PREDICTIVE MODELING PROJECT

Name: Praful Thakare
Course: PGP-DSBA Online
Submission Date: 23-Jan-22

Problem 1

1.1 Read the data and do exploratory data analysis. Describe the data briefly. (Check the null values, Data types, shape, EDA, duplicate values). Perform Univariate and Bivariate Analysis. .. 8

1.2. Impute null values if present, also check for the values which are equal to zero. Do they have any meaning or do we need to change them or drop them? Check for the possibility of combining the sub levels of an ordinal variables and take actions accordingly. Explain why you are combining these sub levels with appropriate reasoning…………………………………………5

1.3 Encode the data (having string values) for Modelling. Split the data into train and test (70:30). Apply Linear regression using scikit learn. Perform checks for significant variables using appropriate method from statsmodel. Create multiple models and check the performance of Predictions on Train and Test sets using Rsquare, RMSE & Adj Rsquare. Compare these models and select the best one with appropriate reasoning………………………………………………..12

1.4 Inference: Basis on this Predictions, what are the business insights and recommendations Please explain and summarise the various steps performed in this project. There should be proper business Interpretation and actionable insights present…………………………………………………………7

Problem 2

2.1 Data Ingestion: Read the dataset. Do the descriptive statistics and do null value condition check, write an inference on it. Perform Univariate and Bivariate Analysis. Do exploratory data analysis………………………………………………………………………………………….5

2.2. Do not scale the data. Encode the data (having string values) for Modelling. Data Split: Split the data into train and test (70:30). Apply Logistic Regression and LDA (linear discriminant analysis)……………………………………………………………………………………………7

2.3. Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model Final Model: Compare both the models and write inference which model is best/optimized……………………………………………………………………………………..7

2.4. Inference: Basis on these predictions, what are the insights and recommendations. Please explain and summarise the various steps performed in this project. There should be proper business interpretation and actionable insights present……………………………………………5

Quality of Business Report (Please refer to the Evaluation Guidelines for Business report checklist. Marks in this criteria are at the moderator's discretion)……………………………………6

LIST OF FIGURES

PROBLEM 1

LINEAR REGRESSION

You are hired by a company Gem Stones co ltd, which is a cubic zirconia manufacturer. You are provided with the dataset containing the prices and other attributes of almost 27,000 cubic zirconia (which is an inexpensive diamond alternative with many of the same qualities as a diamond). The company is earning different profits on different prize slots. You have to help the company in predicting the price for the stone on the bases of the details given in the dataset so it can distinguish between higher profitable stones and lower profitable stones so as to have better profit share. Also, provide them with the best 5 attributes that are most important.

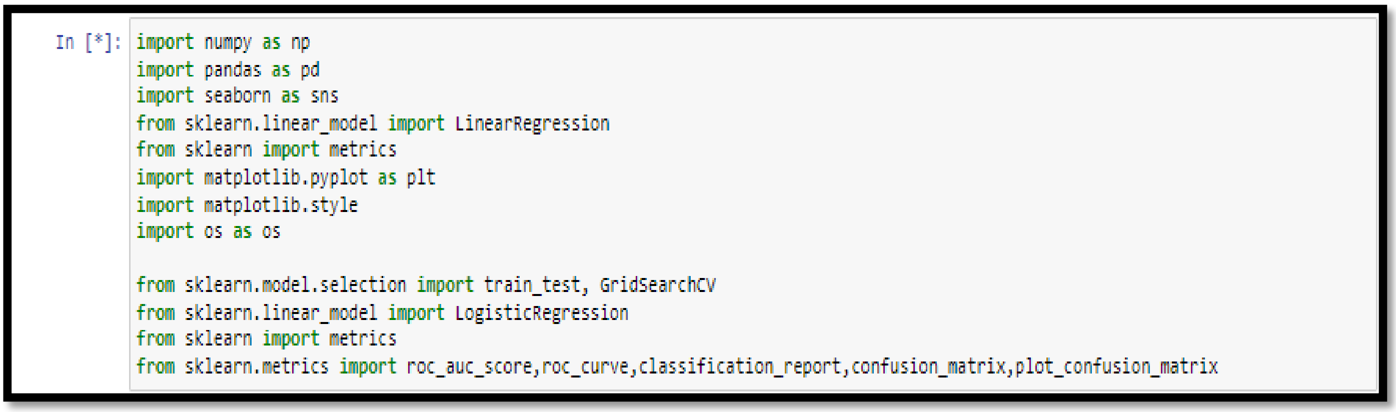
Data Dictionary:

Variable Name	Description
Carat	Carat weight of the cubic zirconia.
Cut	Describe the cut quality of the cubic zirconia. Quality is increasing order Fair, Good, Very Good, Premium, Ideal.
Color	Colour of the cubic zirconia. With D being the worst and J the best.
Clarity	Clarity refers to the absence of the Inclusions and Blemishes. (In order from Worst to Best in terms of avg price) IF, VVS1, VVS2, VS1, VS2, Sl1, Sl2, l1
Depth	The Height of cubic zirconia, measured from the Culet to the table, divided by its average Girdle Diameter.
Table	The Width of the cubic zirconia's Table expressed as a Percentage of its Average Diameter.
Price	The Price of the cubic zirconia.
X	Length of the cubic zirconia in mm.
Y	Width of the cubic zirconia in mm.
Z	Height of the cubic zirconia in mm.

Q1.1 Read the data and do exploratory data analysis. Describe the data briefly. (Check the null values, Data types, shape, EDA, duplicate values). Perform Univariate and Bivariate Analysis.

Answer –

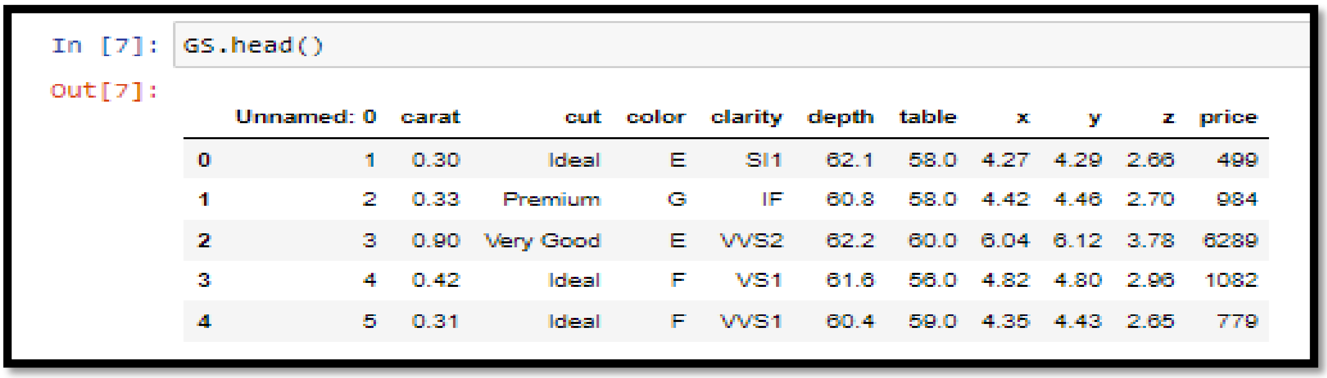
Exploratory Data Analysis or (EDA) is understanding the data sets by summarizing their main characteristics often plotting them visually. This step is very important especially when we arrive at modelling the data. Plotting in EDA consists of Histograms, Box plot, pairplot and many more. It often takes much time to explore the data. Through the process of EDA, we can define the problem statement or definition on our data set which is very important.
Imported the required libraries.



```
In [*]: import numpy as np
        import pandas as pd
        import seaborn as sns
        from sklearn.linear_model import LinearRegression
        from sklearn import metrics
        import matplotlib.pyplot as plt
        import matplotlib.style
        import os as os

        from sklearn.model.selection import train_test, GridSearchCV
        from sklearn.linear_model import LogisticRegression
        from sklearn import metrics
        from sklearn.metrics import roc_auc_score,roc_curve,classification_report,confusion_matrix,plot_confusion_matrix
```

- ➢ Importing the dataset - “cubic_zirconia.csv”
- ➢ Data Summary and Exploratory Data Analysis:

  - ✓ Checking if the data is being imported properly
  - ✓ Head – The top 5 rows of the dataset are viewed using head () function

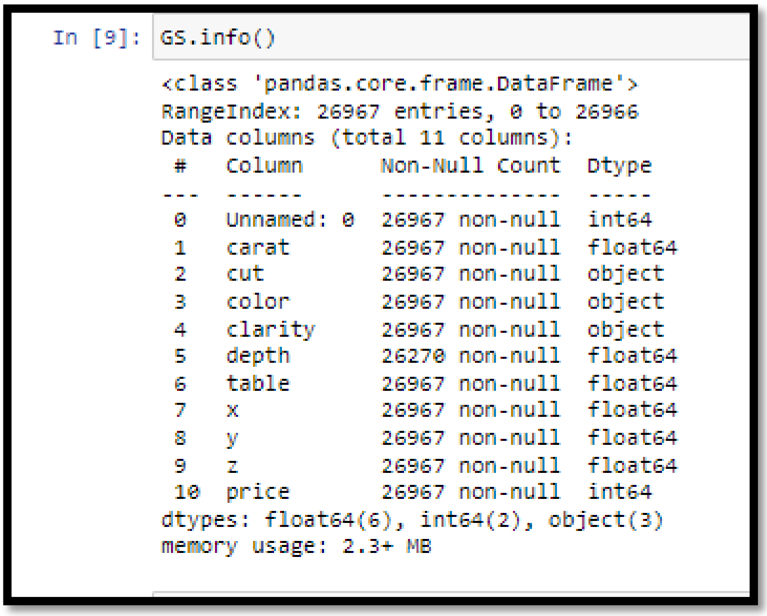


```
In [7]: GS.head()
Out[7]:
```

	Unnamed: 0	carat	cut	color	clarity	depth	table	x	y	z	price
0	1	0.30	Ideal	E	SI1	62.1	58.0	4.27	4.29	2.66	499
1	2	0.33	Premium	G	IF	60.8	58.0	4.42	4.46	2.70	984
2	3	0.90	Very Good	E	VVS2	62.2	60.0	6.04	6.12	3.78	6289
3	4	0.42	Ideal	F	VS1	61.6	56.0	4.82	4.80	2.96	1082
4	5	0.31	Ideal	F	VVS1	60.4	59.0	4.35	4.43	2.65	779

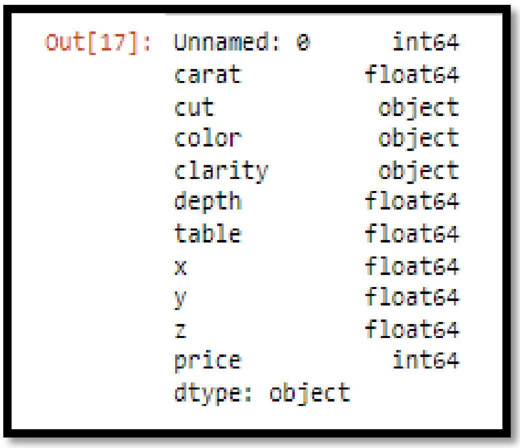
  - ✓ Dimension of the Dataset: The Dimension or shape of the dataset can be shown using **shape** function. It shows that the dataset given to us has 26967 rows and 11 columns or variables

  - ✓ Structure of the Dataset: Structure of the dataset can be computed using **.info ()** function



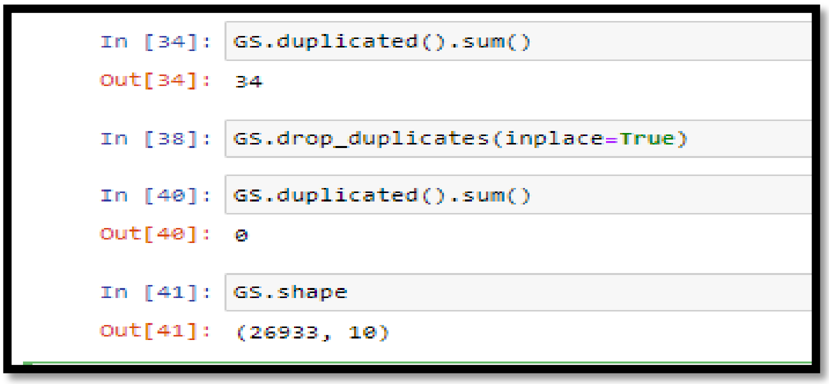
```
In [9]: GS.info()

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 26967 entries, 0 to 26966
        Data columns (total 11 columns):
         #   Column      Non-Null Count  Dtype
        ---  ------      --------------  -----
         0   Unnamed: 0  26967 non-null  int64
         1   carat       26967 non-null  float64
         2   cut         26967 non-null  object
         3   color       26967 non-null  object
         4   clarity     26967 non-null  object
         5   depth       26270 non-null  float64
         6   table       26967 non-null  float64
         7   x           26967 non-null  float64
         8   y           26967 non-null  float64
         9   z           26967 non-null  float64
         10  price       26967 non-null  int64
        dtypes: float64(6), int64(2), object(3)
        memory usage: 2.3+ MB
```

- Data Type is – Integer/Float/Object.



