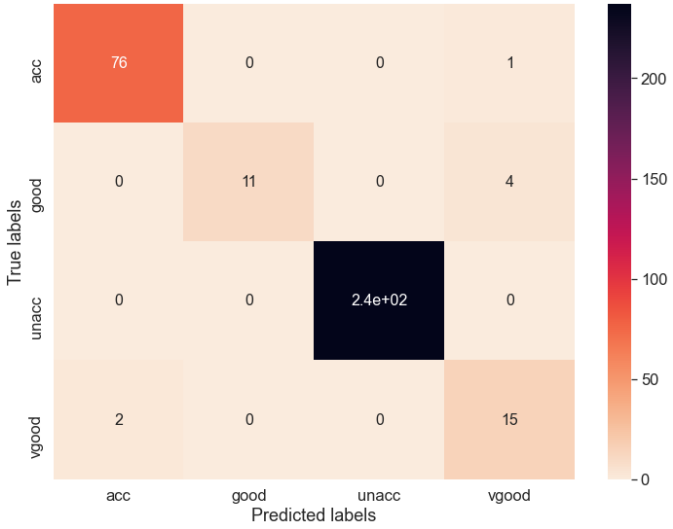

Fig. 11. Confusion Matrix for testing data

	precision	recall	f1-score	support
acc	0.97	0.99	0.98	77
good	1.00	0.73	0.85	15
unacc	1.00	1.00	1.00	237
vgood	0.75	0.88	0.81	17
accuracy			0.98	346
macro avg	0.93	0.90	0.91	346
weighted avg	0.98	0.98	0.98	346

Fig. 12. Scores for testing data

task or dataset.In a model, we have to objectively search different values for model hyperparameters and choose a subset that results in a model that achieves the best performance on a given dataset. This is called hyperparameter optimization.

Random search is a technique where random combinations of the hyperparameters are used to find the best solution for the built model. We use the the scikit-learn's RandomizedSearchCV() function for our logistic regression.

The hyper parameters for decision trees are -

- **max depth** - It is the length of the longest path from the tree root to a leaf. The root node is considered to have a depth of 0
- **max features** - These are the maximum number of features Random Forest is allowed to try in individual tree.
- **min samples leaf** - It specifies the minimum number of samples required to be at a leaf node.
- **min samples split** -It represents the minimum number of samples required to split an internal node.
- **n estimators** - the number of trees you want to build before taking the maximum voting or averages of predictions.

B. XGBoost

XGBoost stands for eXtreme Gradient Boosting.
It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework.

XGBoost is an ensemble tree method that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture.

XGBoost uses the following optimization techniques -

- **Parallelization** - XGBoost approaches the process of sequential tree building using parallelized implementation. This is possible due to the interchangeable nature of loops used for building base learners; the outer loop that enumerates the leaf nodes of a tree, and the second inner loop that calculates the features.

- **Tree Pruning** - The stopping criterion for tree splitting within GBM framework is greedy in nature and depends on the negative loss criterion at the point of split. XGBoost uses 'max depth' parameter as specified instead of criterion first, and starts pruning trees backward.

- **Hardware Optimization** - This algorithm has been designed to make efficient use of hardware resources. This

is accomplished by cache awareness by allocating internal buffers in each thread to store gradient statistics.

C. *Random Search for Hyper Parameter Tuning*

Using scikit-learn's RandomizedSearchCV function on the following params -
'n estimators' : [30,40,50,60,70,80,100],
'max features' : ['auto','sqrt'],
'max depth': [5, 10, 15],
'min samples leaf': [1, 2, 5, 10],
'min samples split': [2,4,8]

The best values comes out to be -
max depth=15, n estimators=60

D. *Applying XGBoost*

XGBoost is one of the most powerful algorithms available and it works the best for our data. It gives a f1 score of 1 on the training set and 0.9826 on testing set. This implies that XGBoost is outperforming random forest classifier.

VI. Conclusions

We were able to achieve a F1 score of 0.98 with a max depth of 11 by fitting Random Forest Classifier on our data set. From our observations, we can conclude the following about the significant features-

- **Doors and Passengers** - Cars with higher number of doors and higher passenger capacity are safer than the smaller cars .
- **Price** - The price of the car does not affect its safety.
- **Luggage Boot Size** - Cars with bigger luggage boot tend to have better safety.
- **Estimated safety** - Quite intuitively, cars with higher estimated safety are more safer.
- **Maintenance** - Cars which require low maintenance are more safe than their high maintenance counterparts.

After Improvements

- Using RandomizedSearchCV, the most optimal value of hyper parameters are max depth=15 and n estimators=60.
- XGBoost Classifier provides the best score of 0.9826.

Thus, as shown by both the qualitative and the quantitative analysis, the safety of the car depends on various factors.

References

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, An introduction to statistical learning: With applications in r. New York, NY: Springer, 2021.

[2] T. Hastie, J. Friedman, and R. Tisbshirani, The elements of Statistical Learning: Data Mining, Inference, and prediction. New York: Springer, 2017.

[3] "Random Forest," Wikipedia, 02-Nov-2021. [Online]. Available: https://en.wikipedia.org/wiki/Randomforest. [Accessed: 11-Nov-2021].

[4] T. Yiu, "Understanding random forest," Medium, 29-Sep-2021. [Online]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2. [Accessed: 11-Nov-2021].

[5] "Sklearn.ensemble.randomforestclassifier," Available: https://scikit-learn.org/stable/module/sklearn.ensemble.RandomForestClassifier.html. [Accessed: 11-Nov-2021].

Assignment 6: Support Vector Machine

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras
me18b114@smail.iitm.ac.in

***Abstract*—This paper gives a brief introduction of Support Vector Machines and the basic mathematical theory behind it. We also look at some metrics to evaluate the performance of our model. Further, we apply the classifier on the Neutron Star Database, containing the various parameters of the star emissions. The prediction task is to classify the star as Pulsars and not Pulsars. Through exploratory data analysis and support vector machines, we investigate trends in data to predict what characteristics make a star a pulsar. Then we evaluate our model using confusion matrix and F1 score. Our findings reveal some interesting trends.**

I. Introduction

There are multiple types of algorithm methods used in machine learning. One such popular and commonly used supervised machine learning method is Support Vector Machine.

The SVM algorithm's purpose is to find the optimum line or decision boundary for categorising n-dimensional space into classes so that additional data points may be readily placed in the proper category in the future. This best decision boundary is called a hyperplane. The extreme points/vectors that assist create the hyperplane are chosen via SVM. Support vectors are the extreme instances, and the method is called a Support Vector Machine.

In this paper, we apply the SVM algorithm to predict whether a neutron star is a pulsar star or not based on the various given features of the star. The features are obtained from the integrated pulse profile and the DM-SNR curve.

The paper will first give a brief explanation of support vector machines. Then, we will try to solve the pulsar prediction and model our data-set by
1. Cleaning the data.
2. Exploratory and visual analysis.
3. Scaling the data.
4. Checking for Correlation.
5. Training the model.
6. Evaluating the model.
7. Interpreting the results.

II. Support Vector Machines Theory

A. *Classification*

Classification is a process of categorizing a given set of data into classes. Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category, label or class.

Many real world problems require classification rather than regression. In the classification setting we have a set of training observations (x1, y1), . . . , (xn, yn) that we can use to build a classifier.

Regression is not useful in problems like if we are trying to predict the medical condition of a patient in the hospital and we have 3 possible diagnoses- stroke, drug overdose, and epileptic seizure based on the symptoms of the patient. We can label these diseases at 1,2 and 3. In this case, even if we create a metric to accurately convert the symptoms to numeric data, regression won't be able to classify the data as it cannot give the output as 1,2 or 3. Alternatively, we can create a threshold that if predicted value is between 0.5 to 1.5, we will assign it to label 1. This brings us to the basic idea behind classification algorithms which can give qualitative response.

