# Project Report

# Clickstream data for E-shop online shopping

Likitha Jagithyala

Jagithyala.l@northeastern.edu

College of Engineering Fall 2022

Northeastern University, Seattle.

**Percentage of Effort Contributed by Student:** 100%

**Signature of Student**: Likitha Jagithyala

**Submission Data**: 17th December 2022

# Table of Contents

# 1. Abstract

Online shopping is a platform with utmost transparency among the buyers and sellers. This omits the need of middlemen and biased pricing technique which is most seen in offline market setup. In such a unclouded scenario we have a lot of information to predict the price of any product based on the previous clickstream pattern. It is often observed that the online shopping websites try to reduce price of the product if it has more views and less purchases assuming that the customer is willing to buy the product, however, is hesitant and price could be one of the reasons. This is one of the most common patterns observed to change the price. However, there could be multiple patterns affecting the pricing decision which is explored in this project contributing for better prediction and classification.

IE-clothing [ www.ie-clothing.com ] is an online shopping site offering clothing for pregnant women. Back in 2008 , clickstream data of this website was collected for certain period where each row is a session of customer's pattern of moving through the website with the intention to understand the navigation patterns of customer. These sessions are basic information of customer clicking at a specific page or product on certain click along with other information related to product like the position of the image , category and so on which also influence the buyer potentiality of the customer.

# 2. Problem Statement

Often customer patterns help in comprehending change in business strategies especially with respect to price. This Project will penetrate through the data to fetch pricing changes of multiple session and classify if the price is over or below the average using multiple classification and regression algorithms wherein the attribute of each click is considered to forecast price at that specific click as well to classify if a certain price is set above or below average.

# 3. Data

## 3.1 Data Source

The dataset is collection of clickstream data on “ie-clothing” website and uploaded in UC Irvine Machine Learning Repository. This dataset is under a Creative Commons Attribution 4.0 International (CC BY 4.0) license.

The following link is used to download the data.
https://archive-beta.ics.uci.edu/dataset/553/clickstream+data+for+online+shopping.

clickstream data for online shopping. (2019). UCI Machine Learning Repository.

£apczyÒski M., Bia≥owπs S. (2013) Discovering Patterns of Users' Behaviour in an E-shop - Comparison of Consumer Buying Behaviours in Poland and Other European Countries, ìStudia Ekonomiczneî, nr 151, ìLa sociÈtÈ de l'information : perspective europÈenne et globale : les usages et les risques d'Internet pour les citoyens et les consommateursî, p. 144-153.

## 3.2 Data Description

The data consists of 165474 instances i.e., rows and 14 attributes/variables i.e., columns which includes multiple information of each session(single click session) back in 2008. Each column description is as below:

| **Variable Name** | **Descriptions** |
|---|---|
| year | Year in which the data was recorded i.e., 2008 |
| month | Month in which the instance was recorded. April(4) to August(8) |
| day | Day number of the respective month of the instance. |
| order | sequence or order of clicks in a particular session |
| country | Country of origin of the IP address in terms of code |
| session ID | ID given to a session or an instance |
| page 1 (main category) | Includes main product categories i.e., trousers, skirts, blouses, & sale |
| page 2 (clothing model) | information about each product code (217 products) |

| | |
|---|---|
| colour | color of product |
| location | Location of the photo on the page, where the screen is divided into 6 parts i.e., top left, bottom right, top middle, bottom left, top right, bottom middle |
| model photography | Model en face or profile |
| price | Cost/Price of the product in US dollars |
| price 2 | Indicates if the product is overpriced when compared to average price |
| page | Page number of the webpage on the website having 5 pages. |

*Table 3.2(a)*

# 3.3 Data Statistics and Explanatory Data Analysis

## 3.3.1 Sample data:

| | year | month | day | order | country | session ID | page 1 (main category) | page 2 (clothing model) | colour | location | model photography | price | price 2 | page |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2008 | 4 | 1 | 1 | 29 | 1 | 1 | A13 | 1 | 5 | 1 | 28 | 2 | 1 |
| 1 | 2008 | 4 | 1 | 2 | 29 | 1 | 1 | A16 | 1 | 6 | 1 | 33 | 2 | 1 |
| 2 | 2008 | 4 | 1 | 3 | 29 | 1 | 2 | B4 | 10 | 2 | 1 | 52 | 1 | 1 |