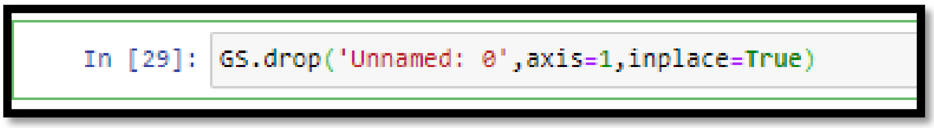
```
Out[17]: Unnamed: 0      int64
         carat         float64
         cut            object
         color          object
         clarity        object
         depth         float64
         table         float64
         x             float64
         y             float64
         z             float64
         price           int64
         dtype: object
```

- Checking for Duplicates: - There are 34 duplicate rows in the dataset as computed using. Duplicated () function. We will drop the duplicates.



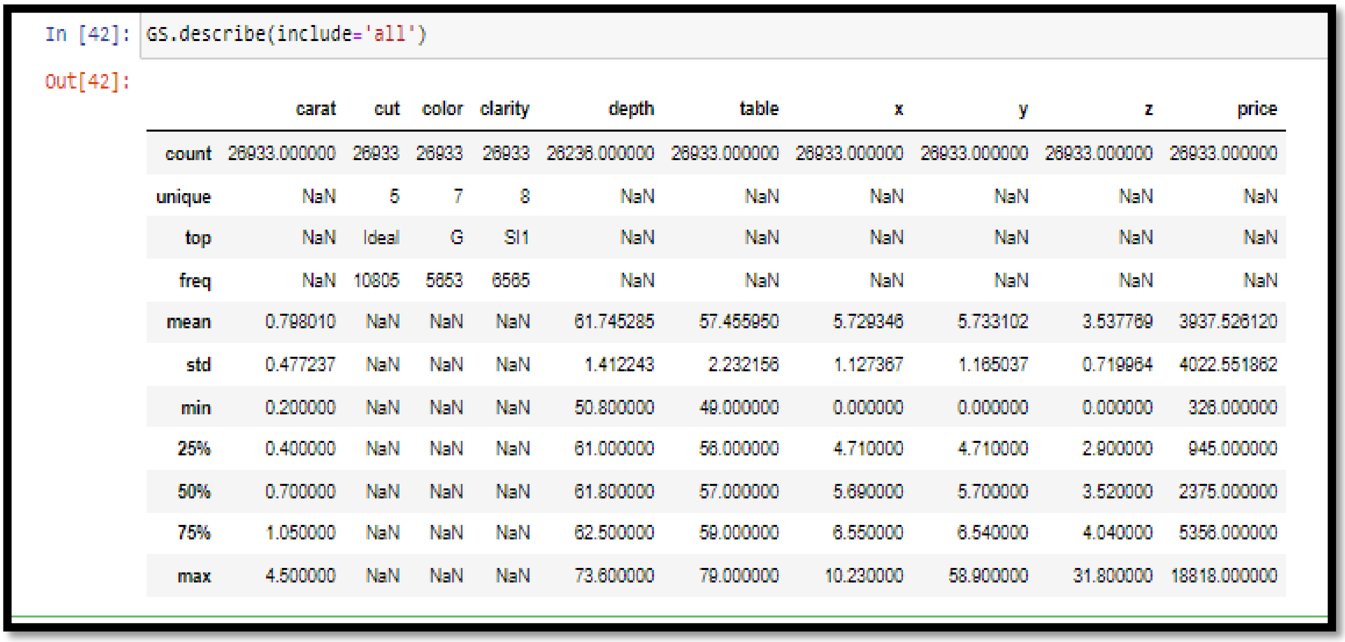
```
In [34]: GS.duplicated().sum()
Out[34]: 34

In [38]: GS.drop_duplicates(inplace=True)

In [40]: GS.duplicated().sum()
Out[40]: 0

In [41]: GS.shape
Out[41]: (26933, 10)
```

- We will drop the first column ‘**Unnamed: 0’** column as this is not important for our study.



```
In [29]: GS.drop('Unnamed: 0',axis=1,inplace=True)
```

Descriptive Statistics for the dataset:

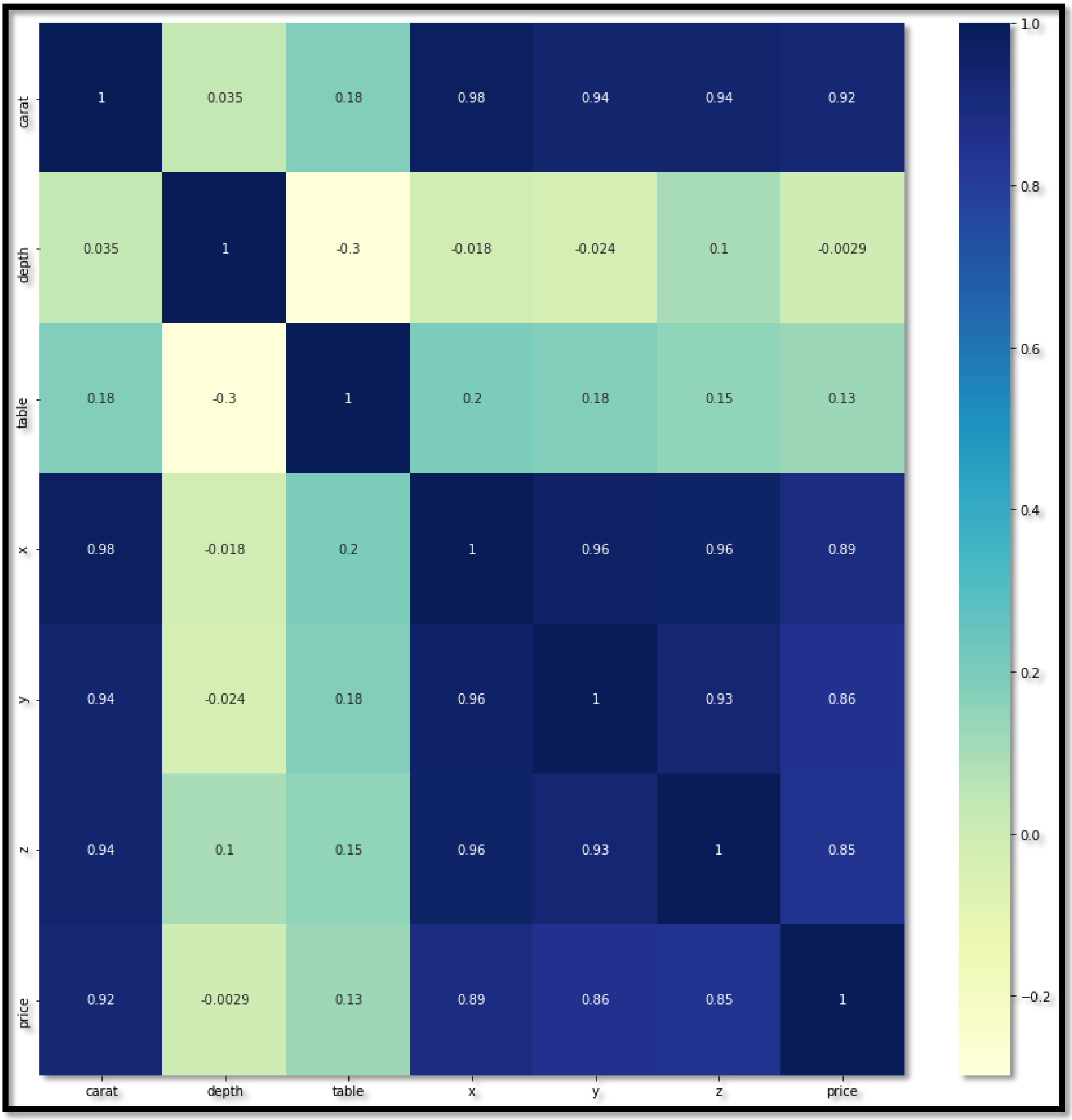


```
In [42]: GS.describe(include='all')
```

```
Out[42]:
              carat    cut  color  clarity         depth         table             x             y             z         price
count  26933.000000  26933  26933    26933  26236.000000  26933.000000  26933.000000  26933.000000  26933.000000  26933.000000
unique          NaN      5      7        8           NaN           NaN           NaN           NaN           NaN           NaN
top             NaN  Ideal      G      SI1           NaN           NaN           NaN           NaN           NaN           NaN
freq            NaN  10805   5653     6565           NaN           NaN           NaN           NaN           NaN           NaN
mean       0.798010    NaN    NaN      NaN     61.745285     57.455950      5.729346      5.733102      3.537769   3937.526120
std        0.477237    NaN    NaN      NaN      1.412243      2.232156      1.127367      1.165037      0.719964   4022.551862
min        0.200000    NaN    NaN      NaN     50.800000     49.000000      0.000000      0.000000      0.000000    326.000000
25%        0.400000    NaN    NaN      NaN     61.000000     56.000000      4.710000      4.710000      2.900000    945.000000
50%        0.700000    NaN    NaN      NaN     61.800000     57.000000      5.690000      5.700000      3.520000   2375.000000
75%        1.050000    NaN    NaN      NaN     62.500000     59.000000      6.550000      6.540000      4.040000   5356.000000
max        4.500000    NaN    NaN      NaN     73.600000     79.000000     10.230000     58.900000     31.800000  18818.000000
```

- **Carat**: - This is an independent variable, and it ranges from 0.2 to 4.5. Mean value is around 0.8 and 75% of the stones are of 1.05 carat value. Standard deviation is around 0.477 which shows that the data is skewed and has a right tailed curve. Which means that majority of the stones are of lower carat. There are very few stones above 1.05 carat.
- **Depth**: - The percentage height of cubic zirconia stones is in the range of 50.80 to 73.60. Average height of the stones is 61.80, 25% of the stones are 61 and 75% of the stones are 62.5. Standard deviation of the height of the stones is 1.4. Standard deviation is indicating a normal distribution
- **Table**: - The percentage width of cubic Zirconia is in the range of 49 to 79. Average is around 57. 25% of stones are below 56 and 75% of the stones have a width of less than 59. Standard deviation is 2.24. Thus the data does not show normal distribution and is similar to carat with most of the stones having less width also this shows outliers are present in the variable.
- **Price**: - Price is the Predicted variable. Prices are in the range of 3938 to 18818. Median price of stones is 2375, while 25% of the stones are priced below 945. 75% of the stones are in the price range of 5356. Standard deviation of the price is 4022. Indicating prices of majority of the stones are in lower range as the distribution is right skewed.
- Variables x, y, and z seems to follow a normal distribution with a few outliers.

Checking Correlation in the data using Heatmap



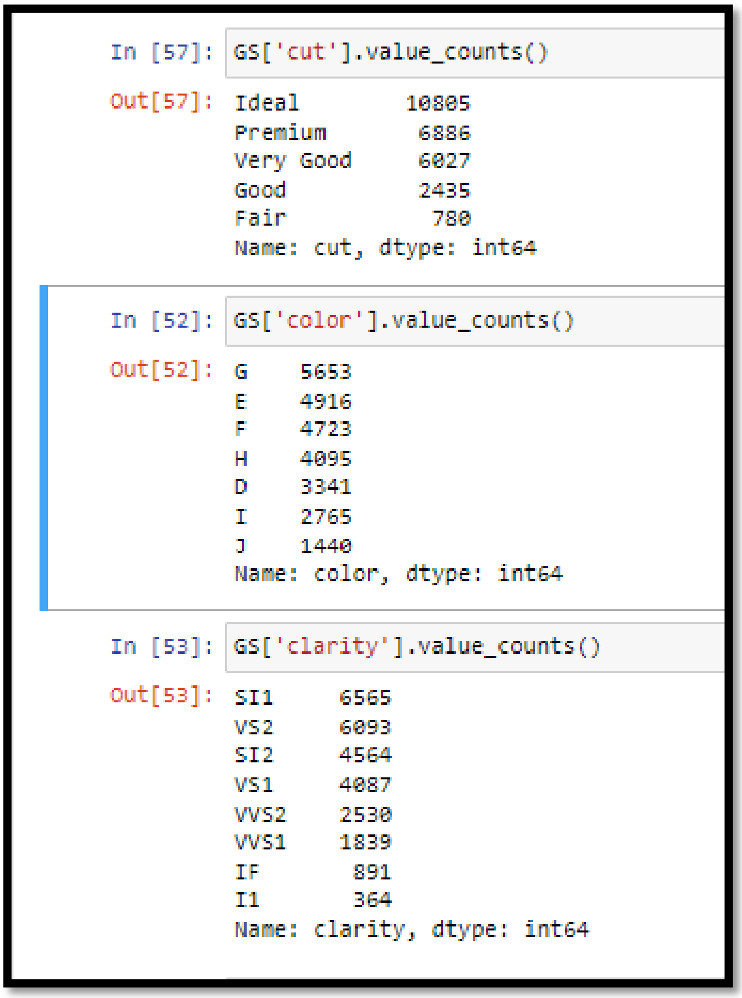
	carat	depth	table	x	y	z	price
carat	1	0.035	0.18	0.98	0.94	0.94	0.92
depth	0.035	1	-0.3	-0.018	-0.024	0.1	-0.0029
table	0.18	-0.3	1	0.2	0.18	0.15	0.13
x	0.98	-0.018	0.2	1	0.96	0.96	0.89
y	0.94	-0.024	0.18	0.96	1	0.93	0.86
z	0.94	0.1	0.15	0.96	0.93	1	0.85
price	0.92	-0.0029	0.13	0.89	0.86	0.85	1