B. *Support Vector Machines*

*1) **Hyperplanes**:* A hyperplane is a subspace whose dimension is one less than that of its ambient space. For example, if a space is 3-dimensional then its hyperplanes are the 2-dimensional planes, while if the space is 2-dimensional, its hyperplanes are the 1-dimensional lines.

In 2D Cartesian coordinates, such a hyperplane can be described with a single linear equation of the following form (where at least one of the a_i is non-zero and b is an arbitrary constant):

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n = b. \tag{1}$$

Any hyperplane of a Euclidean space has exactly two unit normal vectors.

Affine hyperplanes are used to define decision boundaries in many machine learning algorithms such as linear-combination (oblique) decision trees, and perceptrons.

*2) **Linear SVM**:* We are given a training dataset of n points of the form $(x_1, y_1)....(x_n, y_n)$ where the y_i are either 1 or 1, each indicating the class to which the point x_i belongs. We want to find the "maximum-margin hyperplane" that divides the group of points x_i for which $y_i = 1$ from the group of points for which $y_i = -1$, which is defined so that the distance between the hyperplane and the nearest point x_i from either group is maximized.

Any hyperplane can be written as the set of points $\mathbf{x}\mathbf{x}satisfying$

$$w^T x - b = 0 \tag{2}$$

*3) **The cost function**:* To compute the SVM classifier, we have to minimize the following expression -

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i - b)\right)\right] + \lambda\|\mathbf{w}\|^2. \tag{3}$$

To solve it we can use either quadratic programming, gradient descent etc.

*4) **Quadratic Programming for SVM**:* The cost function can be rewritten as a constrained optimization problem with a differentiable objective function in the following way.

For each $i \in \{1, \ldots, n\}$ we introduce a variable $\zeta_i = \max\left(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i - b)\right)$. Note that ζ_i is the smallest nonnegative number satisfying $y_i(\mathbf{w}^T\mathbf{x}_i - b) \geq 1 - \zeta_i$.

Thus we can rewrite the optimization problem as follows

$$\text{minimize } \frac{1}{n}\sum_{i=1}^{n}\zeta_i + \lambda\|\mathbf{w}\|^2$$

$$\text{subject to } \mathrm{y}_i(\mathbf{w}^T\mathbf{x}_i - b) \geq 1 - \zeta_i \text{ and } \zeta_i \geq 0, \text{ for all } i \tag{4}$$

*5) **Sub-gradient descent**:* Sub-gradient descent algorithms for the SVM work directly with the expression.

$$f(\mathbf{w}, b) = \left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i - b)\right)\right] + \lambda\|\mathbf{w}\|^2 \tag{5}$$

The traditional gradient descent (or SGD) methods can be adapted, where instead of taking a step in the direction of the function's gradient, a step is taken in the direction of a vector selected from the function's sub-gradient. This approach has the advantage that, for certain implementations, the number of iterations does not scale with n, the number of data points.

C. Advantages of SVM

- SVM has L2 Regularization feature. So, it has good generalization capabilities which prevent it from over-fitting.
- SVM can be used to solve both classification and regression problems.
- A small change to the data does not greatly affect the hyperplane and hence the SVM. So the SVM model is stable
- SVM can efficiently handle non-linear data.

D. Evaluating the model

In model evaluation, we measure the extent to which the model fits the data.

*1) **Confusion Matrix**:* A confusion matrix is a table that shows the performance of a machine learning model. It displays how many of the model's predictions were right and how many were incorrect and which classes did the model succeed in and which did it fail in. It offers us an understanding of how the model works. If a model fails for a certain class, we may investigate it, figure out why, and try to improve the model.

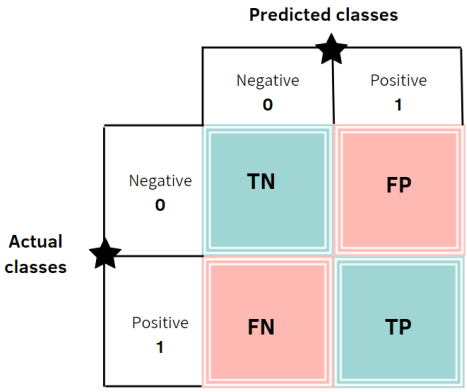

Fig. 1. Sample Confusion Matrix

There are various metrics based on confusion matrix. Some of them are listed below

*2) **Precision and Recall**:* Precision is the ratio between the True Positives and all the Positives. For our model, it will be the measure of number of cars for which we correctly identified its safety category.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \tag{6}$$

Recall is the measure of our model correctly identifying True Positives. For all the cars in the given safety label, recall tells us how many we correctly predicted.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{7}$$

*3) **F1 Score**:* F1 score is the measure which combines both precision and recall. It is calculated as the harmonic mean of precision and recall

-

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8}$$

III. The Problem

Pulsars are a rare type of Neutron star that produces radio emissions detectable here on Earth. They are of considerable scientific interest as probes of space-time, the inter-stellar medium, and states of matter. Machine learning tools are now being used to automatically label pulsar candidates to facilitate rapid analysis. The key task is to Predict if a star is a pulsar start or not. 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 (folded profile). This is an array of continuous variables that describe a longitude-resolved version of the signal that has been averaged in both time and frequency. The remaining four variables are similarly obtained from the DM-SNR curve.

A. *Cleaning the Data*

We check for null values in all columns and find that the around 20 percent data is missing in 'Excess kurtosis of the integrated profile' and 'Standard deviation of the DM-SNR curve' variables and 10 percent data is missing in the 'Skewness of the DM-SNR curve' variable, as shown in the figure below -

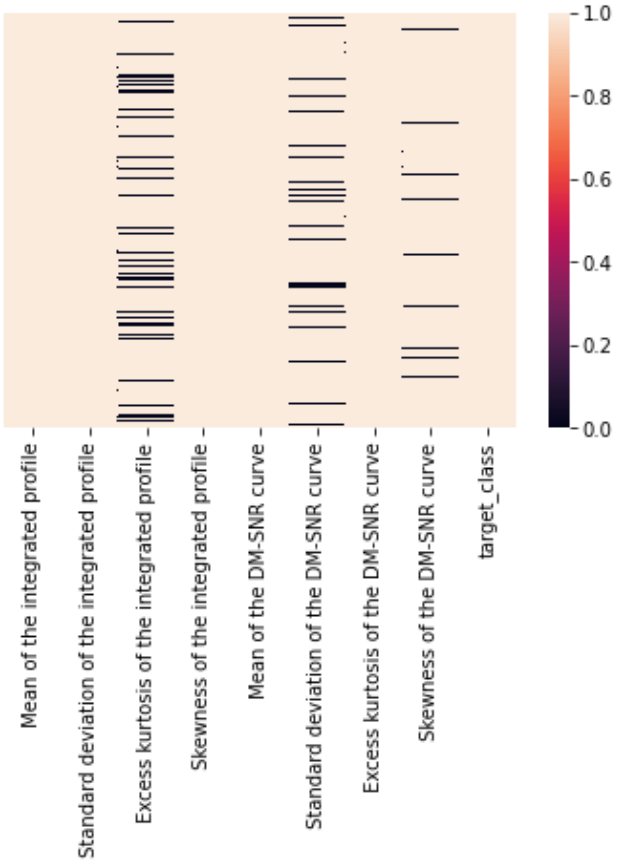

Fig. 2. Missing Values Heatmap

We impute the missing values with the respective column medians.