*Figure 3.3.1(a)*

## 3.3.2 Data Information:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 165474 entries, 0 to 165473
Data columns (total 14 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   year                      165474 non-null  int64
 1   month                     165474 non-null  int64
 2   day                       165474 non-null  int64
 3   order                     165474 non-null  int64
 4   country                   165474 non-null  int64
 5   session ID                165474 non-null  int64
 6   page 1 (main category)    165474 non-null  int64
 7   page 2 (clothing model)   165474 non-null  object
 8   colour                    165474 non-null  int64
 9   location                  165474 non-null  int64
 10  model photography         165474 non-null  int64
 11  price                     165474 non-null  int64
 12  price 2                   165474 non-null  int64
 13  page                      165474 non-null  int64
dtypes: int64(13), object(1)
memory usage: 17.7+ MB
```

*Figure 3.3.2(a)*

The data seems to be very clean with no null values throughout. All the datatypes are int64 except for 'page 2 (clothing model)' which is an object datatype.

### 3.3.2 Statistical Information:

Statistical Information is calculated on numerical and object data which includes count, mean(basic average), median(middle of sorted list), std(standard deviation, a measure of data dispersion), min(minimum value), 25%, 50%, 75% (25, 50 and 75 percentiles of each column out of 100%), max(maximum value) of each column throughout all the rows

| # | year | month | day | order | country | session ID | page 1 |
|---|---|---|---|---|---|---|---|
| **count** | 165474 | 165474 | 165474 | 165474 | 165474 | 165474 | 165474 |
| **mean** | 2008 | 5.586 | 14.525 | 9.817 | 26.953 | 12058.417 | 2.401 |
| **std** | 0 | 1.328 | 8.830 | 13.478 | 7.151 | 7008.419 | 1.144 |
| **min** | 2008 | 4 | 1 | 1 | 1 | 1 | 1 |
| **25%** | 2008 | 4 | 7 | 2 | 29 | 5931 | 1 |
| **50%** | 2008 | 5 | 14 | 6 | 29 | 11967.5 | 2 |
| **75%** | 2008 | 7 | 22 | 12 | 29 | 18219 | 3 |
| **max** | 2008 | 8 | 31 | 195 | 47 | 24026 | 4 |

*Table 3.3.2(a)*

| # | colour | location | Model photography | price | price 2 | page |
|---|---|---|---|---|---|---|
| **count** | 165474 | 165474 | 165474 | 165474 | 165474 | 165474 |
| **mean** | 6.228 | 3.258 | 1.260 | 43.803 | 1.488 | 1.710 |
| **std** | 4.236 | 1.713 | 0.439 | 12.548 | 0.500 | 0.982 |
| **min** | 1 | 1 | 1 | 18 | 1 | 1 |
| **25%** | 3 | 2 | 1 | 33 | 1 | 1 |
| **50%** | 4 | 3 | 1 | 43 | 1 | 1 |
| **75%** | 9 | 5 | 2 | 52 | 2 | 2 |
| **max** | 14 | 6 | 2 | 82 | 2 | 5 |

*Table 3.3.2(a) Contd.*

As the dataset is clean and mostly label encoded, the datatype for all the columns is int and object. Therefore, the statistical information for each column is possible. Based on the statistical information the effective columns that have fluctuating patterns are day, order, country, session ID, colour, location, price.

### 3.3.3 Density Curves:

From the fluctuating columns country , colour and price columns are more effective and important columns to look at for better decision making. Below are the density curves for the same

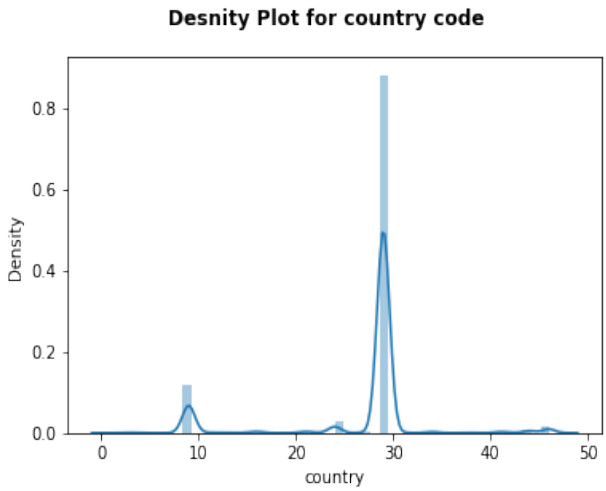


*Figure 3.3.3(a)*

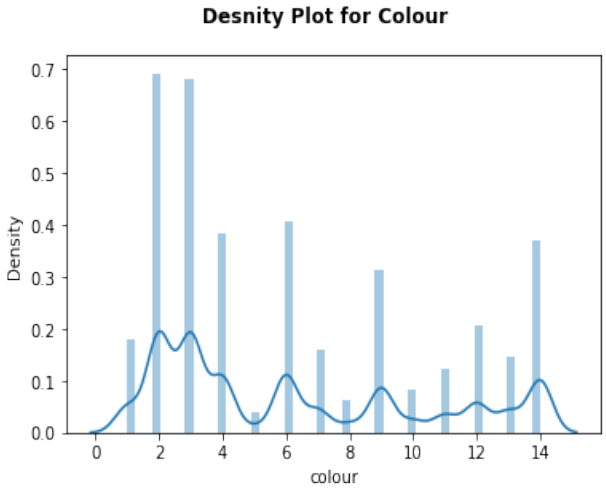


*Figure 3.3.3(b)*

Most of the contribution of click sessions are observed in country code 29 which represents Poland. This refers to having a biased data mostly belonging to Poland.