Fig.1 – HeatMap on Dataset

Observations:

- High correlation between the different features like carat, x, y, z and price.
- Less correlation between table with the other features.
- Depth is negatively correlated with most the other features except for carat

Univariate & Bivariate Analysis
Getting unique counts of Categorical Variables



```
In [57]: GS['cut'].value_counts()

Out[57]: Ideal        10805
         Premium       6886
         Very Good     6027
         Good          2435
         Fair           780
         Name: cut, dtype: int64

In [52]: GS['color'].value_counts()

Out[52]: G    5653
         E    4916
         F    4723
         H    4095
         D    3341
         I    2765
         J    1440
         Name: color, dtype: int64

In [53]: GS['clarity'].value_counts()

Out[53]: SI1     6565
         VS2     6093
         SI2     4564
         VS1     4087
         VVS2    2530
         VVS1    1839
         IF       891
         I1       364
         Name: clarity, dtype: int64
```

Looking at the above unique values for variable “Cut “ we see the ranking given for each unique value like “ **Fair, Good, Ideal, Premium, Very Good**

<u>Distribution of Cut Variable</u>

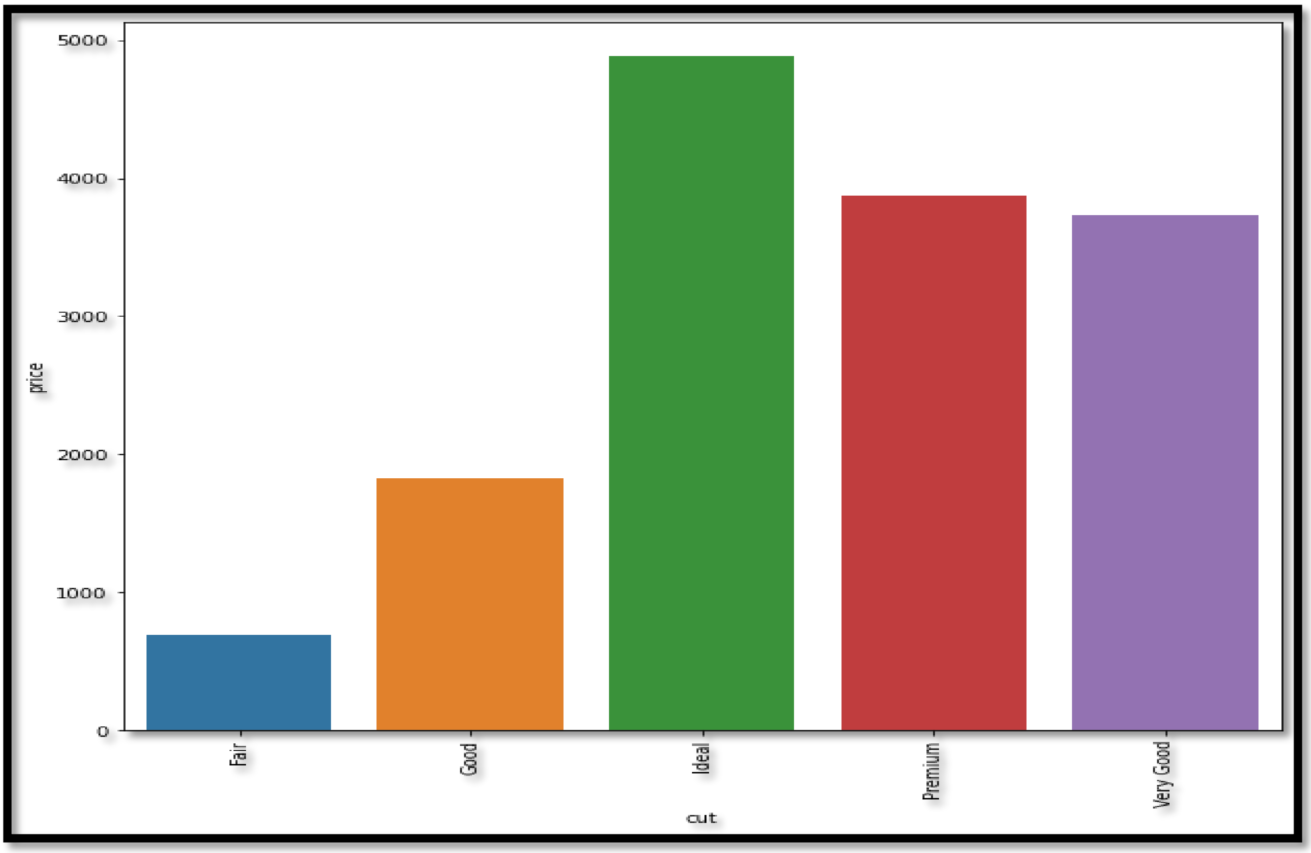


Fig.2 – Distribution of Cut Variable

- For the cut variable we see the most sold is Ideal cut type gems and least sold is Fair cut gems
- Slightly less priced seems to be Ideal type and premium cut type to be slightly more expensive

Distribution of Color Variable

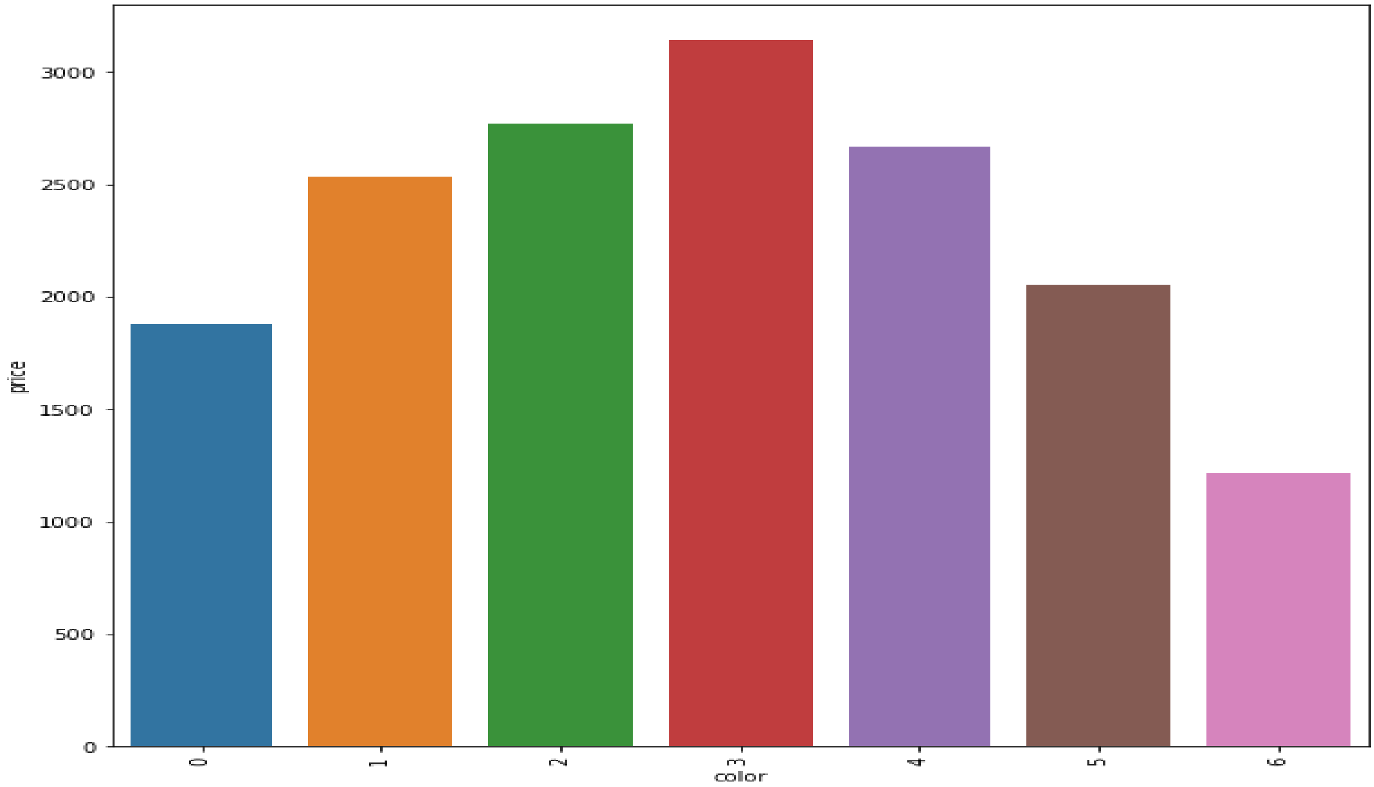


Fig.3 – Distribution of Color variable

- For the color variable we see the most sold is G colored gems and least is J colored gems
- However, the least priced seems to be E type; J and I colored gems seems to be more expensive ‘’

‘’¶

Distribution of Clarity Variable

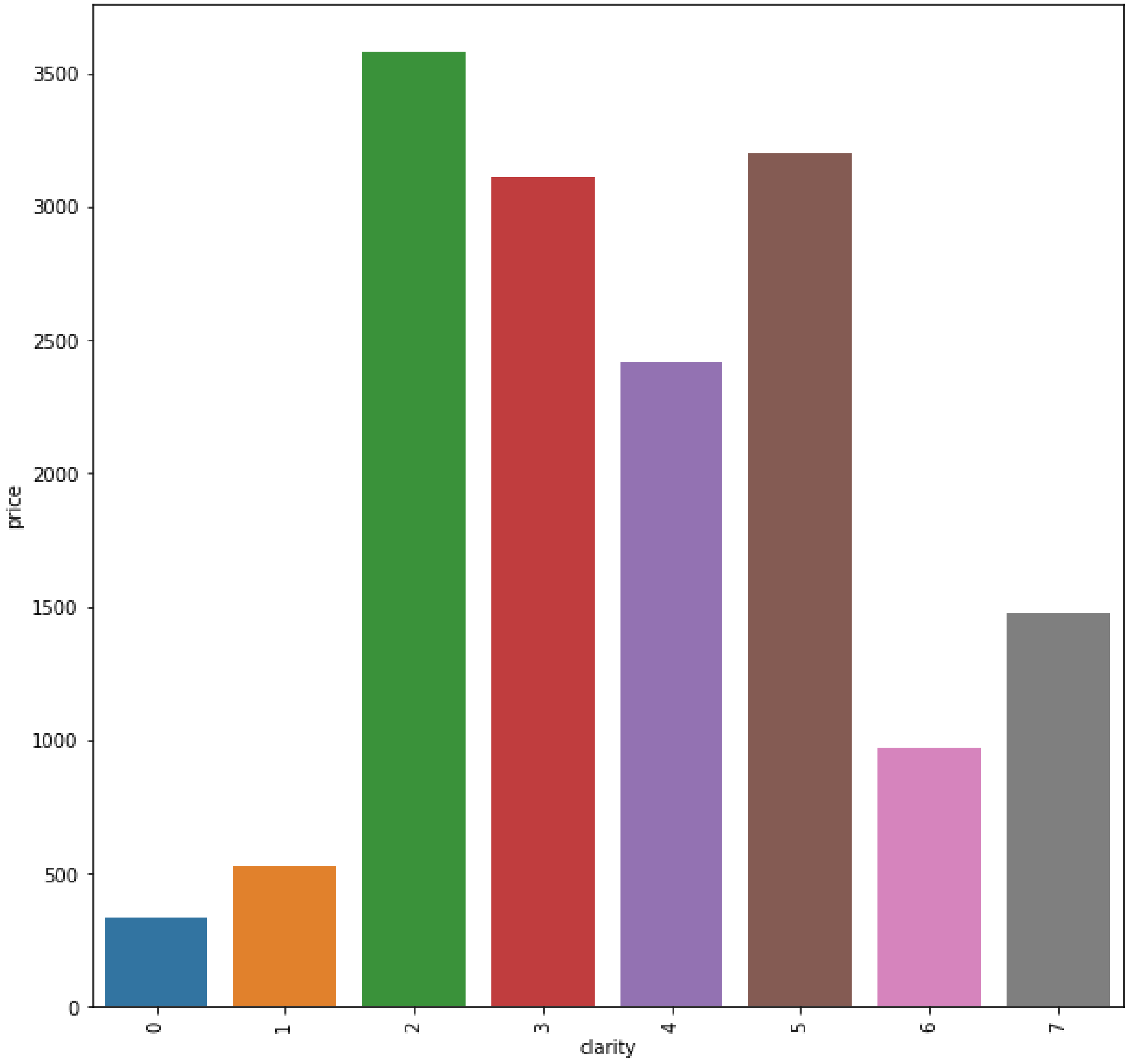


Fig.4 – Distribution of Clarity variable

- For the clarity variable we see the most sold is SI1 clarity gems and least is I1 clarity gems
- Slightly less priced seems to be SI1 type; VS2 and SI2 clarity stones seems to be more expensive