B. *The Variables*

- Mean of the integrated profile
- Standard deviation of the integrated profile
- Excess kurtosis of the integrated profile
- Skewness of the integrated profile
- Mean of the DM-SNR curve
- Standard deviation of the DM-SNR curve
- Excess kurtosis of the DM-SNR curve
- Skewness of the DM-SNR curve

C. *The Target Variable*

We have to predict whether a neutron star is a pulsar or not. In the given data, there is an high imbalance. A countplot is shown below -

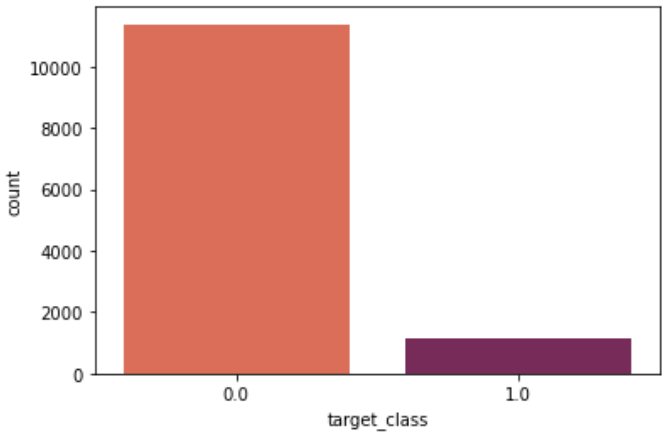

Fig. 3. Target Countplot

D. *Numerical Variables*

*1) **Mean of the integrated profile**:* This variable indicates the mean of the integrated profile of the neutron stars. To visualize its relationship with the target variable, we draw a boxplot -

From the plot, we can see that the pulsar stars' mean of the integrated profile is lesser than non-pulsars.

*2) **Standard deviation of the integrated profile**:* This variable indicates the std. deviation of the integrated profile of the neutron stars. To visualize its relationship with the target variable, we draw a boxplot -

From the plot, we can see that the pulsar stars' std. deviation of the integrated profile is lesser than non-pulsars. This implies the integrated profile of pulsars is less spread out.

*3) **Excess kurtosis of the integrated profile**:* This variable indicates the excess kurtosis of the integrated profile of the neutron stars. To visualize its relationship with the target variable, we draw a boxplot -

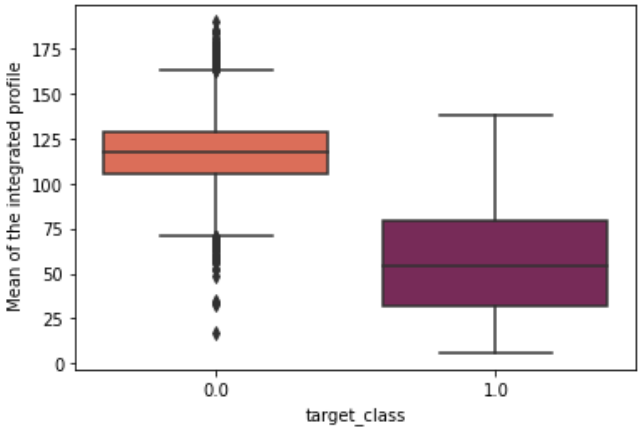

Fig. 4. Mean of the integrated profile vs target

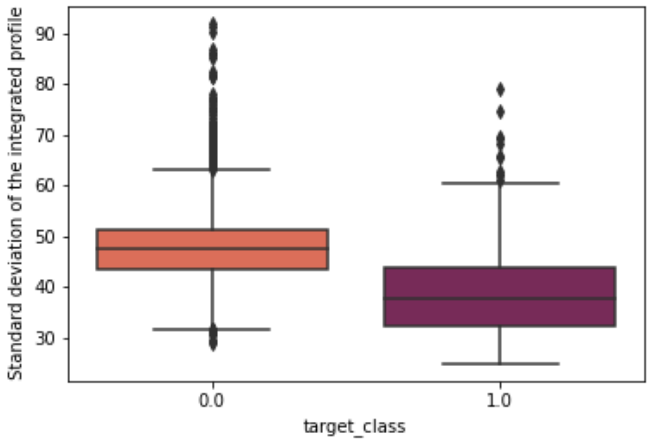

Fig. 5. Std. Deviation of the integrated profile vs target

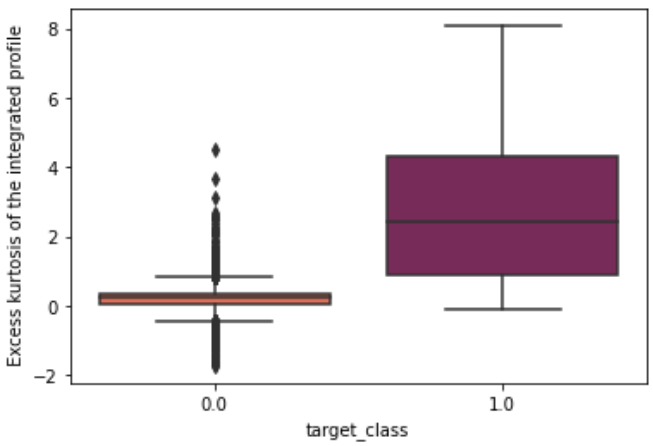

Fig. 6. Excess kurtosis of the integrated profile vs target

From the plot, we can see that the pulsar stars' excess kurtosis of the integrated profile is more than the non-pulsars.

4) ***Skewness of the integrated profile****:* This variable indicates the skewness of the integrated profile of the neutron stars. To visualize its relationship with the target variable, we draw a boxplot -

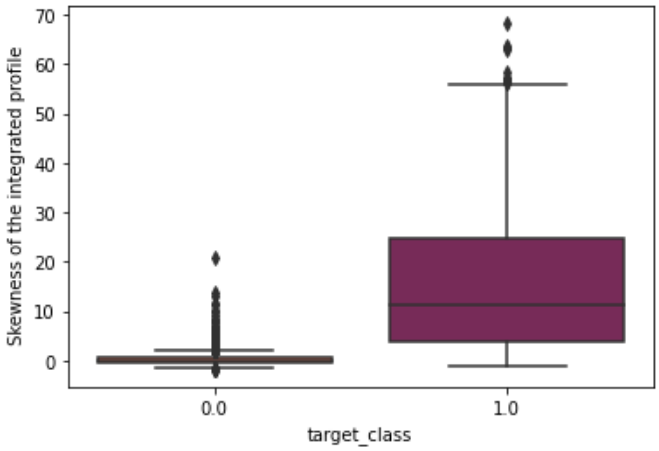

Fig. 7. Skewness of the integrated profile vs target

From the plot, we can see that the pulsar stars' skewness of the integrated profile is more than the non-pulsars.

5) ***Mean of the DM-SNR curve****:* This variable indicates the Mean of the DM-SNR curve of the neutron stars. To visualize its relationship with the target variable, we draw a boxplot -

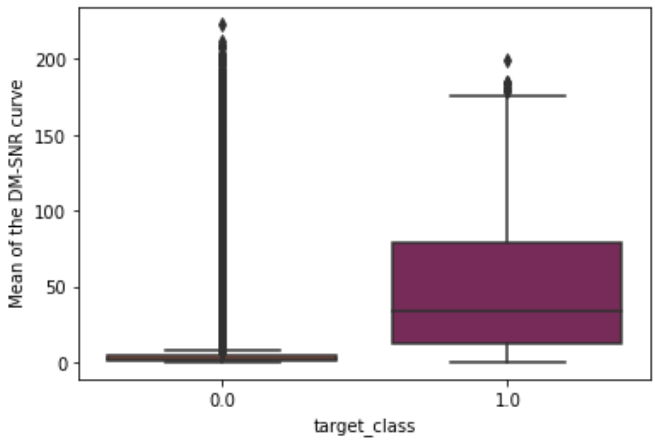

Fig. 8. Mean of the DM-SNR curve vs target

From the plot, we can see that the pulsar stars' mean of the DM-SNR curve is more than the non-pulsars.