The sessions are higher for color black(2) and blue(3) followed by few more fluctuations in color

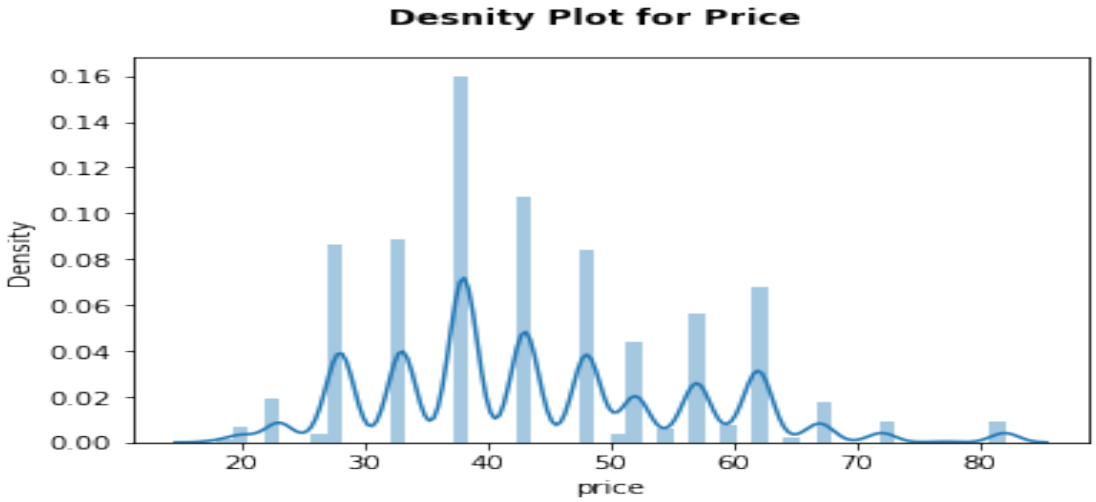


*Figure 3.3.3(c)*

Price density curve is more around ~38 followed by similar distrubtin at range of other prices too. The website has equal session/clicks for varity of prices but focus can be done on a few.

### 3.3.4 Country Analysis:

The Density curves for country has some major fluctuation. Top countries contributing to the count of session will help in taking different decision and understand which country the website is quite popular and where there is scope to improve.


*Figure 3.3.4(a)*

Poland has the highest sessions which also say that the website is quite popular in this country, and it is matching the needs of the Poland citizens. Czech Republic is second most session contributing country however the gap between these two countries is quite big. The fluctuations in the session contribution could be each country having different competitors. There is a high possibility that IE-Shop could be one of the very few sites in Poland whereas the same website is kind of lost in highly developed countries like United States or United Kingdom. Population also plays a key role here along with the generic culture that is followed in the country. Website should try to look at strategies that are working well in Poland or in Czech Republic to use them in other countries to get such high visibility.

### 3.2.5 Click Analysis:

Click analysis gives the average count of click done in placing one order by all the customers in a specific country. Average clicks information is helpful to understand if a customer is taking too many clicks before placing an order or not. If the click count is higher for one order placed, it could be because the customer has been going through varied collection present on the website and is confused to pick one which is very common in clothing website and another strong reason could be the procedure to place an order. If there are multiple complication steps to

finish placing order, it gets irritating and high chance of customer losing interest and website can potential loose orders due to this frustration.

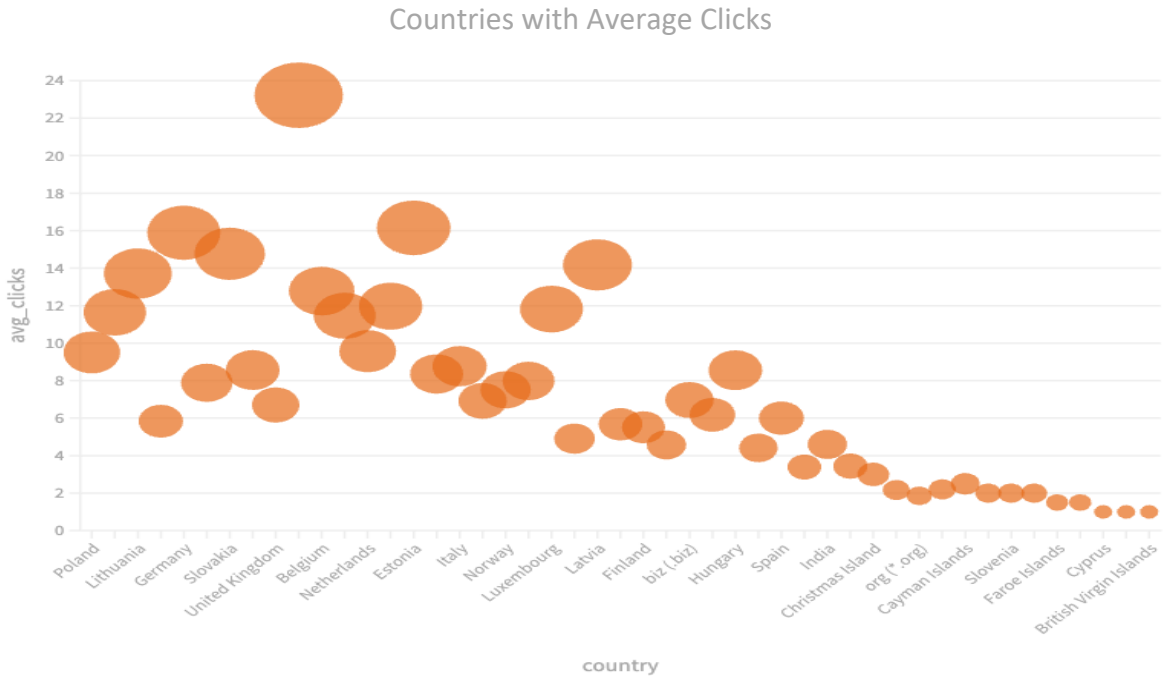


*Figure 3.3.5(a)*

Romania has the highest average clicks of 23 followed by Estonia with 16 and Germany with 15. These countries are not in Top10 therefore the website is not used by a lot of people. Few of the reasons as mentioned earlier could be frustrating procedure to place order and confusing website layout. This could also be the major cause of less popularity of the website in these countries. Poland the country with maximum users has an average click of 9. The website in other countries where it is not performing should try and make the website and procedure of placing order conveniently that a session can be ended within 10 clicks. Countries with less average clicks like British Virgin Island , Cyprus and more should try to enhance the website's variety as users might look at few pages and leave the site because of poor selection

Further to understand the clicks and the reason behind having a lot of clicks for each session helps us to look at the pattern that a customer is following before placing an order or leaving the website without any order. These patterns aid in better decisions making related to enhancing the clothing variety , changing the website layout according to the citizens of a specific country and look at scenarios