Observations:

Independent Variables

- Depth is the only variable which can be considered as normal distribution
- Carat, Table, x, y, z these variables have multiple modes with the spread of data
- Outliers: Large number of outliers are present in all the variables (Carat, Depth, Table, x, y, z)

Price will be the target variable or dependent variable

- It is right skewed with large range of outliers

There are outliers present in all the variables as per the below plots

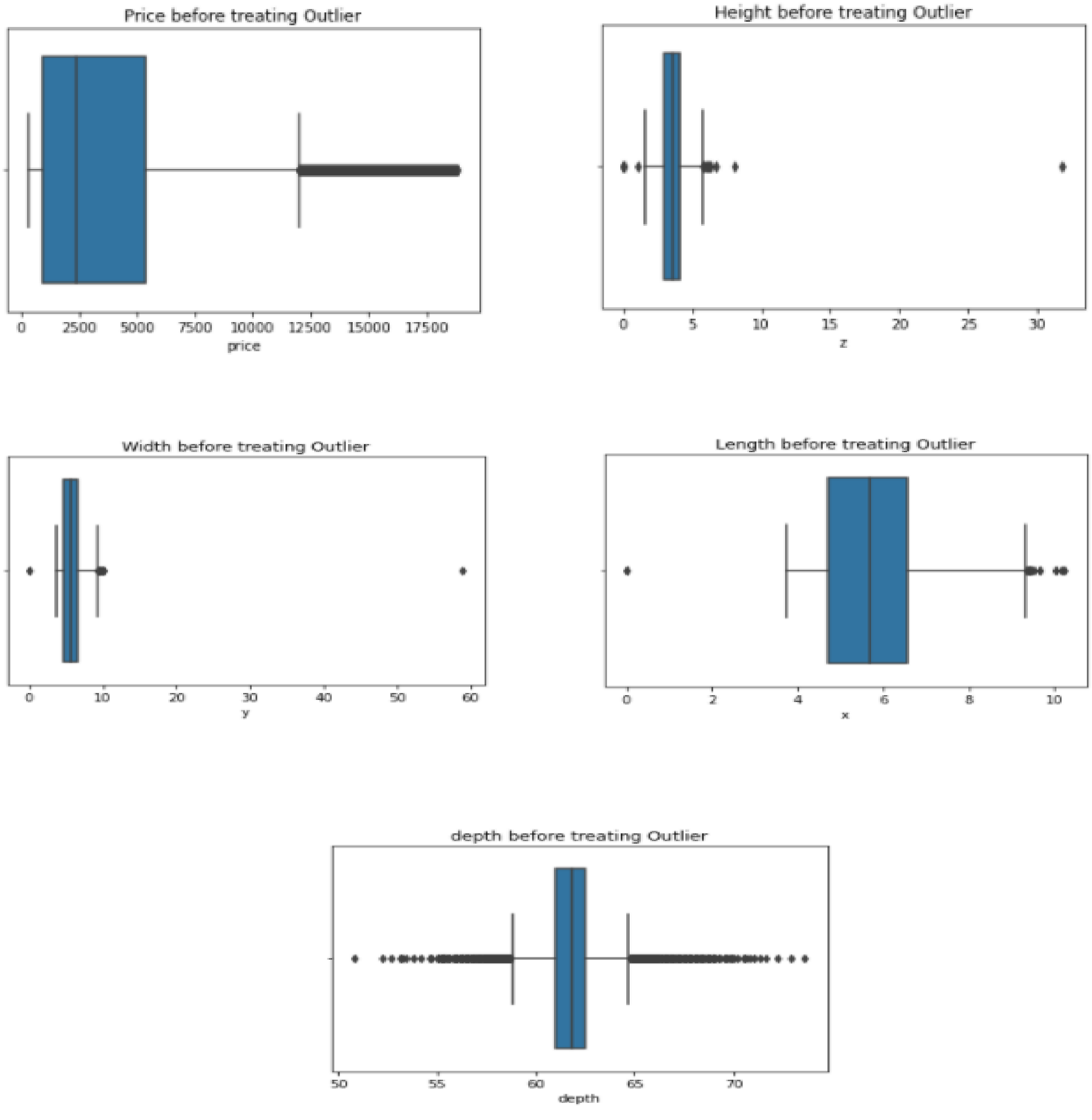


Fig.5 – Box Plot before outlier treatment

Treatment of outliers:

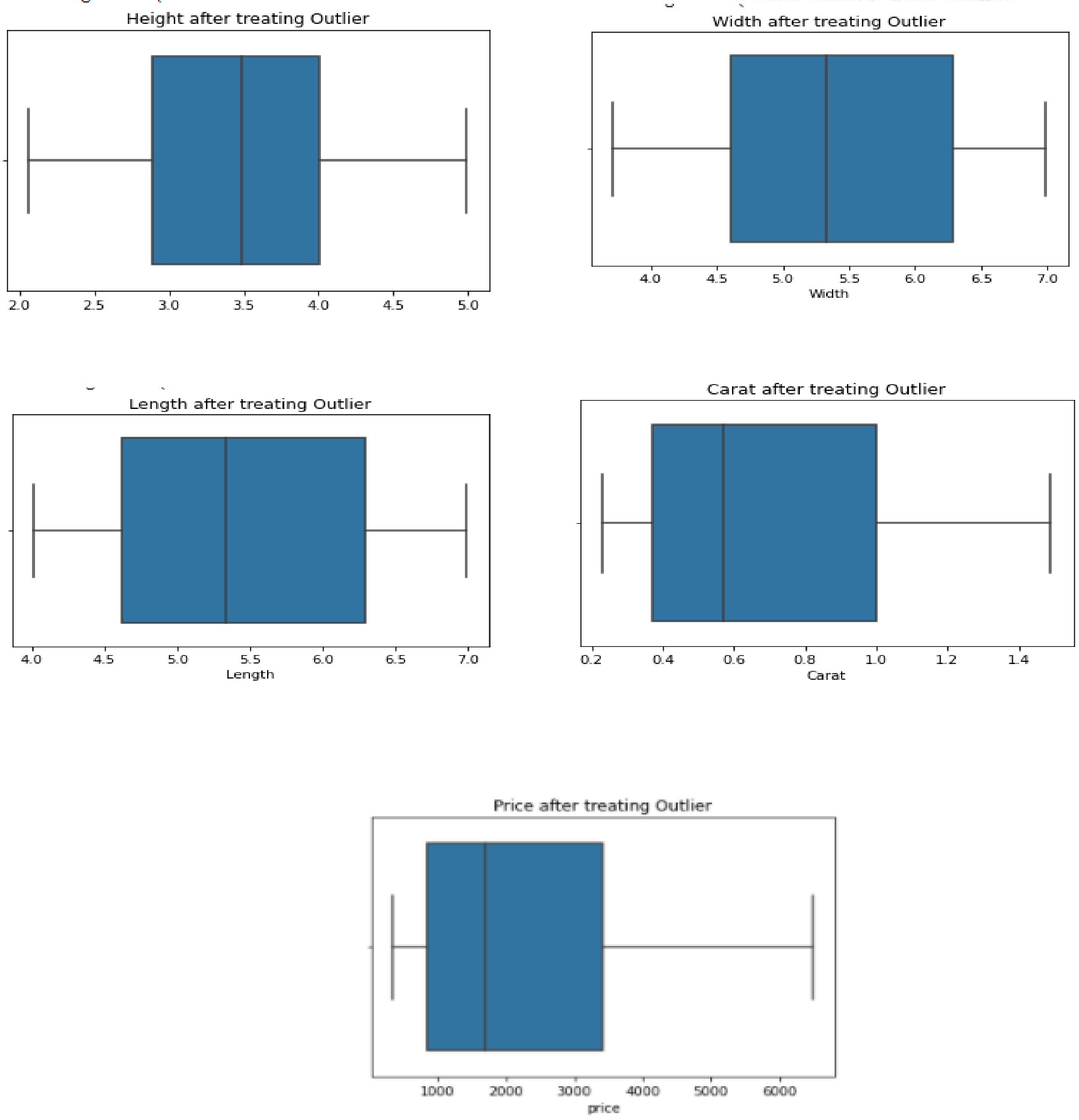


Fig.6 – Box Plot after outlier treatment

- Checked for data Correlation via heatmap: Heatmap showing correlation between variables

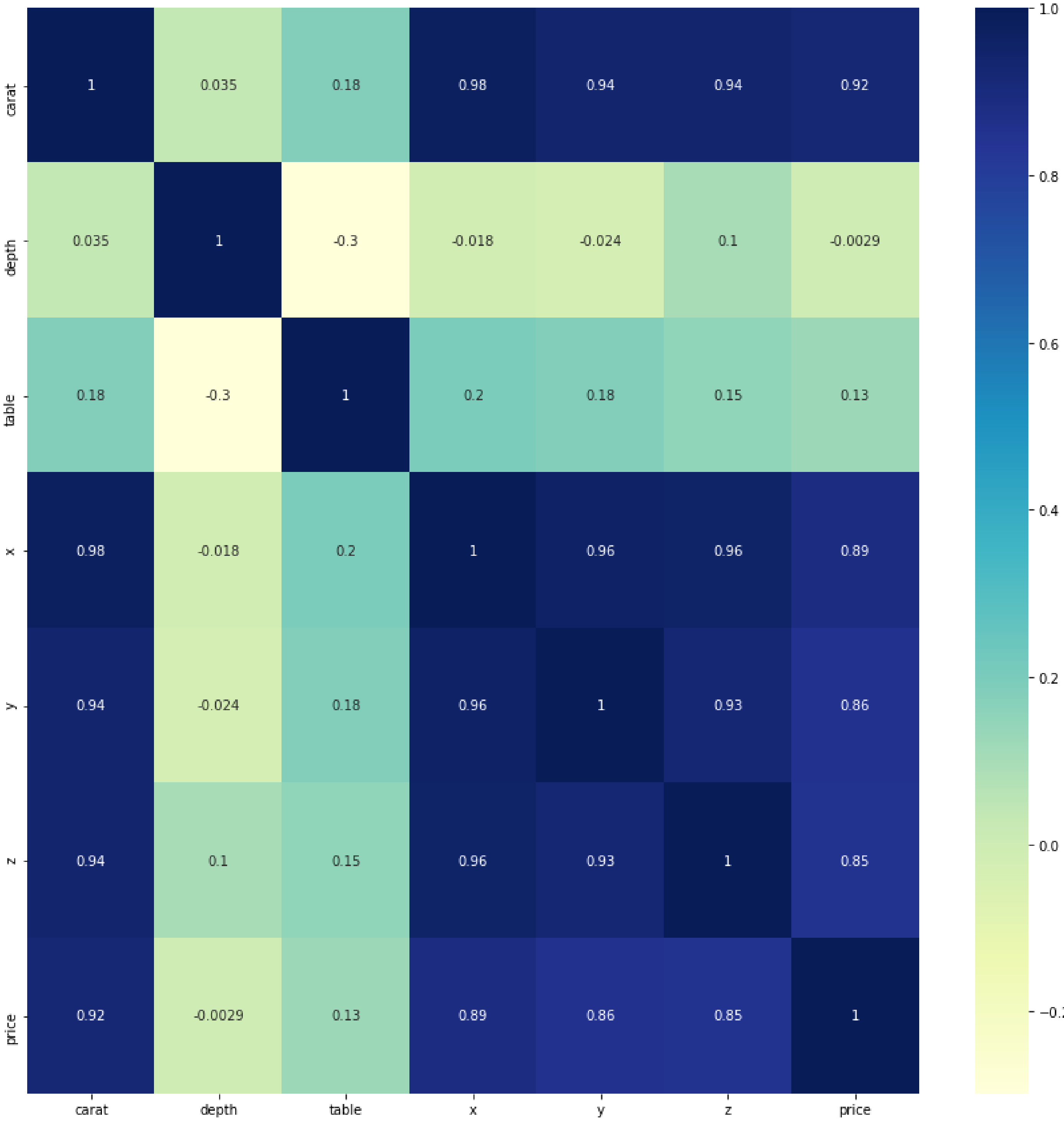


Fig.7 - Heatmap showing correlation between variables

Bivariate Analysis:

Pair Plot:

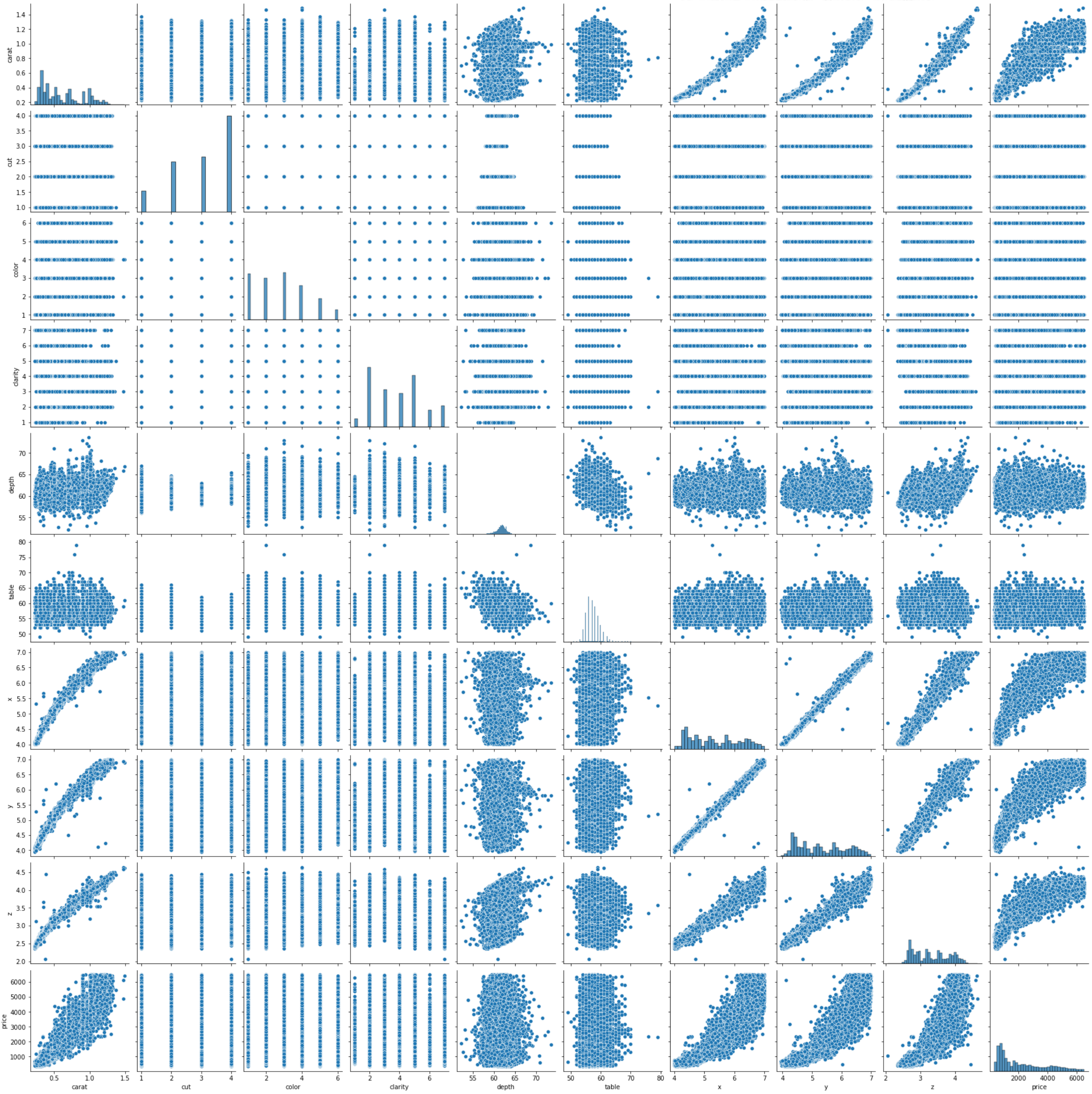


Fig.8 – Pairplot showing bivariate analysis

Observations:

- Pair plot allows us to see both distribution of single variable and relationships between two variables.

Conclusion of EDA:

- Price – This variable gives the continuous output with the price of the cubic zirconia stones. This will be our Target Variable.
- Carat, depth, table, x, y, z variables are numerical or continuous variables.
- Cut, Clarity and color are categorical variables.

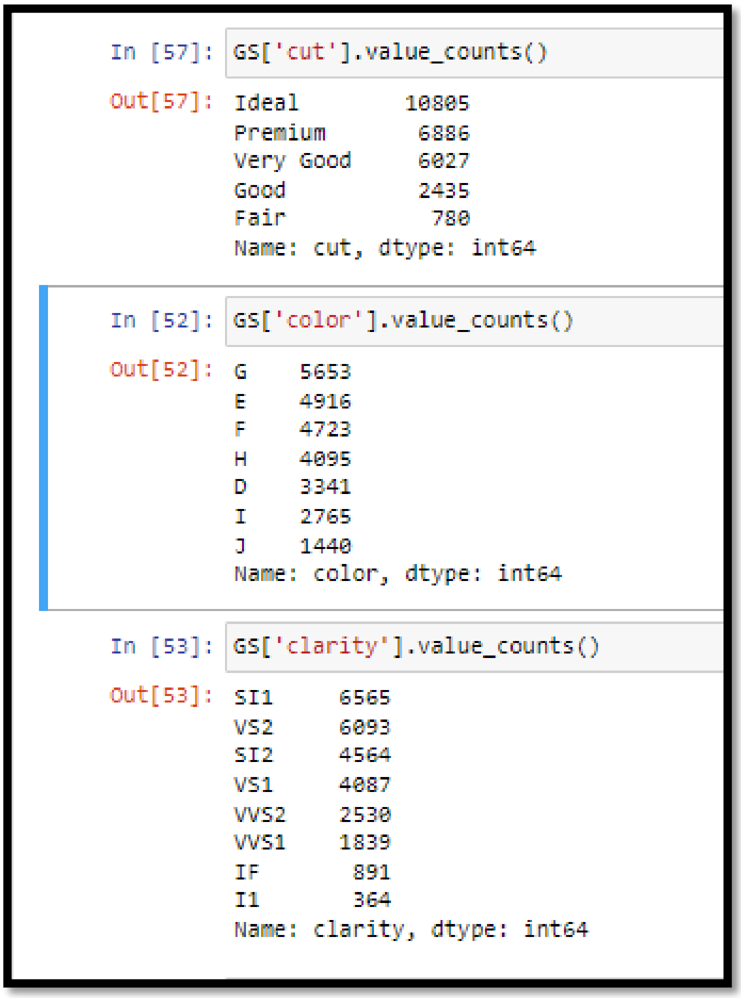
- ➢ We will drop the first column 'Unnamed: 0' column as this is not important for our study which leaves the shape of the dataset with 26967 rows & 10 Columns
- ➢ Only in 'depth 697missing values are present which we will impute by its median values.
- ➢ There are total of 34 duplicate rows as computed using. Duplicated () function. We will drop the duplicates
- ➢ Upon dropping the duplicates – The shape of the data set is – 26933 rows & 10 columns

Q1.2 Impute null values if present, also check for the values which are equal to zero. Do they have any meaning or do we need to change them or drop them? Check for the possibility of combining the sub levels of a ordinal variables and take actions accordingly. Explain why you are combining these sub levels with appropriate reasoning.

Answer –

It is important to check if the data has any missing value or gibberish data in it. We did check about the same for both object and numerical data types and can confirm the following:

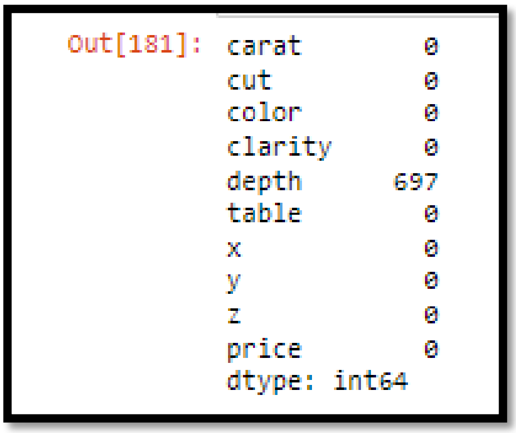
✓ There is no gibberish or missing data in the object type data columns – Cut, color and clarity



```
In [57]: GS['cut'].value_counts()

Out[57]: Ideal        10805
         Premium       6886
         Very Good     6027
         Good          2435
         Fair           780
         Name: cut, dtype: int64

In [52]: GS['color'].value_counts()

Out[52]: G    5653
         E    4916
         F    4723
         H    4095
         D    3341
         I    2765
         J    1440
         Name: color, dtype: int64

In [53]: GS['clarity'].value_counts()

Out[53]: SI1     6565
         VS2     6093
         SI2     4564
         VS1     4087
         VVS2    2530
         VVS1    1839
         IF       891
         I1       364
         Name: clarity, dtype: int64
```

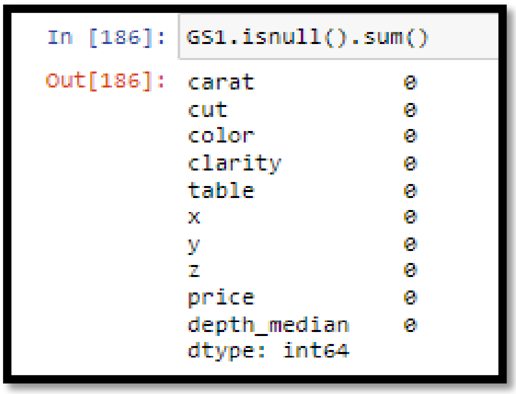
- ✓ There are missing values in the column "depth" – 697 cells or 2.6% of the total data set. We can choose to impute these values using a mean or median. We checked for both the values and the result for both is almost similar.
- ✓ For this case study, I have used median to impute the missing values.

- ✓ Table 1.2.1 – Checking the missing values



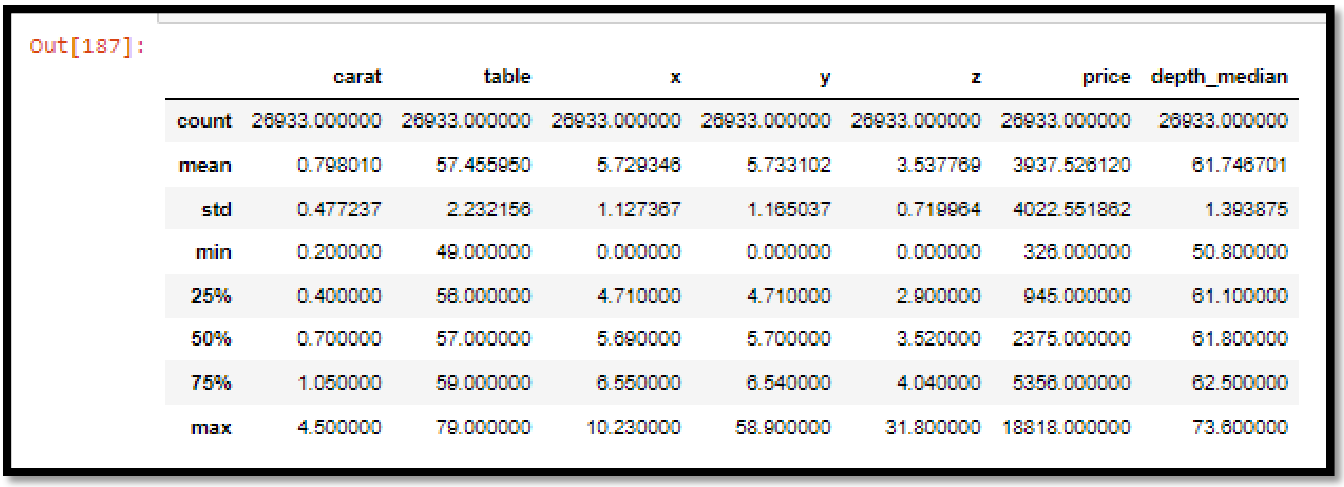
```
Out[181]: carat         0
          cut           0
          color         0
          clarity       0
          depth       697
          table         0
          x             0
          y             0
          z             0
          price         0
          dtype: int64
```

- ✓ Table 1.2.2. – checking for missing values after imputing the values



```
In [186]: GS1.isnull().sum()

Out[186]: carat           0
          cut             0
          color           0
          clarity         0
          table           0
          x               0
          y               0
          z               0
          price           0
          depth_median    0
          dtype: int64
```

- ✓ Table: 1.2.3: Describe Function showing presence of 0 values in x, y, and z columns