*6) **Standard deviation of the DM-SNR curve:*** This variable indicates the std. deviation of the DM-SNR curve of the neutron stars. To visualize its relationship with the target variable, we draw a boxplot -

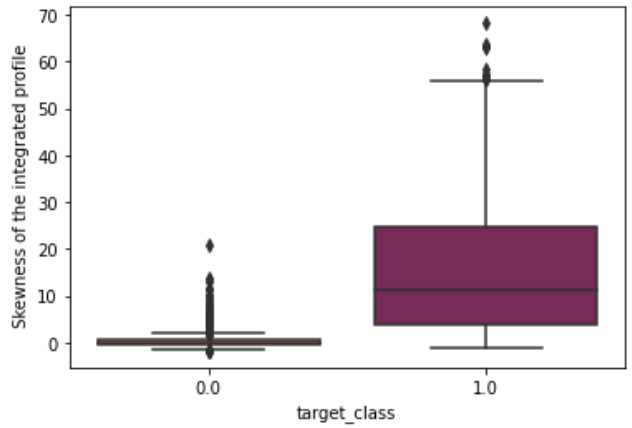

Fig. 9. Std. deviation of the DM-SNR curve vs target

From the plot, we can see that the pulsar stars' std. deviation of the DM-SNR curve is more than the non-pulsars. It implies that the DM-SNR curve is more spread out.

*7) **Excess kurtosis of the DM-SNR curve:*** This variable indicates the excess kurtosis of the DM-SNR curve of the neutron stars. To visualize its relationship with the target variable, we draw a boxplot -

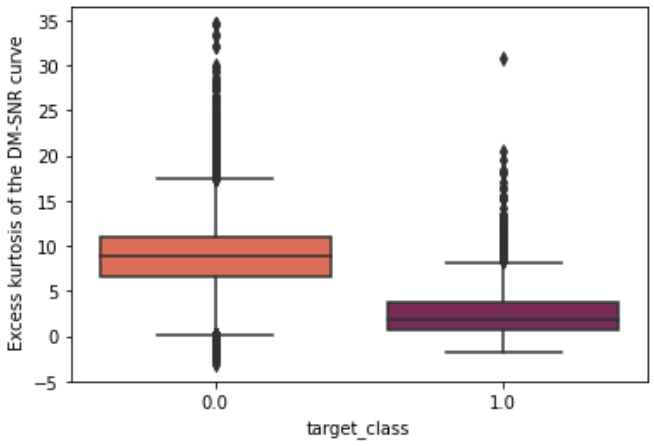

Fig. 10. Excess kurtosis of the DM-SNR curve vs target

From the plot, we can see that the pulsar stars' excess kurtosis of the DM-SNR curve is lesser than the non-pulsars.

*8) **Skewness of the DM-SNR curve:*** This variable indicates the skewness of the DM-SNR curve of the neutron stars. To visualize its relationship with the target variable, we draw a boxplot -

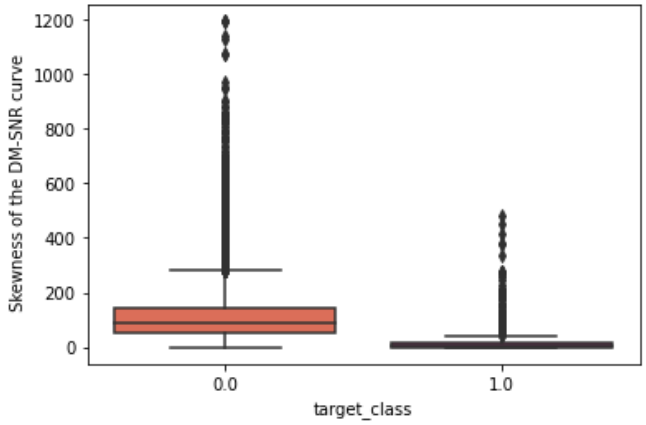

Fig. 11. Skewness of the DM-SNR curve vs target

From the plot, we can see that the pulsar stars' skewness of the DM-SNR curve is klesser than the non-pulsars.

E. Pairplot and Correlation

Through the below pairplot and correlation heatmap, we can see the values of correlation among our numeric variables-

Fig. 12. Pairplot

There is a high positive correlation between following features:

Excess kurtosis of the integrated profile - Skewness of the integrated profile (0.87)

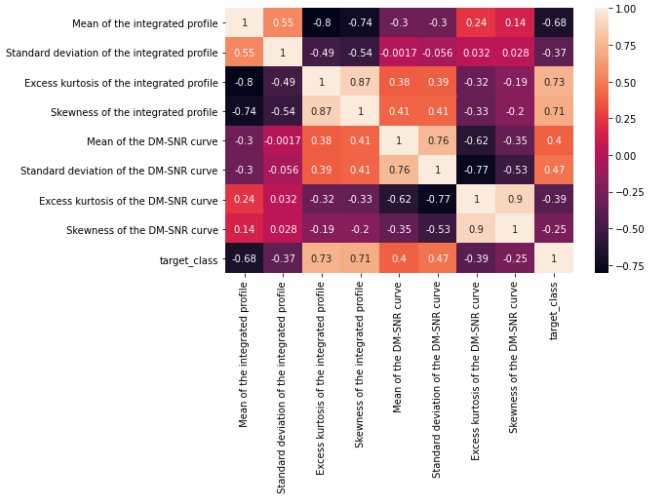

Fig. 13. Correlation Heatmap

Mean of the DM-SNR curve - Standard deviation of the DM-SNR curve(0.76)
Excess kurtosis of the DM-SNR curve - Skewness of the DM-SNR curve (0.9)

There is a high negative correlation between following features:

Mean of the integrated profile - Excess kurtosis of the integrated profile (-0.8)
Mean of the integrated profile - Skewness of the integrated profile (-0.74)
Standard deviation of the DM-SNR curve - Excess kurtosis of the DM-SNR curve (-0.77)

IV. The Model

A. Train Test Split

We have the following X and y for our final model, which we divide our data into training and testing data..
X = [' Mean of the integrated profile', ' Standard deviation of the integrated profile', ' Excess kurtosis of the integrated profile', ' Skewness of the integrated profile', ' Mean of the DM-SNR curve', ' Standard deviation of the DM-SNR curve', ' Excess kurtosis of the DM-SNR curve', ' Skewness of the DM-SNR curve']
Y = Target class

B. Training the Model and Grid Search

Using scikit-learn.ensemble's SVC() function on our X and y variables, we develop a support vector classifier model.

To find out the best hyper-parameters for SVC model, we employ scikit-learn's RandomSearchCV method to do a grid search with the below given parameters -

'C': [0.01,0.1, 1, 10, 100, 1000],
'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
'kernel': ['rbf'],
'tol':[0.01,0.001,0.0001],
'degree': [2,3,4,5]

C. Best Model

The grid search yielded the following results for the best model -
SVC(C=100, degree=5, gamma=0.001, random state=1)

D. Predictions and Metrics

The confusion matrix for our testing set in shown below

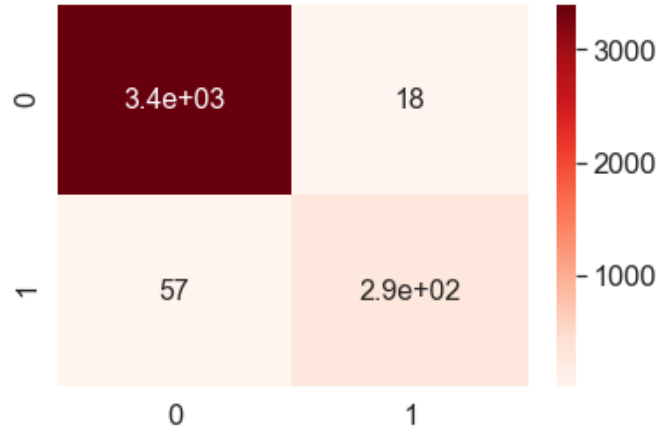

Fig. 14. Confusion Matrix for testing data

Based on the matrix, we can calculate the following (Fig 12) -

	precision	recall	f1-score	support
0.0	0.98	0.99	0.99	3415
1.0	0.94	0.83	0.88	344
accuracy			0.98	3759
macro avg	0.96	0.91	0.94	3759
weighted avg	0.98	0.98	0.98	3759

Fig. 15. Scores for testing data

The f1 score for our best fit model after grid search is 0.98.