where customer is leaving the website without placing the order due to frustrating procedure.

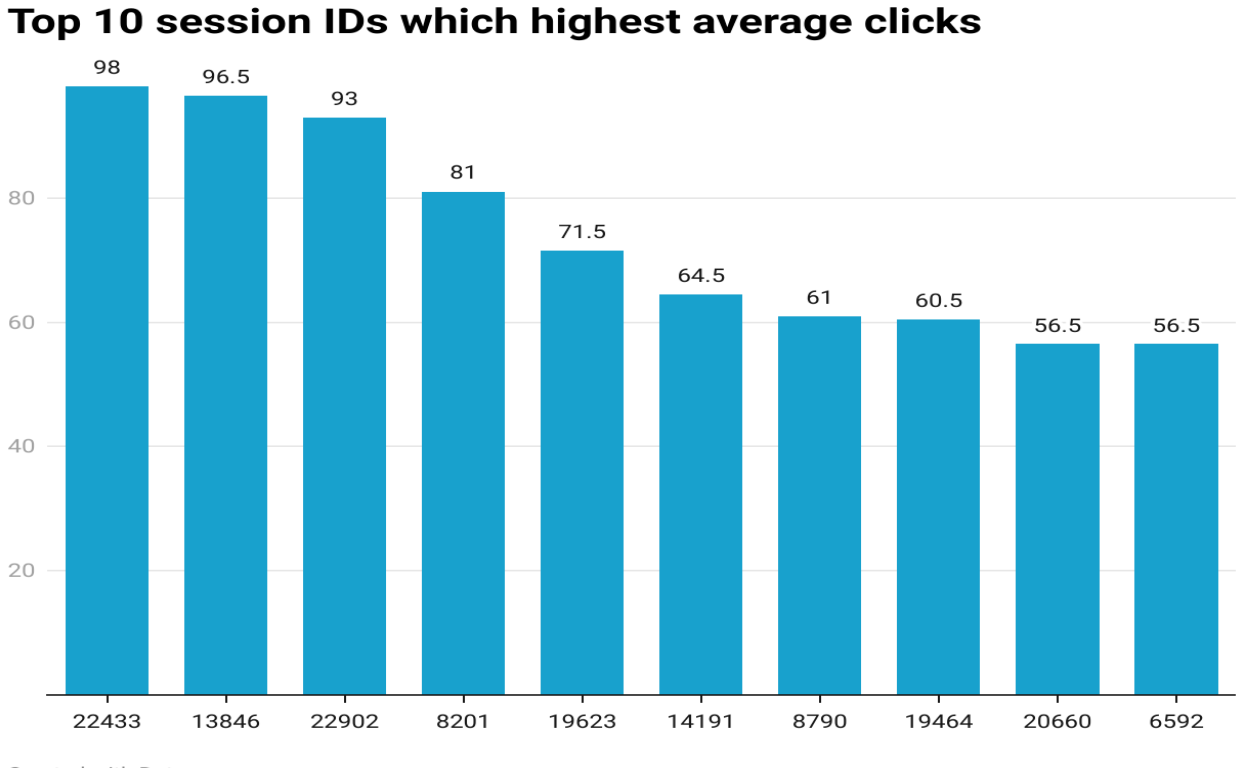


*Figure 3.3.5(b)*

These Top 10 session ID will guide us on pursing customer pattern and the website owner should try and improve the drawbacks found while comparing. The highest average clicks throughout different countries are 23 and for most popular country Poland has average clicks of 9. However, for these above-mentioned session IDs the highest clicks 98. There are three specific session ID ranging above 90. Deep observing these session ID will guide in enchaining customer experience while indulging with IE-shop website

### 3.3.6 Correlation:

The analysis can go on as much as we wish to explore however for our intention of the project to classify session where price is above and below average and to predict price range based on previous session. To move further identifying the relationship against each column is very helpful. A correlation matrix shows the correlation score between two columns. Highly positive correlation says that columns increase in value simultaneously and decrease the same way. Highly

negative correlation says that if one column's value increases the other columns tends to drop.

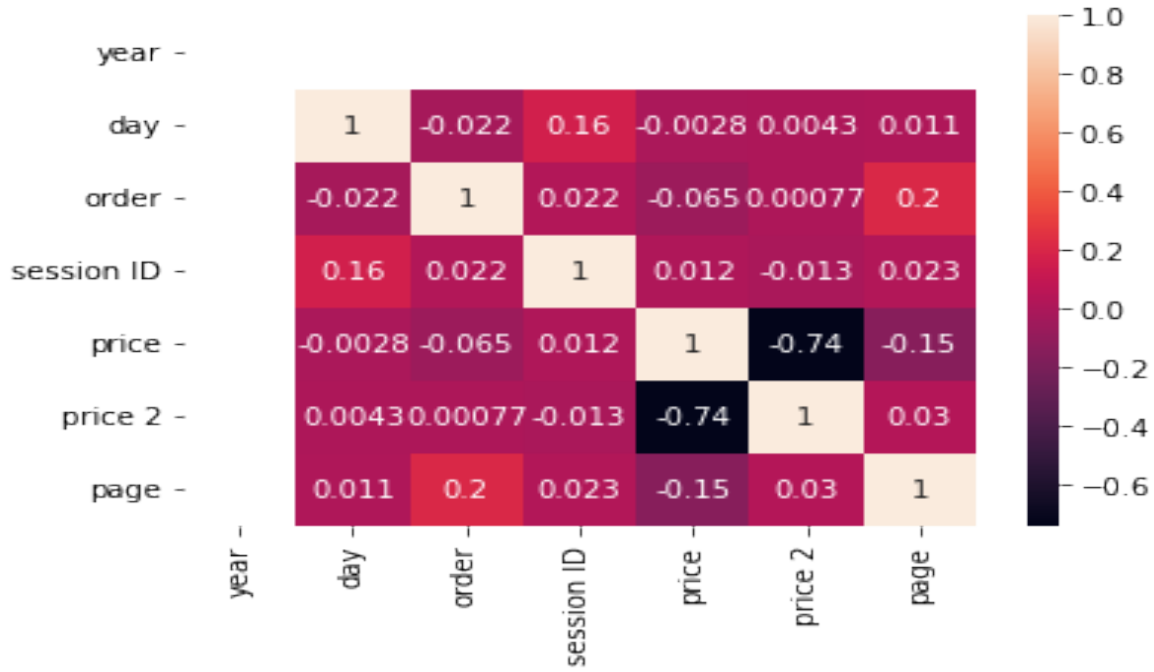


*Figure 3.3.6(a)*

The above is a correlation matrix clearly gives the degree of correlation against each of the comparable columns. Year column is least correlated as the year of data collected is one specific year and the column remains same throughout. The most correlated columns are price and price 2 with -0.74 followed by order against page with correlation of 0.2. Order against page correlation tells us that the sequence in which a customer accessed each page and how are they related. Session ID is correlated with date as a customer will usually continue one session is short period of time

### 3.3.7 Price Vs Price 2:

Price 2 is a condition where the price of a session is higher or lower than average price. Price against price 2 is an understanding whether a particular product is priced higher or lower than the average price and exploring this relationship will be quite helpful for our analysis as these two columns is the basis for our modelling

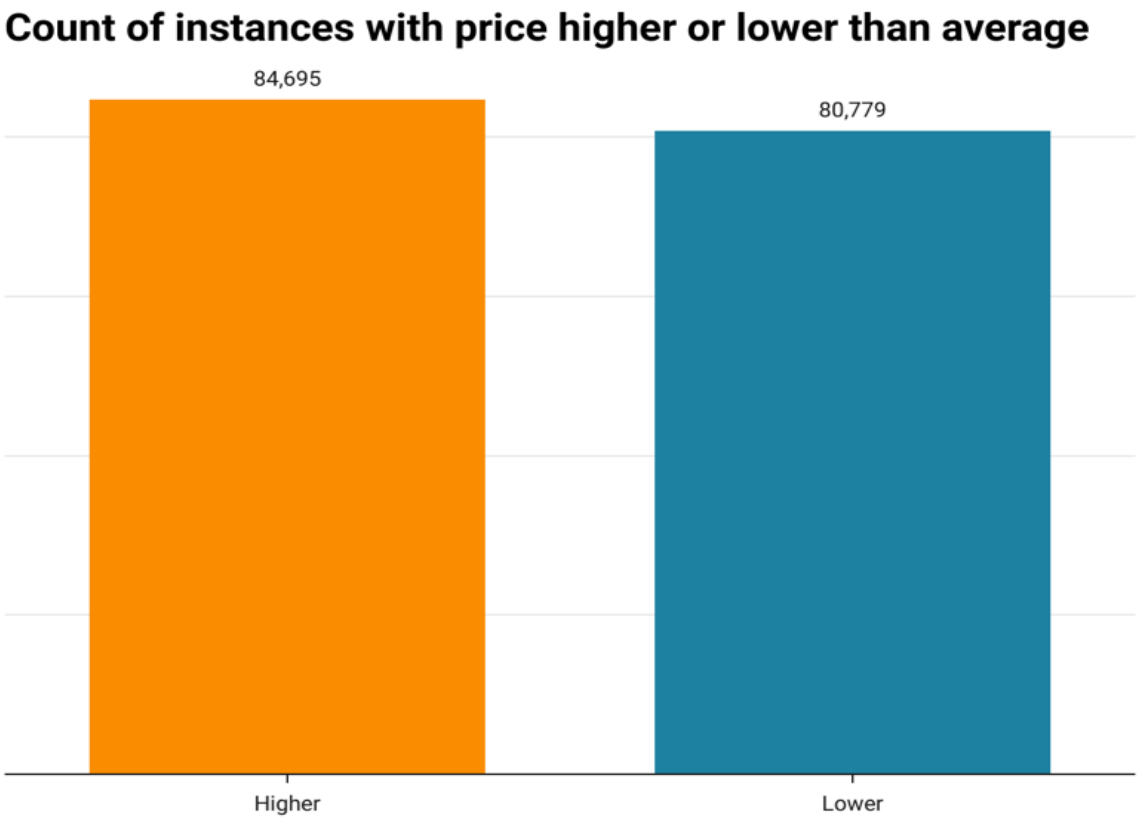


*Figure 3.3.7(a)*

So, from the plot we understand that the instances where prices are higher than average with respect to prices lower than average are close to each other with a difference of ~4K.
Thus , helping us to establish a proper relationship with X(major independent variable) price and Y(dependent variable) as price 2. The price2 data for both the classes is approximately close to each other therefore the chance of model being biased towards one specific class just because of disparity in the size of class will not occur.

## 3.4 Data Preprocessing

The data is pretty much clean and basic preprocessing that the data requires is to make it modelling ready. The data needs label encoding to make sure that all the columns are encoding well. Few of the models that we implement also need the data to be scaled and finally split the data in train and test dataset with proper definition of X(independent features) and Y(target variable). Preprocessing for

both classification and regression model is slightly different and X and Y in both the situation is different.

## Classification of Price 2

**Preprocessing for Classification model**