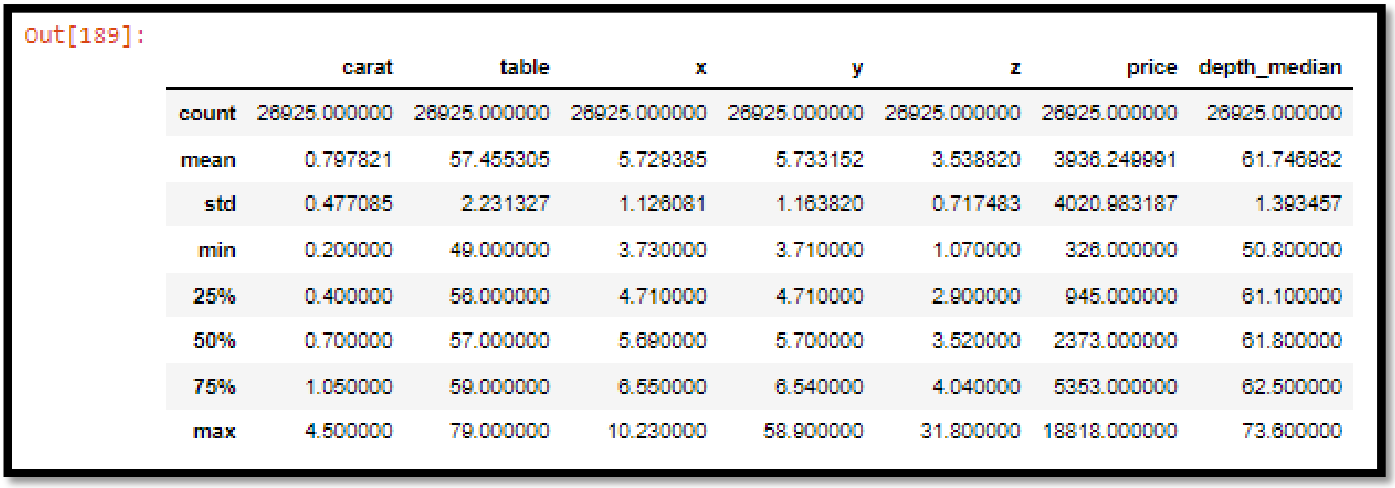
Out[187]:

	carat	table	x	y	z	price	depth_median
count	26933.000000	26933.000000	26933.000000	26933.000000	26933.000000	26933.000000	26933.000000
mean	0.798010	57.455950	5.729346	5.733102	3.537769	3937.526120	61.746701
std	0.477237	2.232156	1.127367	1.165037	0.719964	4022.551862	1.393875
min	0.200000	49.000000	0.000000	0.000000	0.000000	326.000000	50.800000
25%	0.400000	56.000000	4.710000	4.710000	2.900000	945.000000	61.100000
50%	0.700000	57.000000	5.690000	5.700000	3.520000	2375.000000	61.800000
75%	1.050000	59.000000	6.550000	6.540000	4.040000	5356.000000	62.500000
max	4.500000	79.000000	10.230000	58.900000	31.800000	18818.000000	73.600000

- ✓ While there are no missing values in the numerical columns, there are a few 0 in columns – x (3 in count), y(3 in count) and z (9 in count) – in the database. A single row was dealt with during checking for duplicates and the other eight rows were taken care here. Since the total number of rows that had 0 value in them was 8 only, it accounts for a negligible number and for this case study we could have avoided them or dropped. Also, when I checked the correlation values, it seems there is a strong multicollinearity between all three columns. There is a most likely case that I won't even use them in creating my Linear Regression model. I have chosen to drop those rows as it represented an insignificant number when compared to the overall dataset and it won't add much value to the analysis here.

✓ GS1 = GS1[(GS1["x"] != 0) & (GS1["y"] != 0) & (GS1["z"] != 0)]

Table: 1.2.4: Describe Function confirming there are no 0 values in x, y, and z columns



Out[189]:

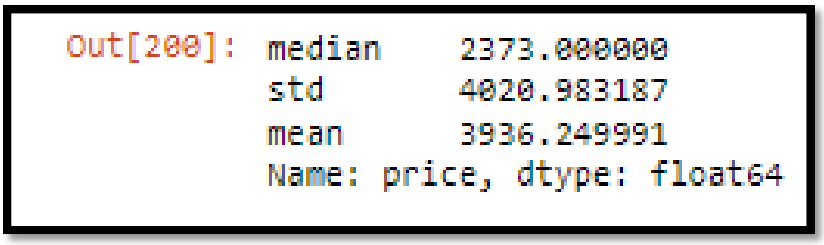
	carat	table	x	y	z	price	depth_median
count	26925.000000	26925.000000	26925.000000	26925.000000	26925.000000	26925.000000	26925.000000
mean	0.797821	57.455305	5.729385	5.733152	3.538820	3936.249991	61.746982
std	0.477085	2.231327	1.126081	1.163820	0.717483	4020.983187	1.393457
min	0.200000	49.000000	3.730000	3.710000	1.070000	326.000000	50.800000
25%	0.400000	56.000000	4.710000	4.710000	2.900000	945.000000	61.100000
50%	0.700000	57.000000	5.690000	5.700000	3.520000	2373.000000	61.800000
75%	1.050000	59.000000	6.550000	6.540000	4.040000	5353.000000	62.500000
max	4.500000	79.000000	10.230000	58.900000	31.800000	18818.000000	73.600000

We now have a dataset with 26,925 rows as supposed to 26,933 after treating duplicate entries.

Q.1.3 Encode the data (having string values) for Modelling. Split the data into train and test (70:30). Apply Linear regression using scikit learn. Perform checks for significant variables using appropriate method from statsmodel. Create multiple models and check the performance of Predictions on Train and Test sets using Rsquare, RMSE & Adj Rsquare. Compare these models and select the best one with appropriate reasoning.

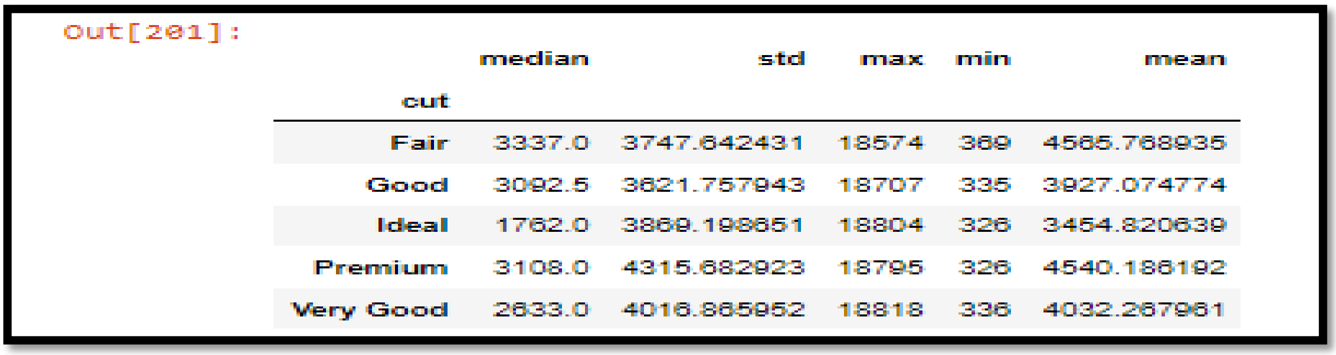
Answer –

We have three object columns, which have string values - cut, color, and clarity
Let us read the data in brief before deciding on what kind of encoding technique needs to be used.
I would quickly check the statistical inference of our target variable – price, with the following output:



```
Out[200]: median      2373.000000
          std         4020.983187
          mean        3936.249991
          Name: price, dtype: float64
```

Let us now check the first object column – cut for quick read. I am using the groupby function to check the relationship of cut with price and have used some aggregator functions likes mean, median, and standard deviation to have a quick read.

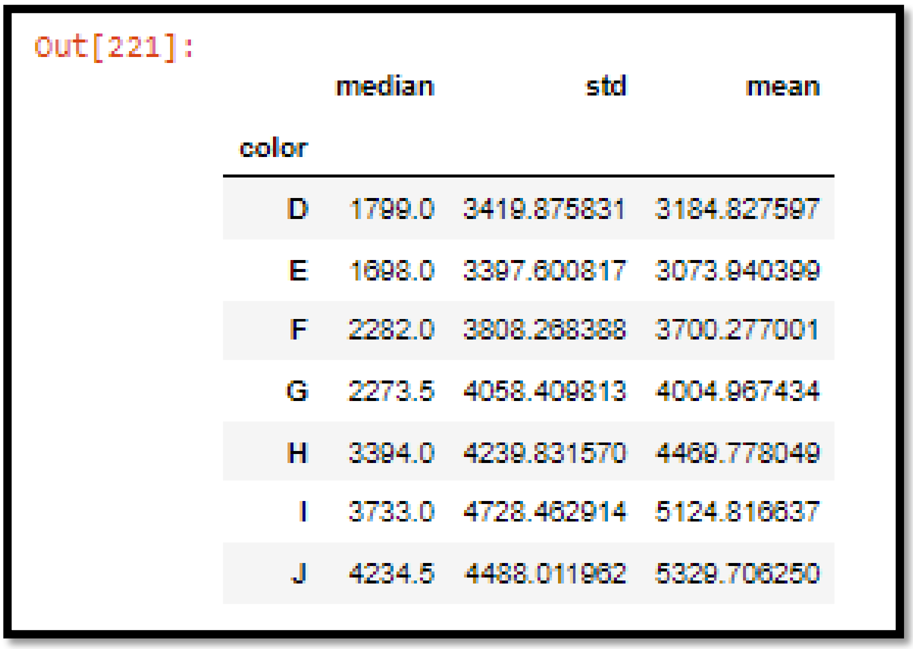


Out[201]:

cut	median	std	max	min	mean
Fair	3337.0	3747.642431	18574	369	4565.768935
Good	3092.5	3621.757943	18707	335	3927.074774
Ideal	1762.0	3869.198651	18804	326	3454.820639
Premium	3108.0	4315.682923	18795	326	4540.186192
Very Good	2633.0	4016.865952	18818	336	4032.267961

We can establish that there is an order in ranking like mean price is increasing from ideal then good to very good, premium and fair. Fair segment has the highest median value as well. Since, I can see the ordered ranking here, I have classified them using scale – Label encoding - and won't use one-hot encoding here. However, it is absolutely fine for us to go ahead and treat this variable using one-hot encoding as well. We can certainly try that approach if we would like to see the impact of varied kind of cut variables with price target variable. Overall, with the grid above, I don't think we will find cut as a strong predictor and hence I have stayed with using label encoding for the rest of the case study.

GS1 = GS1.replace({'Ideal':1,'Premium':4,'Very Good':3,'Good':2,'Fair':5})



Out[221]:

color	median	std	mean
D	1799.0	3419.875831	3184.827597
E	1698.0	3397.600817	3073.940399
F	2282.0	3808.268388	3700.277001
G	2273.5	4058.409813	4004.967434
H	3394.0	4239.831570	4469.778049
I	3733.0	4728.462914	5124.816637
J	4234.5	4488.011962	5329.706250

For me, its hard to pick an order and the variable seems to be having a direct impact on the price variable. We should read this further while creating our Linear Relationship model.
In one-hot encoding, the integer encoded variable is removed and a new binary variable is added for each unique integer value. A one hot encoding allows the representation of categorical data to be more expressive. Many machine learning algorithms cannot work with categorical data directly and hence, the categories must be converted into numbers. This is required for both input and output variables that are categorical. The only disadvantage being, that for high cardinality, the feature space can really blow up quickly and we will then need to struggle with dimensionality.

For this case study, I have used one-hot encoding for color and clarity independent variables.

We have now set the precedence that there are a lot of variables (Carat, x,y, and z) that are demonstrating strong correlation or multicollinearity. So, before proceeding with the Linear Regression Model creation, we need to get rid of them from the model creation exercise. I have decided to drop x,y, and z from my Linear Regression model creation step. I have decided to keep the carat column of the lot as it has the strongest relation with the target variable – Price out of the four columns. Depth column also did not have much impact on the price column and hence I have chosen to not use it as well.