V. Conclusions

We were able to achieve a F1 score of 0.98 with a max depth of 11 by fitting Support Vector Classifier on our data set. The parameters for the best model were C=100, degree=5, gamma=0.001

From our observations, we can conclude the following about the significant features-

- **Mean of the integrated profile** - The mean of the integrated profile is lesser for pulsar stars than non-pulsars.
- **Excess kurtosis and skewness of the integrated profile** - For pulsars, the excess kurtosis and skewness is higher and more spread out as compared to non-pulsars.
- **Mean of the DM-SNR curve** - The mean of the integrated profile is higher and spread out for pulsar stars than non-pulsars.
- **Excess kurtosis and skewness of the DM-SNR** - For pulsars, the excess kurtosis and skewness is lower and less spread out as compared to non-pulsars.

Thus, as shown by both the qualitative and the quantitative analysis, the pulsar stars can be classified depending on its various features.

REFERENCES

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, An introduction to statistical learning: With applications in r. New York, NY: Springer, 2021.

[2] T. Hastie, J. Friedman, and R. Tisbshirani, The elements of Statistical Learning: Data Mining, Inference, and prediction. New York: Springer, 2017.

[3] Medium. 2021. Support Vector Machine — Introduction to Machine Learning Algorithms. [online] Available at: ¡https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47¿ [Accessed 26 November 2021].

[4] En.wikipedia.org. 2021. Support-vector machine - Wikipedia. [online] Available at: ¡https://en.wikipedia.org/wiki/Support-vector-machine¿ [Accessed 26 November 2021].

[5] scikit-learn. 2021. sklearn.svm.SVC. [online] Available at: ¡https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html¿ [Accessed 26 November 2021].

EE4708: Stock Market Analysis

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras
me18b114@smail.iitm.ac.in

***Abstract*—In this paper, we analyse the trends and gain some crucial insights from the given data set containing the stock prices data for 6 companies, 3 from the banking sectors - SBI, HDFC and ICICI Bank and 3 from the IT sector- Cognizant, HCL, and Infosys. We employ various data visualisation and analysis techniques on our data sets.**

I. Introduction

A stock is the small chunk of ownership in the company. The stock price of the company reflects the net evaluation of the company and also gives a little insight into its performance. These stocks are traded on exchanges and their prices are constantly changing due to their demand and supply in the market. If a stock is in high demand and low in supply i.e. more people want to buy it and fewer people are willing to sell it then the price for the stock will go up and similarly if the stock is in low demand and high on supply which means people more people are ready to sell it but fewer people are willing to buy it then its prices go down.

The sudden increase in the demand for the stock can be due to various reasons including positive news about the company or an announcement from the company. After a period of time when the demand for the stock vanishes its prices slowly creep down as the investor loses interest in it. These stock prices going up and down is an iterative process and repeated. This volatility of stock makes investors nervous while investing in a company. So to understand the risk associated with it there must be a proper analysis of stock before buying it. In this paper, we explore the stock data provided to us and derive crucial insights.

II. Financial and Mathematical tools

A. Simple Moving Average

In financial applications a simple moving average (SMA) is the unweighted mean of the previous k data-points. An example of a simple equally weighted running mean is the mean over the last k entries of a data-set containing n entries. Let those data-points be $p_1, p_2, , p_n$ This could be closing or opening prices of a stock. The mean over the last k data-points (days in this example) is denoted as SMA and calculated as:

$$SMA_k = \frac{p_{n-k+1} + p_{n-k+2} \cdots + p_n}{k} = \frac{1}{k} \sum_{i=n-k+1}^{n} p_i \quad (1)$$

B. Returns

In finance, return is a profit on an investment.[1] It comprises any change in value of the investment, and/or cash flows (or securities, or other investments) which the investor receives from that investment, such as interest payments, coupons, cash dividends, stock dividends or the payoff from a derivative or structured product. It may be measured either in absolute terms (e.g., dollars) or as a percentage of the amount invested.

Returns can be calculated as -

$$R = \frac{V_f - V_i}{V_i} \quad (2)$$

where:
V_f= final value, including dividends and interest
V_i = initial value

C. Volatility

Volatility is a statistical measure of the dispersion of returns for a given security or market index. In most cases, the higher the volatility, the riskier the security. Volatility is often measured as either the standard deviation or variance between returns from that same security or market index.

III. Data Analysis

A. Trends with time

Using Visualization tools, we plot the closing value of all the stocks given to us. From the plot, we can draw the following inferences -

- All the stocks plummeted down in March-April 2020 due the COVID-19 pandemic.
- Cognizant stocks remained constant throughout the pandemic.
- HDFC stocks dipped the most but displayed bullish behaviour and rose back up in the last fiscal year.
- Post-pandemic all stocks gave steady returns except Cognizant whose price remained almost the same.
- During the last year, Infosys overtook the HDFC stock from behind and ended with the highest price in the given stocks.

We can also plot the 7-day moving average to get a smoother curve. It is plotted below for both banking and IT sector.