```
#Defining dependent and independent variables
X = data.drop('price 2', axis=1)

Y = data["price 2"]
#storing column names of X in names
names = np.array(X.columns)

#Encoding the columns
def label_transform(x):
    le = LabelEncoder()
    Encoded_le = le.fit_transform(x)
    return Encoded_le

for x in names:
    X[x] = label_transform(X[x])

#Scaling data
SCX = StandardScaler().fit_transform(X)

#Splitting data into train and test sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=41)

#Splitting scaled data into train and test sets
SCX_train, SCX_test, SCY_train, SCY_test = train_test_split(SCX, Y, test_size=0.2, random_state=41)
```

*Figure 3.4(a)*

The preprocessing for classification model includes defining X as all the columns except that of “price 2” column which is defined as Y. Columns names will be stored in an array “name” which will be used in decision tree model as well as in label encoding where all the columns present in the X are encoded. Scaled data of X shall be stored in new SCX as these shall be used to prepare train and test datasets of X and Y along with scaled X and Y. This should be done because few model required scaled X values and few required usual X values. The train and test split ratio are 80:20

## Regression on Price

### Preprocessing for Regression model

```
#Defining dependent and independent variables
RX = data.drop('price', axis=1)

RY = data["price"]
#storing column names of X in names
names = np.array(RX.columns)

#Encoding the columns
def label_transform(x):
    le = LabelEncoder()
    Encoded_le = le.fit_transform(x)
    return Encoded_le

for x in names:
    RX[x] = label_transform(RX[x])

#Scaling data
RSCX = StandardScaler().fit_transform(RX)

#Splitting data into train and test sets
RX_train, RX_test, RY_train, RY_test = train_test_split(RX, RY, test_size=0.2, random_state=41)