Table 1.3.3: Heatmap after dropping x, y,z, and depth columns before creating Linear Regression model

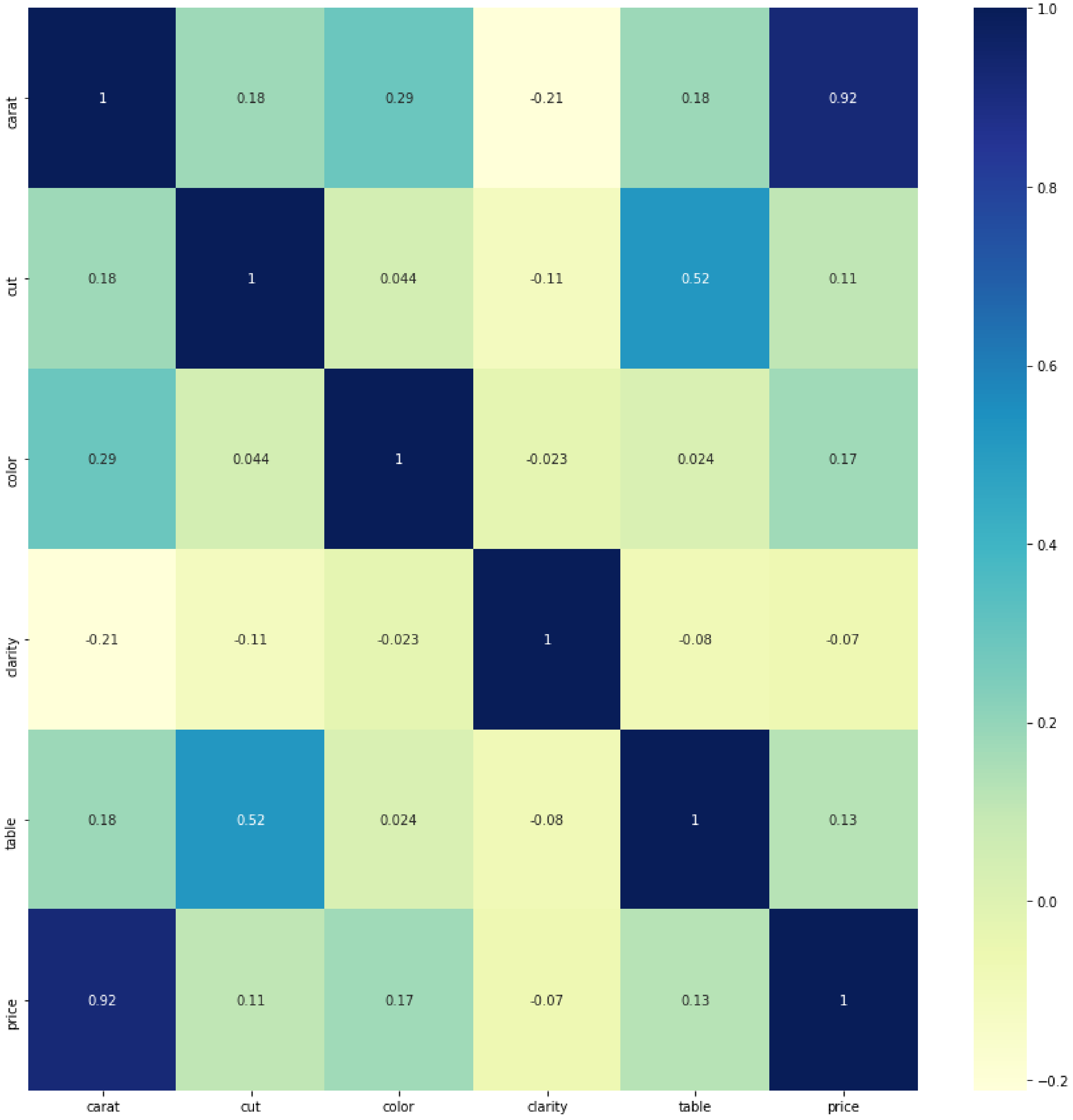


Fig.9 – Heatmap after dropping x, y,z, and depth columns before creating Linear Regression model

Q1.4 Inference: Basis on these predictions, what are the business insights and recommendations.

Answer -

We have a database which have strong correlation between independent variables and hence we need to tackle with the issue of multicollinearity which can hinder the results of the model performance. Multicollinearity makes it difficult to understand how one variable influence the target variable. However, it does not affect the accuracy of the model. As a result while creating the model, I had dropped a lot of independent variables displaying multicollinearity or the ones with no direct relation with the target variable.

While we looked at the data during univariate analysis, we were able to establish that Carat is strongly related with the price variable, and also with a lot of other independent variables - x, y, and z, and low correlation with variables such as table and cut as well. It can be established that Carat will be a strong predictor in our model creation.

The same trend was displayed even after the object columns were encoded. The carat variable continues to display strong to low correlation with most of the variables, making its claim to be the most important predictor firm.
Figure – Heatmap with encoded variables, before the model creation stage

After the Linear Regression model was created, we can see the assumption coming true as the carat variable emerged as the single biggest factor impacting the target variable, followed with a few others within clarity variables. Carat variable has the highest coefficient value as compared to the other studies variables for this test case.

If we are looking to fine-tune the model we can simply drop these columns from our Linear Regression model to see how the results pan out and can check our model performance.
As an alternate approach, we can choose to use the “cut” variable as one hot encoding or dummy encoding and run the model again to check on the overall model score or tackle the issue of multicollinearity. This can allow us to read the impact of cut variables on the target variable – Price, if company intends to study that as well.

However, for this case study and for the reasons mentioned above, I have not used one-hot encoding on the cut variable.
For the business based on the model that we have created for the test case, some of the key variables that are likely to positively drive price change are (top 5 in descending order):

1. Carat

2. Clarity_IF

3. Clarity VVS_1

4. Clarity VVS_2

5. Clarity_vs1

Recommendations:

As expected Carat is a strong predictor of the overall price of the stone.
Clarity refers to the absence of the Inclusions and Blemishes and has emerged as a strong predictor of price as well.
Clarity of stone types IF, VVS_1, VVS_2 and vs1 are helping the firm put an expensive price cap on the stones.
Color of the stones such H, I and J won't be helping the firm put an expensive price cap on such stones.
The company should instead focus on stones of color D, E and F to command relative higher price points and support sales.
This also can indicate that company should be looking to come up with new color stones like clear stones or a different color/unique color that helps impact the price positively.
The company should focus on the stone's carat and clarity so as to increase their prices. Ideal customers will also contribute to more profits. The marketing efforts can make use of educating customers about the importance of a better carat score and importance of clarity index. Post this, the company can make segments, and target the customer based on their income/paying capacity etc, which can be further studied.

PROBLEM 2

LOGISTIC REGRESSION AND LDA

You are hired by a tour and travel agency which deals in selling holiday packages. You are provided details of 872 employees of a company. Among these employees, some opted for the package and some didn't. You have to help the company in predicting whether an employee will opt for the package or not on the basis of the information given in the data set. Also, find out the important factors on the basis of which the company will focus on particular employees to sell their packages.

Dataset for Problem 2: Holiday_Package.csv

Data Dictionary:

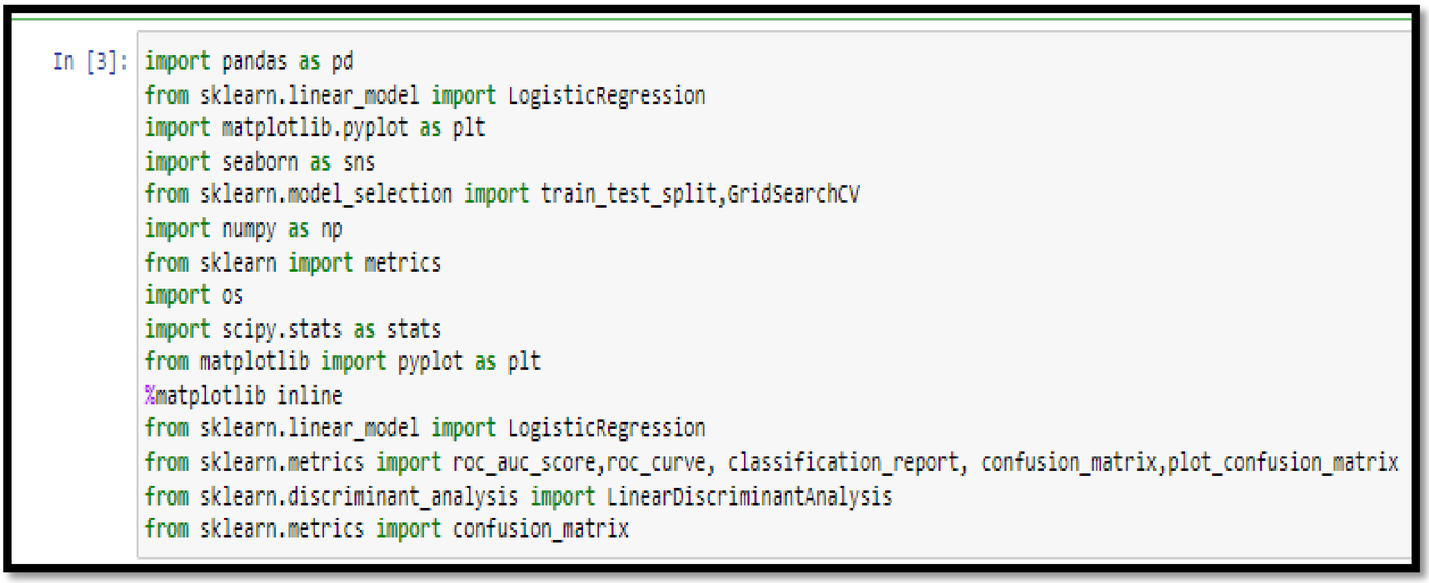
Variable Name	Description
Holiday_Package	Opted for Holiday Package yes/no?
Salary	Employee salary

age	Age in years
edu	Years of formal education
no_young_children	The number of young children (younger than 7 years)
no_older_children	Number of older children
foreign	foreigner Yes/No

Q.2.1 Data Ingestion: Read the dataset. Do the descriptive statistics and do null value condition check, write an inference on it. Perform Univariate and Bivariate Analysis. Do exploratory data analysis.

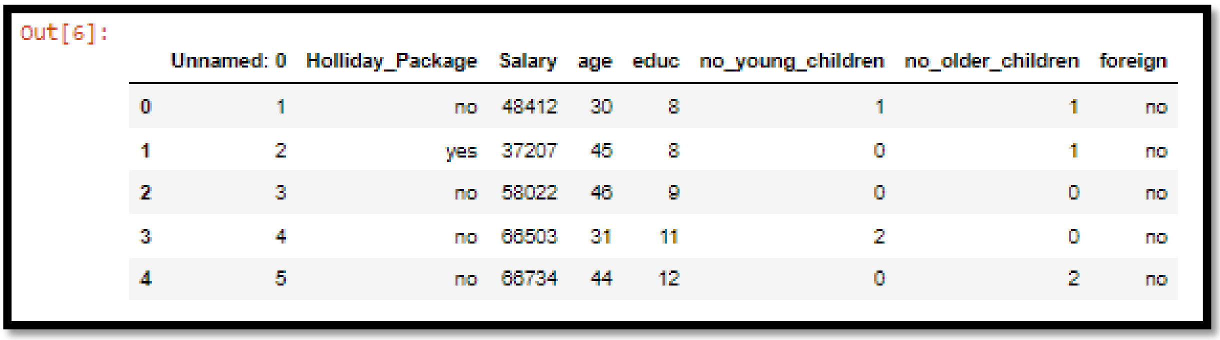
Answer –

Exploratory Data Analysis or (EDA) is understanding the data sets by summarizing their main characteristics often plotting them visually. This step is very important especially when we arrive at modelling the data. Plotting in EDA consists of Histograms, Box plot, pairplot and many more. It often takes much time to explore the data. Through the process of EDA, we can define the problem statement or definition on our data set which is very important.
Imported the required libraries.
To build the Logistic Regression Model and LDA on our dataset we need to import the following packages:



```
In [3]: import pandas as pd
        from sklearn.linear_model import LogisticRegression
        import matplotlib.pyplot as plt
        import seaborn as sns
        from sklearn.model_selection import train_test_split,GridSearchCV
        import numpy as np
        from sklearn import metrics
        import os
        import scipy.stats as stats
        from matplotlib import pyplot as plt
        %matplotlib inline
        from sklearn.linear_model import LogisticRegression
        from sklearn.metrics import roc_auc_score,roc_curve, classification_report, confusion_matrix,plot_confusion_matrix
        from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
        from sklearn.metrics import confusion_matrix
```