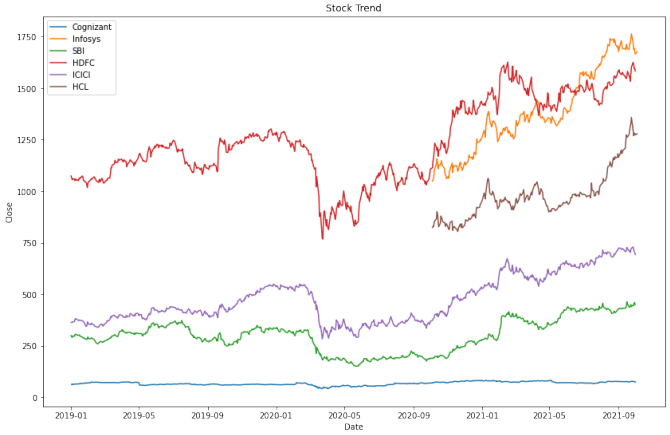

Fig. 1. Lineplot with time

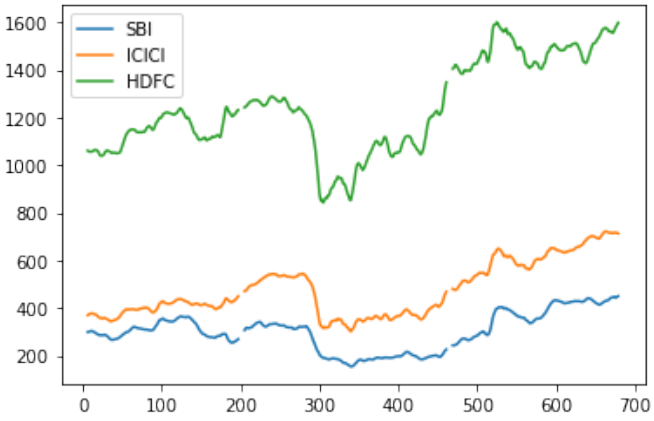

Fig. 2. Banking sector moving average

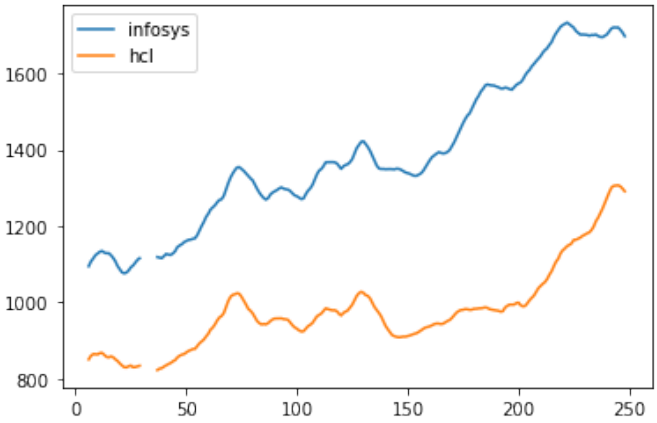

Fig. 3. IT sector moving average

B. *Stock Prices Comparison*

On comparison of just the mean values, we witness that mean value of Infosys stock is the highest while the Cognizant is least priced.

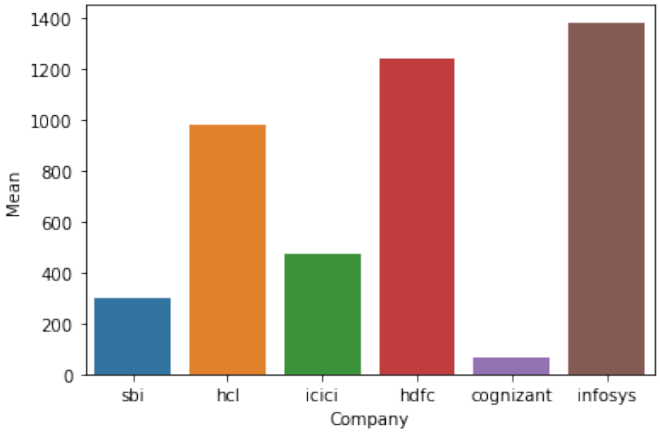

Fig. 4. Comparison of mean values of stocks

C. *Total stocks traded trend*

The total traded value is calculated by multiplying the Closing price and the total volume traded. We can see some spikes in the banking plots. There is a nearly infinite number of factors that can cause the stock market to move significantly in one direction or another, including economic data, geopolitical events, and market sentiment. For example we can see the biggest spike in the end of 2019 in the ICICI bank stock because the govt. announced reduction in the corporation tax rates.

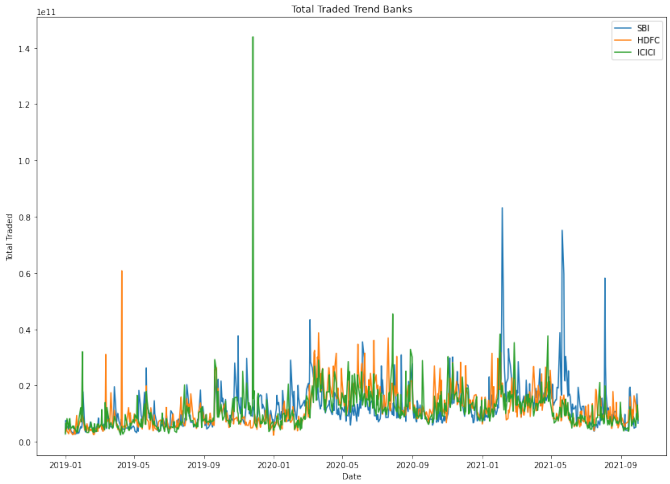

Fig. 5. Total Traded in banking stocks

D. *Scatter Plots and Correlation*

We have 3 banking and 3 IT companies in our data. We can hypothesize that the price rise and fall of a stock of the

companies from the same sector will be correlated. To verify this, we can construct correlation heatmaps and scatter plots for both the sectors.

1) ***Banks****:* We can witness that there is high correlation among all the banking sector's stocks. It can be understood as the change in government policies and ups and downs in economy affect all the banks.

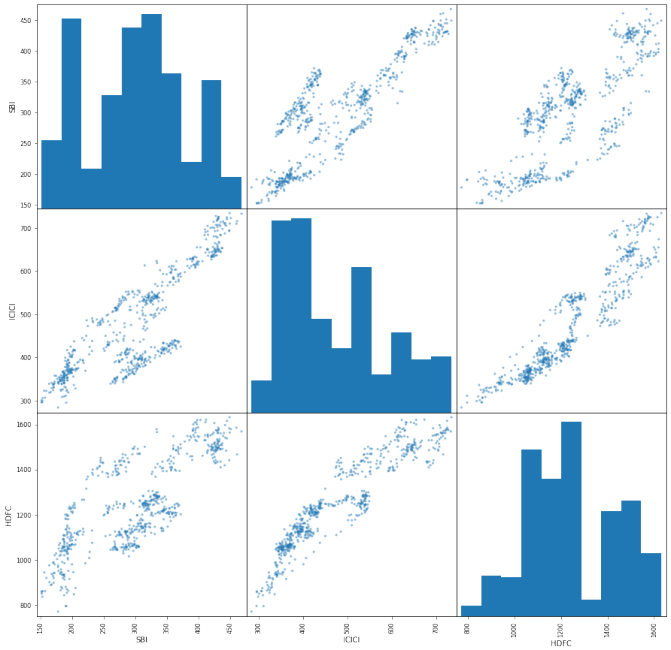

Fig. 6. Banking scatterplot

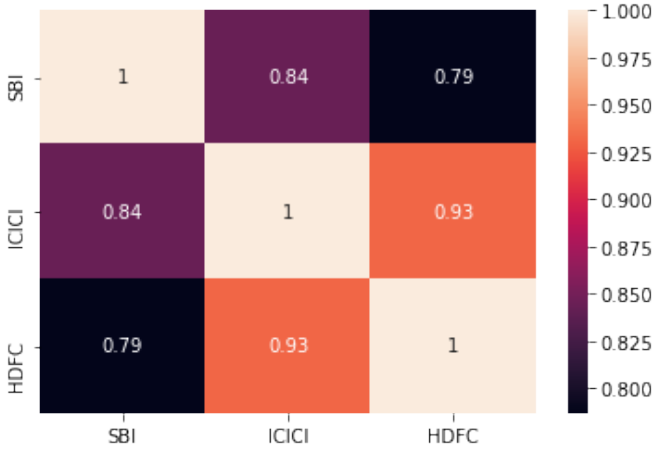

Fig. 7. Correlation Heatmap

2) ***IT****:* Similar to banking, Infosys and HCL have a high correlation coefficient of 0.88.