#Splitting scaled data into train and test sets
RSCX_train, RSCX_test, RSCY_train, RSCY_test = train_test_split(RSCX, RY, test_size=0.2, random_state=41)
```

*Figure 3.4(b)*

A similar preprocessing is required for creating train and test dataset for regression model too. The major difference in this preprocessing is defining X and Y where X shall be all columns except of price column which is Y. X shall then be label encoded and scaled for the same reasons. Train and test ratio of 80:20 shall be created with both X and scaled X values.

# 4. Modelling

There are two intents of this project where one is to classify price2 if it is higher or lower than average price and the other is to predict price of each product. All the models are built from scratch without the help of any libraries except NumPy. The purpose of modelling is to establish multiple patterns using different techniques that are discussed below.
To measure the model performance, the below performance metrics are used.
*I. Classification performance metrics:*

- *Accuracy Score.* It is a simple calculated of comparing actual vs predicted where in the actual matched with predicted is divided by total predicted.

$$Accuracy\ score = Correct\ Predictions / Total\ Predictions$$

- *Confusion Matrix*. It is matrix with class of predictions compared with classes of actual. In a grid format where quadrant of X and Y is quadrant of actual and predicted for that specific class.

*II. Regression performance metrics:*

- *RSquared [$R^2$]*. It is the coefficient of determination tells how well model fits the data also the goodness of the fit. It is calculated as Residual sum of squares(RSS) by Total sum of squares(TSS) and then subtracted by 1.

$$R^2 = 1 - (RSS/TSS)$$

- *Root Mean Squared Error.* It is the average length from each actual to its predicted value. It is calculated as sum of total square of predicted value at i(Pi) subtracted by actual value at i(Ai) , total divided by 2 and square root of remaining is considered

$$RMSE = \sqrt{\Sigma(Pi - Ai)2 / n}$$

A custom function is considered for calculated these metrices.

## 4.1 Classification of instances where price is high/low than average price[Y=Price 2]

The independent features considered for this modelling are all other models except of price 2 columns the efforts are to classify instances into two classes with one class having price more than the average and another class with lower than the average.

### 4.1.1 Logistic Regression Classification:

- ***Definition:***

A simple custom logistic regression classifier is a linear classifier using a linear function called as logit. It is less time consuming and convenient. This model either predicts in 0s or 1s therefore creating a sigmoid function or S-shaped curve of X and that is the reason it is most suitable for binary classification.

- ***Methodology:***

Logits are calculated as f(x) = $b_0 + b_1x_1 + \cdots + b_ix_i$ where $b_0, b_1, \dots, b_i$ are weights to variables X. With the help of logit, we calculate the logistic regression function i.e., a sigmoid function S(x) = $\frac{1}{1+e^{-f(x)}}$ giving predicted output of 1s and 1-S(x) the predicted output of 0s. To get the appropriate weights, we utilize maximum likelihood estimation to maximize log-likelihood function L(θ)= $\sum_{j=1}^{n} \ln f_j(y_j \mid \theta)$. Making use of these we fit the model and predict the results.

- ***Performance:***

The train and test data both are ran through the model to fit the performance of the same is as below.

*Accuracy score of Logistic Regression Classifier*

| Metrics/dataset | Train | Test |
|---|---|---|
| Accuracy Score | 51.07% | 51.64% |

*Table 4.1.1(a)*

There is a underfitting issue in the model as both the train and test accuracy is way low than accepted values. To further understand below is the confusion matrix of the predictions against the true/ actual values.

*Confusion Matrix of Logistic Regression Classifier*

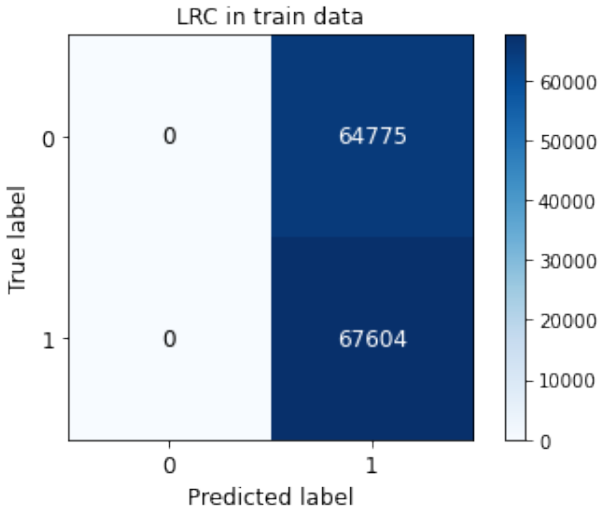


*Figure 4.1.1(a)*

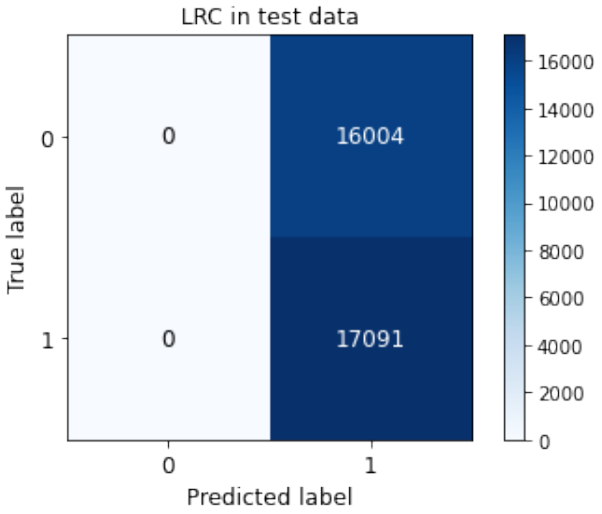


*Figure 4.1.1(b)*

In the above, confusion matrix is divided into four grids. The top left grid is for those count of instances where the actual value of Y is 0 and the model accurately predicted as 0. Similarly the bottom right is accurate predictions for Y value as 1. The Top right grid is for those count of instances where the actual value of Y is 0 however the model incorrectly predicted as 1 also known as false positives. Similarly the bottom left is for incorrection predictions of 1 to be 0 also known as false negatives. This model has not predicted a single instance of 0 or 'No' scenarios in both train and test and that is the major reason for low accuracy.

- ***Important features.***

Most of the time due to high count of columns in X the model tends to get diverted. Often we do not need that high count of X columns as Y is usually dependents of few columns and excluding irrelevant columns will help the model to understand the patterns better. In order to identify which feature has contributed the most we look at the co-efficient/ weights of the feature and below are the important of each features

| feature | importance |
|---|---|
| page 2 (clothing model) | 2.587970 |
| session ID | 2.502849 |
| price | 1.276730 |
| colour | 0.138294 |
| page 1 (main category) | 0.066499 |
| month | 0.004877 |
| year | 0.000000 |
| model photography | -0.017947 |
| page | -0.020902 |
| order | -0.070037 |
| location | -0.076205 |
| day | -0.077894 |
| country | -0.095273 |

*Figure 4.1.1(b)*

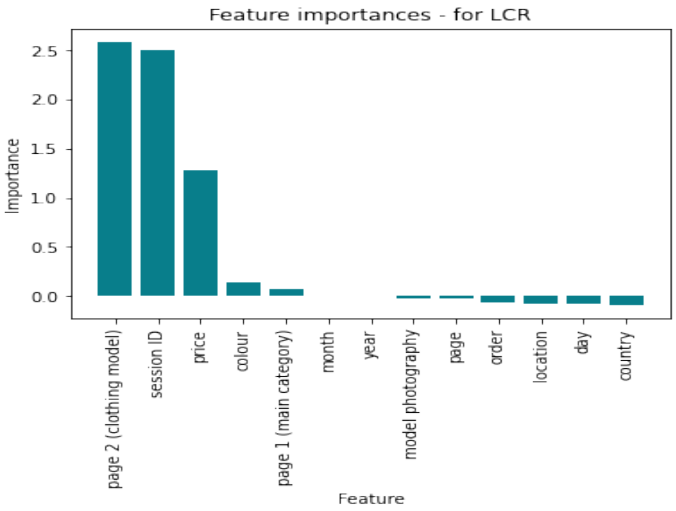


*Figure 4.1.1(c)*

Based on the above Page 2 , Session Id and price have contributed the most for prediction of Y. Session ID is actually a row number therefore the contributed for a dummy features is high.

- ***Recommendations:***

In order to improve model accuracy , noise reduction is encouraged where PCA can be done to reduce the X's dimension to top 3 principal components. Tuning the hyperparmeters like learning rate and epochs will be helpful for model to carefully look for patterns , thus reducing the underfitting issue.Along with this data augumentation can be done in order to have increased patterns.Excluding few dummy data as session ID will also help.

### 4.1.2 Decision Tree:

- ***Definition***

Decision Tree modeling is a method of representing the path of predictions by branching it out in a tree format which has a root node , multiple decision nodes and finally leaf nodes which gives our prediction. Here all X are split basis certain rules to classify and is best suited for classification. This model is one of the popular ml models as it is easy to interpret and does not impact with Y being binary or multi class classification

- ***Methodology***

To construct a decision tree, it is important to identify impurities present in each feature. Impurity is measured basis information required or needed to decide also known as information gain and calculated with the help of Entropy and Gini Index. Entropy, E = $\sum_{i=1}^{n} p_i * \log(p_i)$ is the amount of information needed to accurately identify the homogeneity of some sample. If a sample has all same elements, then E=0 and keeps increasing with decrease in homogeneity till E<=1. Gini index, GI = $1 - \sum_{i=1}^{n} p_i^2$, is the measure of inequality and ranges from 0 to 1 where 1 is the maximum inequality in the sample. With the help of these Information gain is calculated and the feature with highest information gain is given more priority to make decisions.

- ***Performance***

Based on the custom mode for Decision tree classifier the below decision tree is built

```
DTC.print_tree()
```

```
|- price <= 7 :Information Gain 0.2825
 left:|- page 1 (main category) <= 2 :Information Gain 0.0884
  left:|- price <= 6 :Information Gain 0.0179
    left:0
    right:|- page 1 (main category) <= 1 :Information Gain 0.3455
        left:0
        right:1
  right:|- price <= 5 :Information Gain 0.4999
    left:0
    right:1
 right:|- colour <= 0 :Information Gain 0.009
  left:|- page 1 (main category) <= 0 :Information Gain 0.4703
    left:1
    right:0
  right:|- location <= 4 :Information Gain 0.0014
    left:1
    right:|- colour <= 1 :Information Gain 0.0338
        left:1
        right:1
```

The tree has the root node, decision node and leaf nodes with information gain values and can be interpreted as price with IG of 0.2825 is the root node and for instances where price(root node) is less than or equal to 7, the tree expands to left with decision nodes as page 1 and price. If page 1 is less than or equal to 2, the tree checks if price is less than or equal to 6 and predicts price2 as No/0, else it moves on checking page 1 is less or equal to 1 and predicts No/0 else Yes/1. Else if page 1

is greater than 2 , and price is less than or equal to 5 No/0 is predicted , if price is greater than 5 Yes/1 is predicted. Similarly, if the root node is greater than 7 , the tree expands to right side to check color is less than or equal to 0 or not. If yes, along with page1 less or equal to 0 the predication is 1 else 0. If no, location is checked where if location is less or equal to 4, 1 is predicted and if greater to 4 along with color less or equal to 1 or greater to 1 the prediction is 1.

*Accuracy score of Decision Tree Classifier*

| Metrics/dataset | Train | Test |
|---|---|---|
| Accuracy Score | 99.61% | 99.64% |

*Table 4.1.2(a)*

*Confusion Matrix of Decision Tree Classifier*

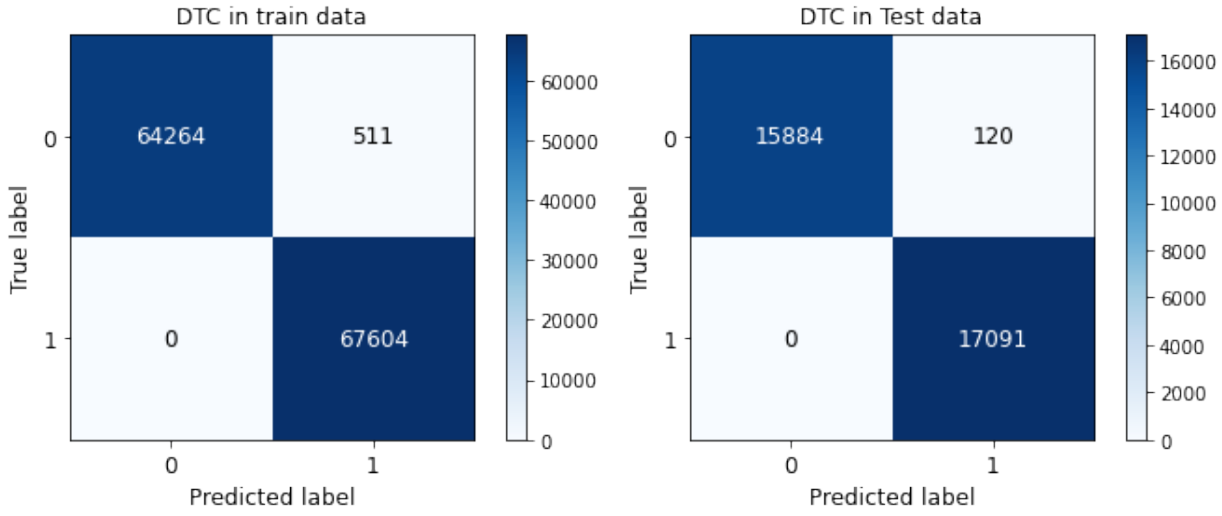


*Figure 4.12(a)* *Figure 4.12(b)*

The model is performing extremely well and so far, there seems to be only one issue that all the incorrect predictions of the models are false positives

- ***Important Features:***

Based on the interpretations the features with the count of their occurrences along with the position or type of node they occurred at is as below.

| Features | Occurrences | Node Type |
|---|---|---|
| Price | 3 | Root , Decision |
| Page1 | 3 | Decision , Leaf |
| Colour | 2 | Decision , Leaf |
| Location | 1 | Decision |

*Table 4.1.2(b)*

- ***Recommendation***

The model's accuracy is extremely good; however, the model's run time is ~10 hours. Changing the custom model or looking at alternative models that fetch similar accuracy with less run time is encouraged.

### 4.1.3 Random Forest:

- ***Definition***

Random forest considers a base model and tries to build a lot of individual decision tree(stumps) that appears like a forest and operate as an ensemble. Each tree is a random split of classes and classes with most votes becomes the models' predictions.

- ***Methodology***

The base model for Random Forest in this project is decision tree. The concept used here is to compare randomly generated trees in a forest with each other and check for class with the most votes to return the class as prediction. The assumption here is to go ahead with the uncorrelated models and decision from each tree helps in other tree too because there is a chance of one tree decision incorrect class , but having most of the tree predicting the same class is of high confidence

- ***Performance***

The train and test data both are ran through the model to fit the performance of the same is as below.

*Accuracy score of Random Forest*

| Metrics/dataset | Train | Test |
|---|---|---|
| Accuracy Score | 100.0% | 100.0% |

*Table 4.1.3(a)*

The accuracy of the model is too good to be true and there is no point of showing a confusion matrix as all predicted are equal to actuals.

- ***Recommendation***

As stated, the accuracy is too good to be true. In a way it is possible because the base model decision tree accuracy is ~99% therefore a little from boost from this model will mostly move into 100%.
With respect to runtime the model is ~5 hours which is better than decision tree . Looking at other models with similar accuracy and less time is recommended.

### 4.1.4 Gradient Boost:

- ***Performance***

The train and test data both are ran through the model to fit the performance of the same is as below.

*Accuracy score of Gradient Boost*

| Metrics/dataset | Train | Test |
|---|---|---|
| Accuracy Score | 51.07% | 51.64% |

*Table 4.1.4(a)*

*Confusion Matrix of Gradient Boost*

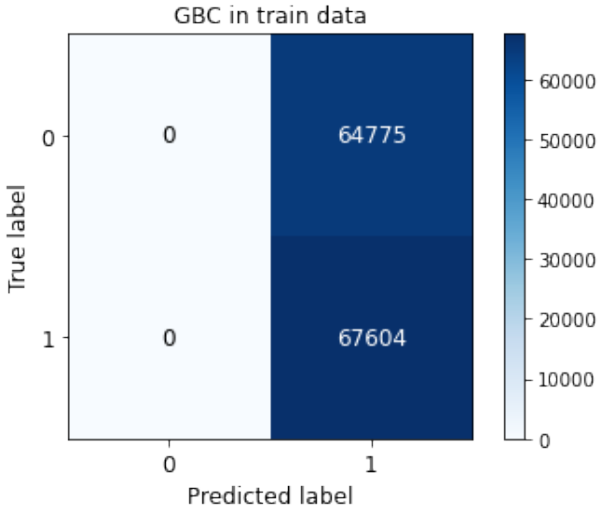


*Figure 4.14(a)*

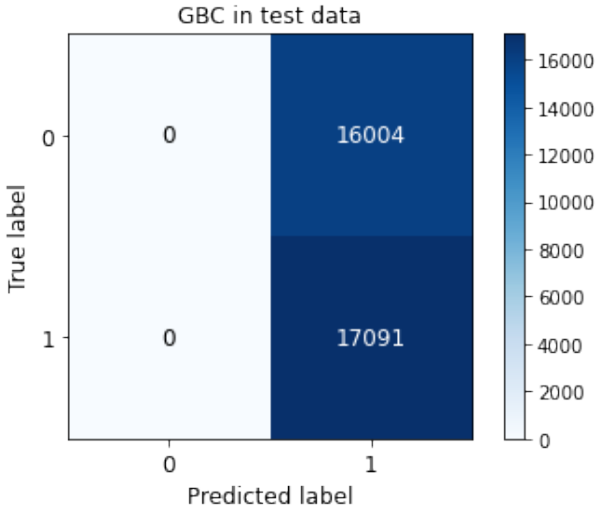


*Figure 4.14(b)*

- ***Recommendation***

There is a definite underfit issue and the model is highly biased towards predicting 1s i.e., ‘Yes’ like earlier models. This

### 4.1.5 Simple Vector Machine[SVM]:

- ***Definition***

SVM is a supervised machine learning algorithm which can be used for both classification and regression model however is most suited for classification. Here our aim is to plot each data point on a space with dimensions equal to the count of features used to classify and try to find the hyperplane that divides data into classes.

- ***Methodology***

The focus is to plot the datapoints and identify the hyperplane. Identifying the hyperplane can be tricky and our intention is to maximize the margin size between data points and hyperplane. The loss function such used is called Hinge loss. Hinge loss is calculated as $min_w \delta||w||^2 + \sum_{i=1}^{n}(1 - y_i\,(x_i, w))$ wherein we try to minimize the training error and maximize the margin. With the help of hinge loss and weights we find gradients where in the weights are in partially derivation. The weights will update using the gradient.
If there is no error in classification , the model is accurate, and it moves ahead to update gradient using regularization parameter. If there is an error in classification, model will include the loss along with the regularization parameter to update the gradient

- ***Performance***