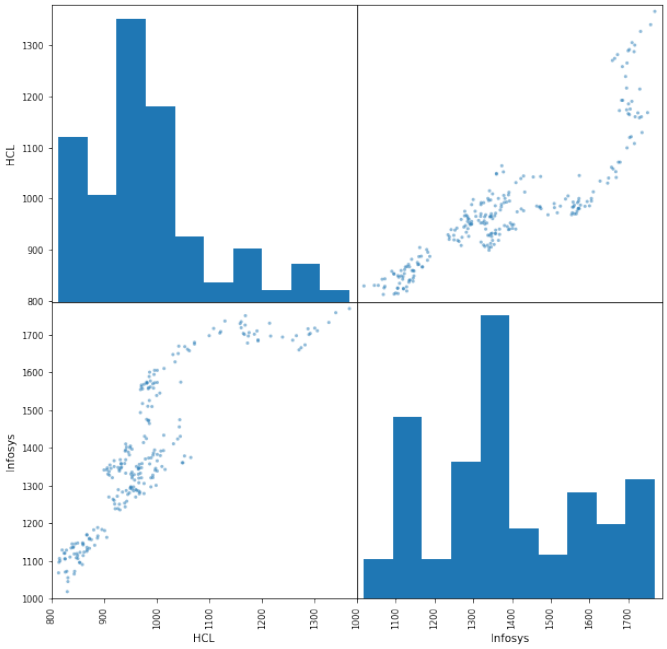

Fig. 8. IT scatterplot

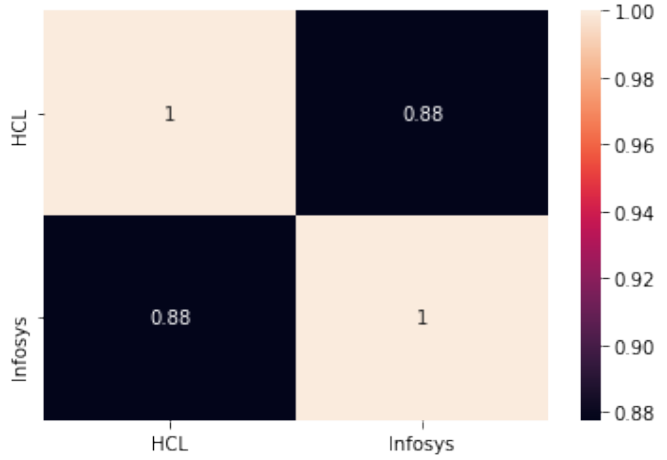

Fig. 9. Correlation Heatmap

E. ***Returns***

The returns of a stock can be calculated as the change in price of the stock over time, which may be represented in terms of price change or percentage change. We can find the returns of the given stocks by calculating the percentage change in the Closing value of the stocks. We plot the returns in the below given lineplot.

1) ***Volatility****:* It is hard to measure and quantify the returns through the line plot. So we can measure the volatility of the returns to get a better understanding. Volatility is a reflection of the degree to which price moves. A stock with a price that fluctuates wildly—hits new highs and lows or moves erratically—is considered highly volatile. A stock that

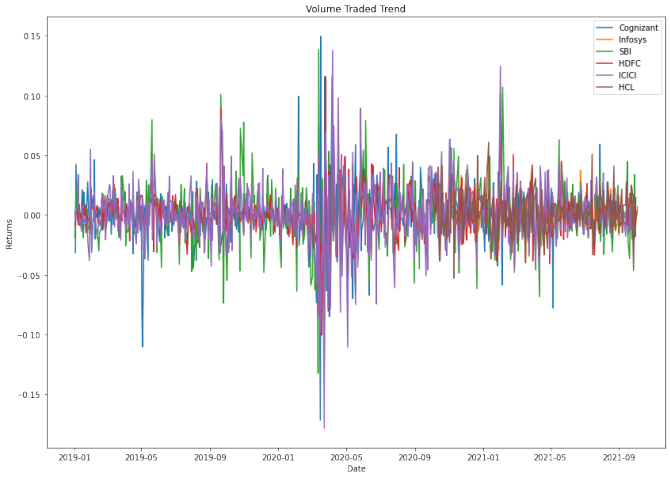

Fig. 10. Returns Lineplot

maintains a relatively stable price has low volatility.

We can plot the distribution of the returns to measure the volatility.

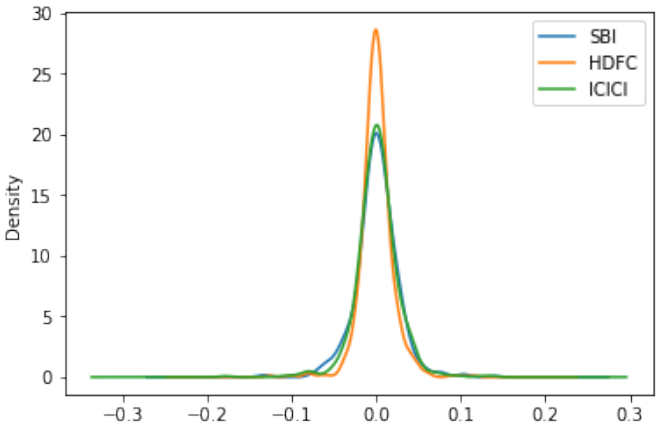

Fig. 11. Banks Volatility

In banking plot, we can see that SBI is less volatile than HDFC and ICICI, and it gives steady returns.

In IT sector, HCL is less volatile than the Infosys stocks.

*2) **Box Plots**:* Another way of visualising the returns is through box plots. Box plots display variation in samples of a statistical population without making any assumptions of the underlying statistical distribution. The box plots are given below.

From the box plots, the mean return of ICICI bank is highest, followed by SBI and HDFC, respectively.

From the IT sector box plot, the mean return of HCL and Infosys are almost the same, but HCL returns are more spread out.

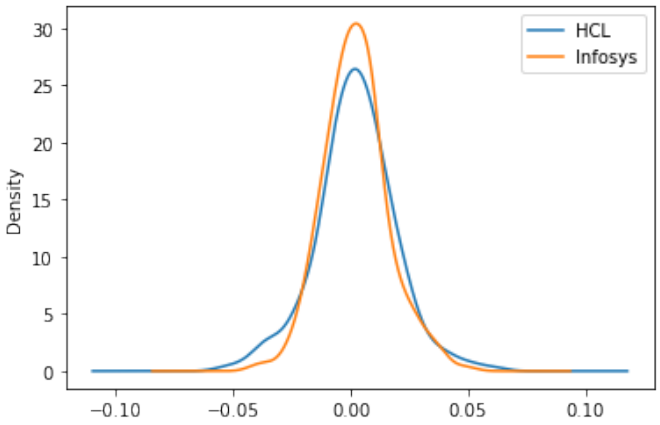

Fig. 12. IT Volatility

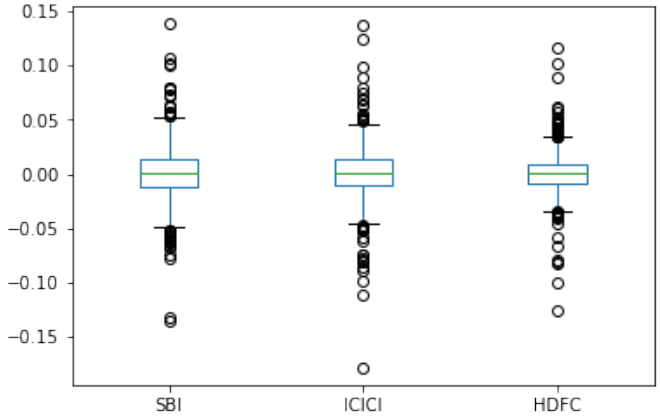

Fig. 13. Banking Boxplot

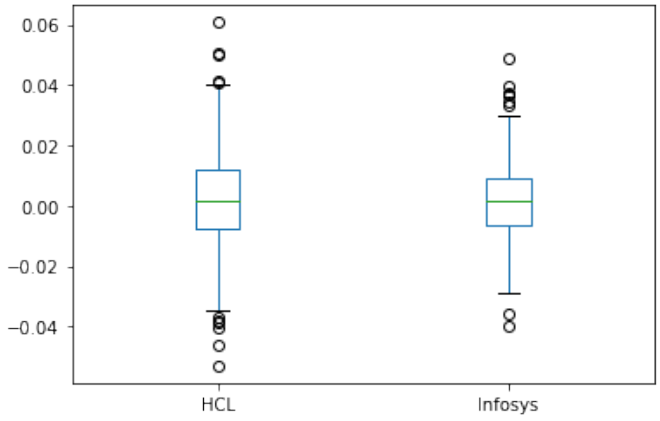

Fig. 14. IT Boxplot

F. *Cumulative Returns*

A cumulative return on an investment is the aggregate amount that the investment has gained or lost over time, independent of the amount of time involved. It is the total change in the investment price over a set time. We can plot the cumulative returns for both banking and IT sector.