The train and test data both are ran through the model to fit the performance of the same is as below.

*Accuracy score of SVM*

| Metrics/dataset | Train | Test |
|---|---|---|
| Accuracy Score | 51.12% | 51.71% |

*Table 4.1.5(a)*

There is a underfitting issue in the model as both the train and test accuracy is way low than accepted values. To further understand below is the confusion matrix of the predictions against the true/ actual values.

*Confusion Matrix of SVM*

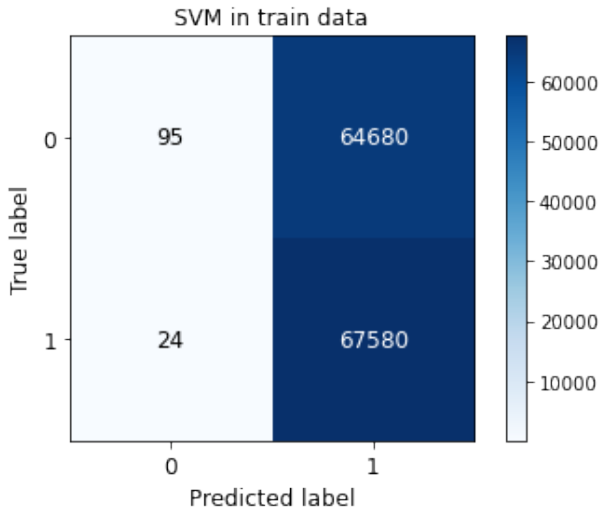


*Figure 4.15(a)*

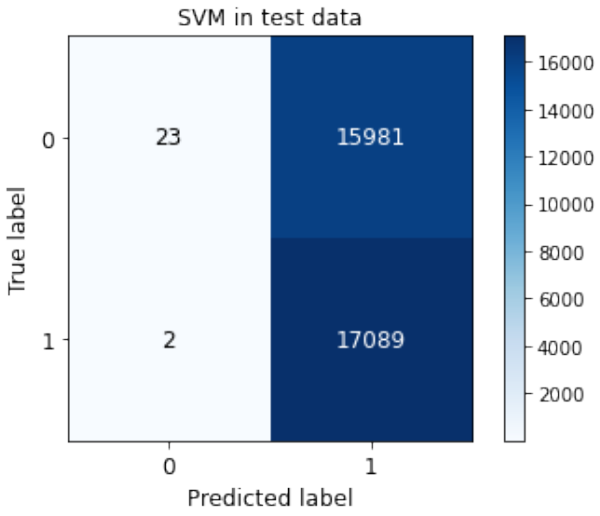


*Figure 4.15(b)*

The strucuture of the confusion matrix is similar to the earlier one however the results are slightly better. This model is slighted more biased towared predicting 1s(Yes) than 0s(No) and that is the major reason for low accuracy. There is a neligble amout of better predictions for No

- ***Recommendation***

The similar situation is seen in SVM too and this is because we have too much of noise in the data which contributes to overlapping of target classes. Excluding few dummy values will help us get better results. The model has failed to perform due to skewness or biased samples towards few specific patterns. Better sampling can improve the performance

### 4.1.6 K-nearest neighbor[KNN]:

- ***Definition***

KNN assumes similar data point exists in proximity and using distance to consider this assumption , Euclidean distance is commonly used

- ***Methodology***

We initial K to a chosen neighbor and for each datapoint the distance is calculated . Distance and index are added and sorted in ascending order. A k is picked from sorted collection and label from k entries. In this model we return the model of k labels

- ***Performance***

The train and test data both are ran through the model to fit the performance of the same is as below.

*Accuracy score of KNN*

| Metrics/dataset | Train | Test |
|---|---|---|
| Accuracy Score | 49.18% | 48.64% |

*Table 4.1.6(a)*

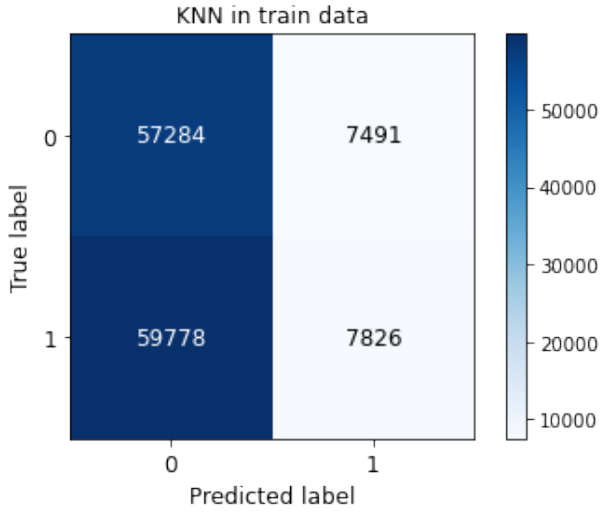


*Figure 4.16(a)*

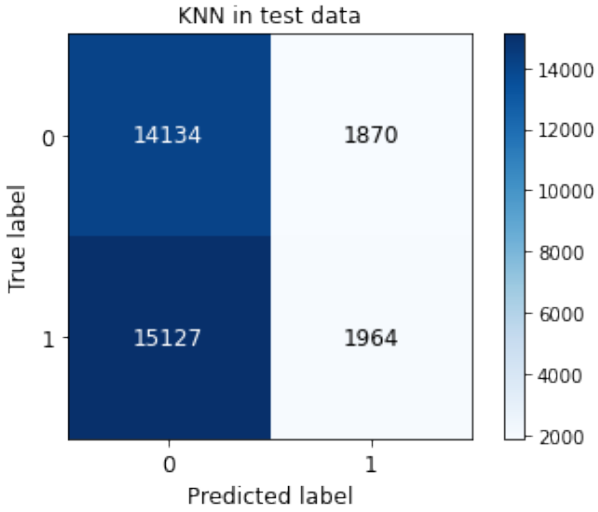


*Figure 4.16(b)*

- ***Recommendation***

The model is not suitable for this use case as the assumption of similar datapoint being close to each out might not be true for the current dataset Thus causing the model to confuse the pattern and there is a definite underfit.

### 4.1.7 Neural Network [NN]:

- ***Definition***

A Neural network where there is an input layer , output layer and few hidden layers in between

- ***Methodology***

The neural network model with 10 hidden layers along with sigmoid activation function and Stochastic Gradient descent with momentum shall be used in the current project

- ***Performance***

The train and test data both are ran through the model to fit the performance of the same is as below.

*Accuracy score of NN*

| Metrics/dataset | Train | Test |
|---|---|---|
| Accuracy Score | 100.0% | 100.0% |

*Table 4.1.7(a)*

# 5. Conclusion and Future Work

## 5.1 Model Comparison

The model performances for both classification and regression model is as below along with the running time so as to compare the model with similar accuracy.

*Performance Comparision for Classification Models*

| Model | AS_train | AS_test | Runtime(sec) | RT in hours |
|---|---|---|---|---|
| Logistic Regression classifier | 51.07% | 51.64% | 3.58 | 0.00 |
| Decision Tree classifier | 99.61% | 99.64% | 38547.88 | 10.71 |
| Random Forest | 100.00% | 100.00% | 18842.12 | 5.23 |
| Gradient Forest | 51.07% | 51.64% | 68807.02 | 19.11 |
| SVM | 51.12% | 51.71% | 1546.21 | 0.43 |
| KNN | 49.18% | 48.64% | 0.03 | 0.00 |
| **NN** | **100.00%** | **100.00%** | **3.67** | **0.00** |

*Table 5.1(a)*

*Performance Comparision for Regression Models*

| R.model | R2_train | R2_test | RMSE_Train | RMSE_Test | Runtime(sec) | RT in hours |
|---|---|---|---|---|---|---|
| Lasso | 78.44% | 78.52% | 5.8279 | 5.8081 | 1.4 | 0.00 |
| **Decision Tree** | **86.63%** | **86.74%** | **4.5897** | **4.5638** | **19012.94** | **5.28** |
| Random Forest | 99.82% | 99.81% | 0.526 | 0.5525 | 53496.19 | 14.86 |
| Gradient Boost | -95.31% | -96.63% | 17.5418 | 17.5733 | 7529.78 | 2.09 |

*Table 5.1(b)*

Apart from individual recommendation for each model mention in the above portion of the report the final thoughts on all the models for this dataset is for classification of instances with higher/lower price going ahead with neural network

along with 10 hidden layer , sigmoid activation function and Stochastic Gradient descent with momentum has fetch better results with utmost time efficiency.
With respect to Regression going ahead with Random forest is better than the other models and it give a very good accuracy . The major drawback is the runtime.

## 5.2 Future Work

For classification model enough model have been tried and test. To explore more we can look at removing some dummy data to get better accuracy on other models. Along with this one can explore Adboost and other boosting techniques to get better performance withing very less time.

For Regression, I would like to explore other models like SVM , KNN and simple Neural network to achieve better accuracy with time efficiency.

# 6. Reference

https://github.com/LikithaJagithyala/E-shop-classification-regression

# 7. Reference


- clickstream data for online shopping. (2019). UCI Machine Learning Repository.

£apczyÒski M., Bia≥owπs S. (2013) Discovering Patterns of Users' Behaviour in an E-shop - Comparison of Consumer Buying Behaviours in Poland and Other European Countries, ìStudia Ekonomiczneî, nr 151, ìLa sociÈtÈ de l'information : perspective europÈenne et globale : les usages et les risques d'Internet pour les citoyens et les consommateursî, p. 144-153.

- Hastie, T., Tibshirani, R., & Friedman, J. (2022). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)* (2nd ed.). Springer.
- QUANTIFYING HEALTH. (n.d.). *Interpret Logistic Regression Coefficients [For Beginners]*. https://quantifyinghealth.com/interpret-logistic-regression-coefficients/

- Kanade, V. (2022, April 18). *What Is Logistic Regression? Equation, Assumptions, Types, and Best Practices*. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/