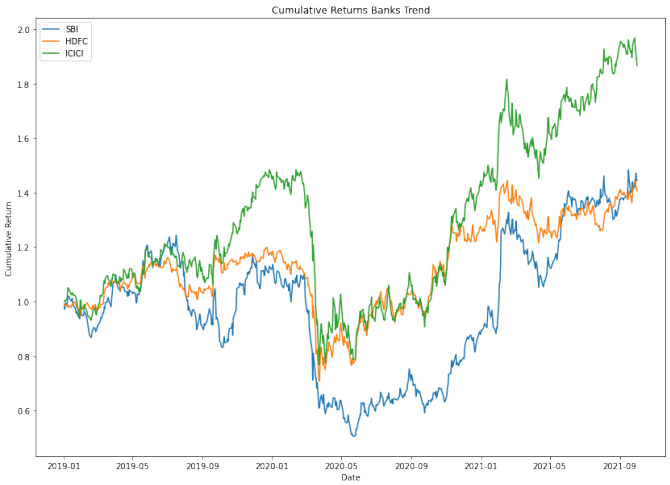

Fig. 15. Cumulative Returns Banking

*1) **Banks**:* As visible in the plot, all the returns fell in the peak pandemic months. SBI recovered from the pandemic the best as it overtook HDFC in cumulative returns by the end of November 2021, even though it was significantly lower than HDFC last year. HDFC showed low growth rates post pandemic. ICICI was the best performing stock in terms of cumulative returns.

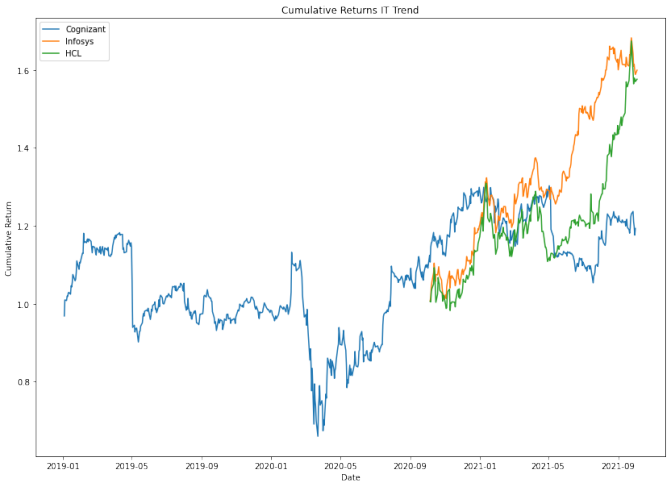

Fig. 16. Cumulative Returns IT

*2) **IT**:* Infosys and cognizant have the same level of cumulative returns at the end of 2021 and are significantly higher than that of Cognizant, which showed really slow growth. Due to the slow growth of Cognizant, HCL and Infosys were able to overtake it in cumulative growth.

G. *USD-INR Exchange Rate*

We plot the exchange rate for the 3 years for which data is available.

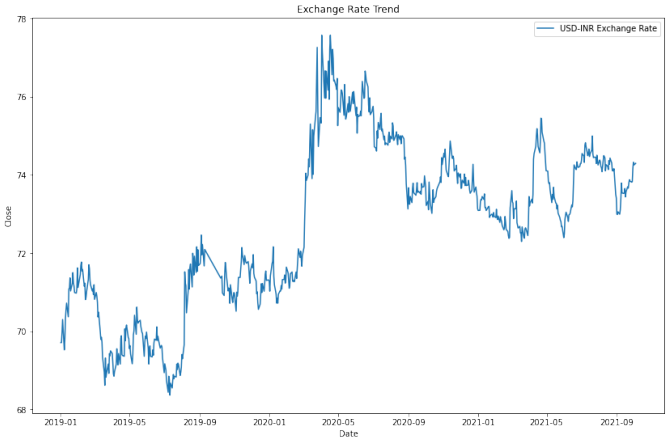

Fig. 17. USD INR exchange rate

The 7 day moving average can also be plotted to visualize a smoother trend,

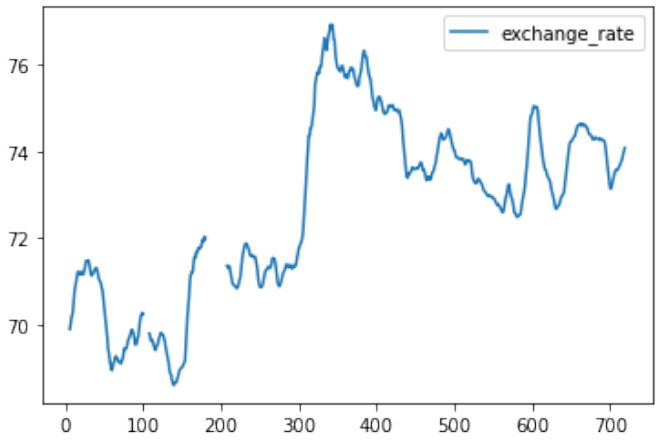

Fig. 18. USD INR exchange rate moving average

The peaks in the rates in May 2020 and May 2021 correspond to the the COVID-19 waves in India, which resulted in derailing the economy of India by hitting small and midsize enterprises particularly hard, besides delaying the recovery in banks asset quality and resulting in job losses and business closures.

IV. Conclusions

Through our analysis of the stocks using various techniques, we can conclude that -

- All stocks except Cognizant took a big dip in the pandemic and then regained momentum by the end of year 2020, with HDFC showing both the steepest dip and the highest rise.
- Out of all the stocks in the data set, Infosys is the highest priced and Cognizant is the lowest priced in terms of mean price.
- We witnessed several spikes in the total amount traded in the given stocks due to a variety of reasons over the month. The highest spike came in the ICICI stock because of a change in government policy at the time.
- In both banking and IT sectors, the rise and fall of stock prices are highly correlated, as displayed in the scatter plots.
- Through box plots and scatter plots, we were able to determine the returns and the associated volatility of all the stocks. We concluded that SBI and HCL stocks were the least volatile in terms of returns, while ICICI bank had the highest mean returns.
- In terms of cumulative returns over the given time period, ICICI and Infosys stocks were the best performing in their respective sectors while cognizant stocks gave constant but low returns over time.
- The peaks in the USD-INR exchange rate corresponds to the COVID-19 waves in Indian economy.

REFERENCES

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, An introduction to statistical learning: With applications in r. New York, NY: Springer, 2021.

[2] T. Hastie, J. Friedman, and R. Tisbshirani, The elements of Statistical Learning: Data Mining, Inference, and prediction. New York: Springer, 2017.

[3] A. Hayes, "What Is a Stock?," Investopedia, 01-Dec-2021. [Online]. Available: https://www.investopedia.com/terms/s/stock.asp. [Accessed: 03-Dec-2021].

[4] Lonnieqin, "Stock Market Analysis Prediction using LSTM," Kaggle, 25-Mar-2021. [Online]. Available: https://www.kaggle.com/lonnieqin/stock-market-analysis-prediction-using-lstm. [Accessed: 03-Dec-2021].

[5] "Portfolio cumulative returns," Python. [Online]. Available: https://campus.datacamp.com/courses/introduction-to-portfolio-analysis-in-python/introduction-to-portfolio-analysis?ex=7. [Accessed: 03-Dec-2021].