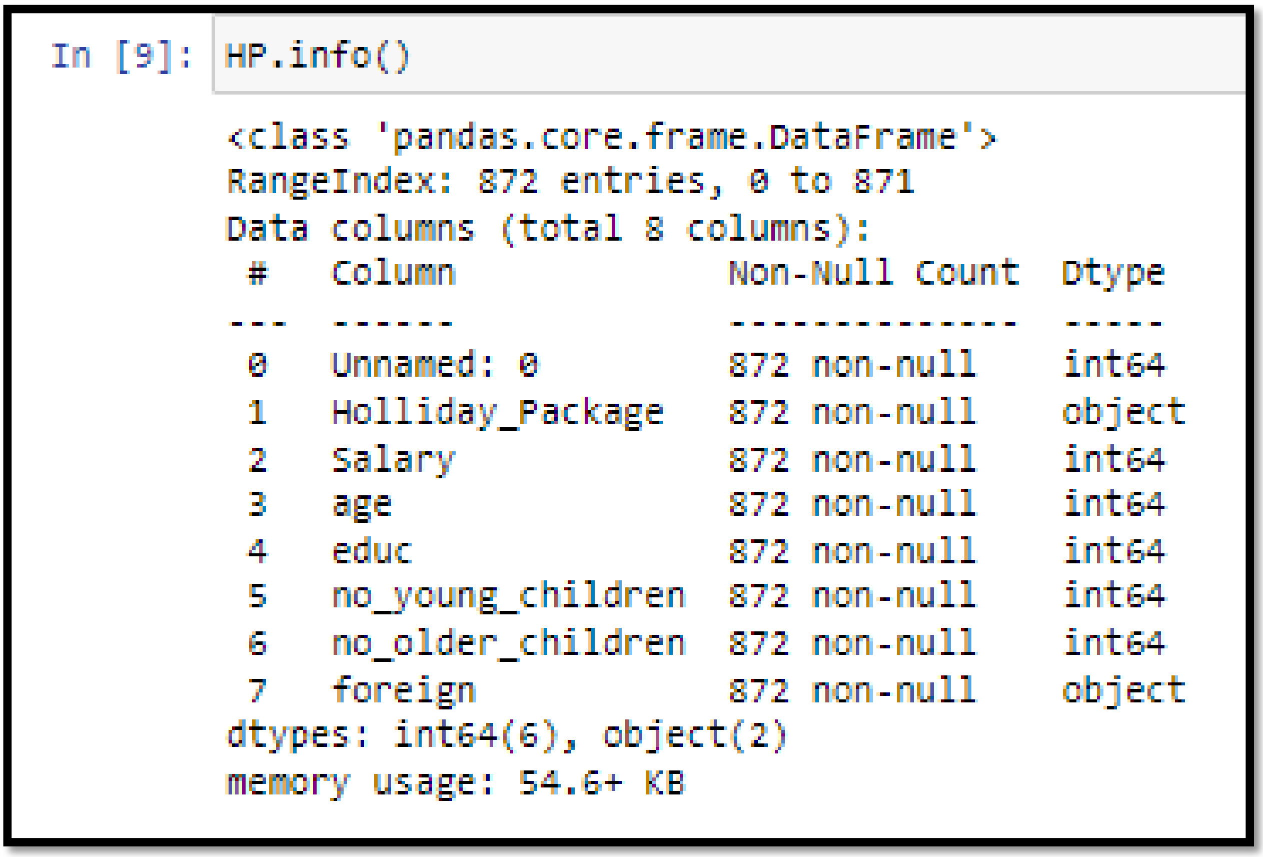
- ➢ Import the dataset – "Holiday_Package.csv"
- ➢ Data Summary and Exploratory Data Analysis.
  - ○ Checking if the data is being imported properly
  - ○ Head – The top 5 rows of the dataset are viewed using head () function



Out[6]:

	Unnamed: 0	Holliday_Package	Salary	age	educ	no_young_children	no_older_children	foreign
0	1	no	48412	30	8	1	1	no
1	2	yes	37207	45	8	0	1	no
2	3	no	58022	46	9	0	0	no
3	4	no	66503	31	11	2	0	no
4	5	no	66734	44	12	0	2	no

- Dimension of the Dataset: The Dimension or shape of the dataset can be shown using **shape** function. It shows that the dataset given to us has 872 rows and 8 columns or variables
- Structure of the Dataset : Structure of the dataset can be computed using **.info()** function

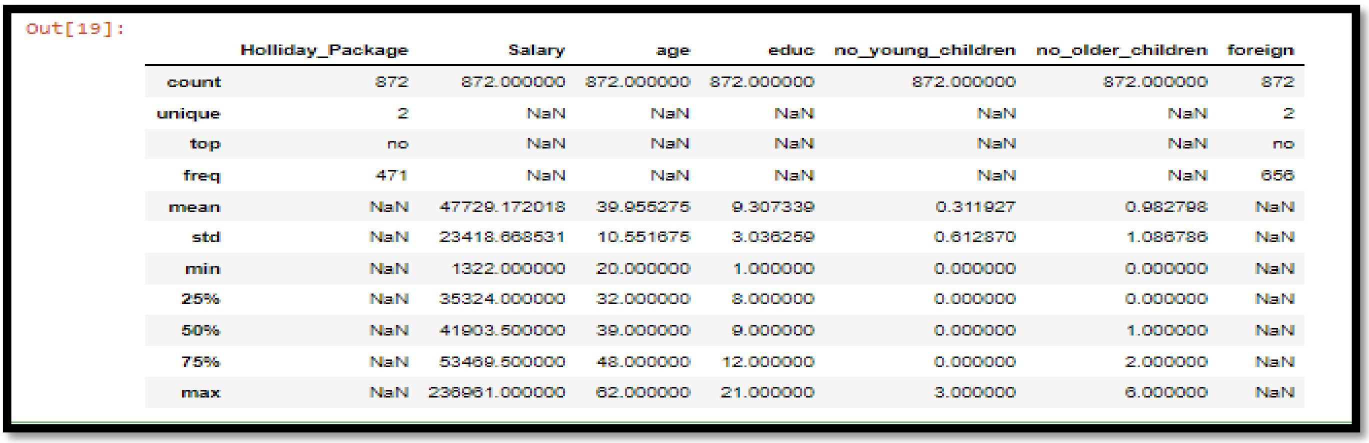


```
In [9]: HP.info()

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 872 entries, 0 to 871
        Data columns (total 8 columns):
         #   Column             Non-Null Count  Dtype
        ---  ------             --------------  -----
         0   Unnamed: 0         872 non-null    int64
         1   Holliday_Package   872 non-null    object
         2   Salary             872 non-null    int64
         3   age                872 non-null    int64
         4   educ               872 non-null    int64
         5   no_young_children  872 non-null    int64
         6   no_older_children  872 non-null    int64
         7   foreign            872 non-null    object
        dtypes: int64(6), object(2)
        memory usage: 54.6+ KB
```

- Data Type is – Integer/Object
- Checking for Duplicates: - There are No duplicate rows in the dataset as computed using. Duplicated () function.
- We have dropped the first column 'Unnamed: 0' column as this is not important for our study. The shape would be – 872 rows and 7 columns

Univariate Analysis

Descriptive Statistics for the dataset:



Out[19]:

	Holliday_Package	Salary	age	educ	no_young_children	no_older_children	foreign
count	872	872.000000	872.000000	872.000000	872.000000	872.000000	872
unique	2	NaN	NaN	NaN	NaN	NaN	2
top	no	NaN	NaN	NaN	NaN	NaN	no
freq	471	NaN	NaN	NaN	NaN	NaN	656
mean	NaN	47729.172018	39.955275	9.307339	0.311927	0.982798	NaN
std	NaN	23418.668531	10.551675	3.036259	0.612870	1.086786	NaN
min	NaN	1322.000000	20.000000	1.000000	0.000000	0.000000	NaN
25%	NaN	35324.000000	32.000000	8.000000	0.000000	0.000000	NaN
50%	NaN	41903.500000	39.000000	9.000000	0.000000	1.000000	NaN
75%	NaN	53469.500000	48.000000	12.000000	0.000000	2.000000	NaN
max	NaN	236961.000000	62.000000	21.000000	3.000000	6.000000	NaN

Summary of the Dataset

- Holiday Package – This variable is a categorical Variable and this will be our Target Variable.
- Salary, age, educ, no_young_children, no_older_children, variables are numerical or continuous variables.
- Salary ranges from 1322 to 236961. Average salary of employees is around 47729 with a standard deviation of 23418. Standard deviation indicates that the data is not normally distributed. Skew of 0.71 indicates that the data is right skewed and there are few employees earning more than an average of 47729. 75% of the employees are earning below 53469 while 25% of the employees are earning 35324.
- Age of the employee ranges from 20 to 62. Median is around 39. 25% of the employees are below 32 and 25% of the employees are above 48. Standard deviation is around 10. Standard deviation indicates almost normal distribution.
- Years of formal education ranges from 1 to 21 years. 25% of the population has formal education for 8 years, while the median is around 9 years. 75% of the employees have formal education of 12 years. Standard deviation of the education is around 3. This variable is also indicating skewness in the data
- Foreign is a categorical variable
- We have dropped the first column 'Unnamed: 0' column as this is not important for our study. Unnamed is a variable which has serial numbers so may not be required and thus it can be dropped for further analysisThe shape would be – 872 rows and 7 columns
- There are no null values
- There are no duplicates

Getting unique counts of Numerical Variables

- HP.Salary.value_counts()
- HP.age.value_counts().sort_index(ascending=False)
- HP.groupby(['Salary']).Salary.value_counts()
- HP.no_young_children.value_counts().sort_index(ascending=False)
- HP.no_older_children.value_counts().sort_index(ascending=False)
- HP.educ.value_counts().sort_index(ascending=False)

Checking the unique values for the target Variables 'Holiday Package":

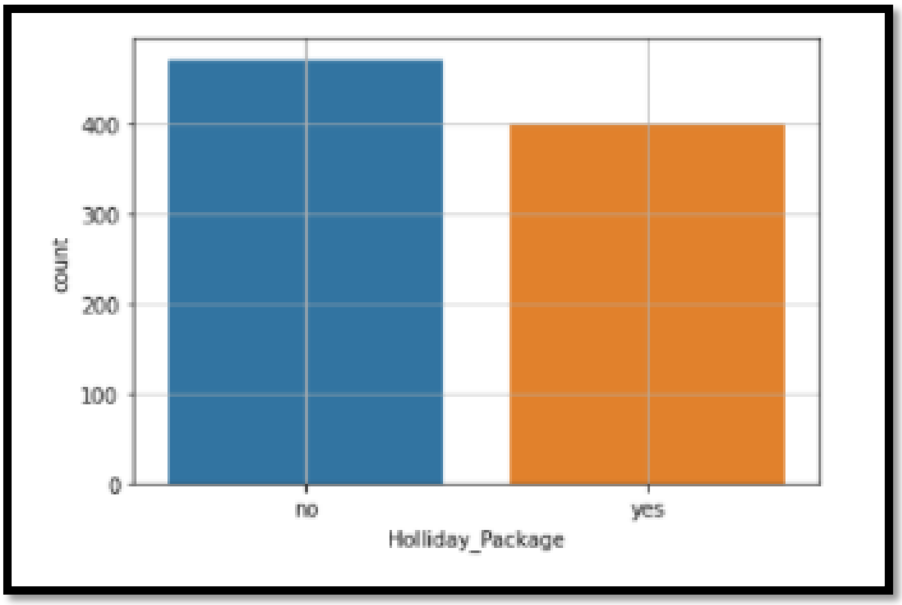


Fig.10 – Countplot on Holiday Package to check unique count

We can observe that 54% of the employees are not opting for the holiday package and 46% are interested in the package. This implies we have a dataset which is fairly balanced

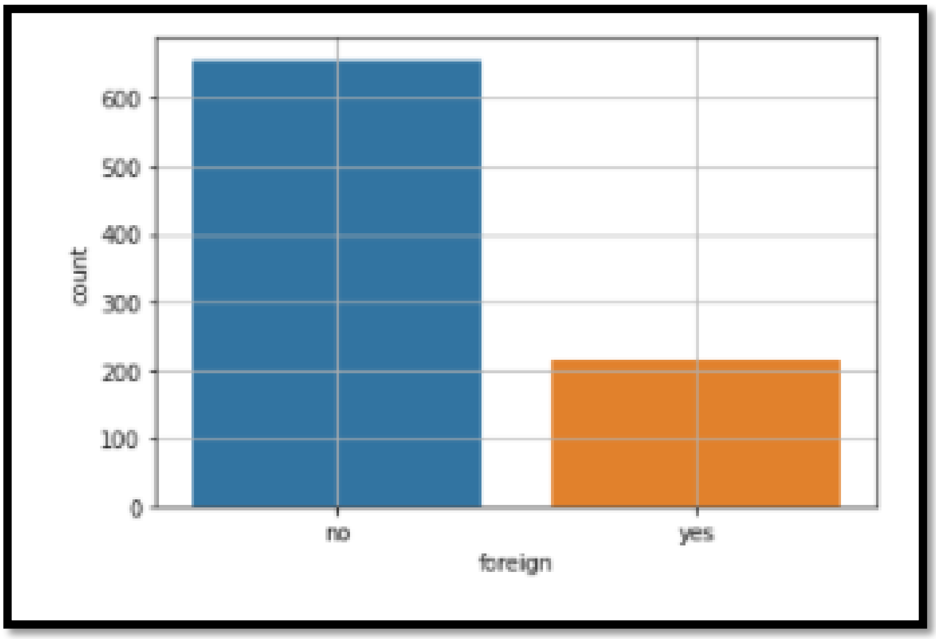


Fig.11 – Countplot on Foreign variable to check unique count

We can observe that 75% of the employees are not Foreigners and 25% are foreigners

- Checked for data distribution by plotting histogram

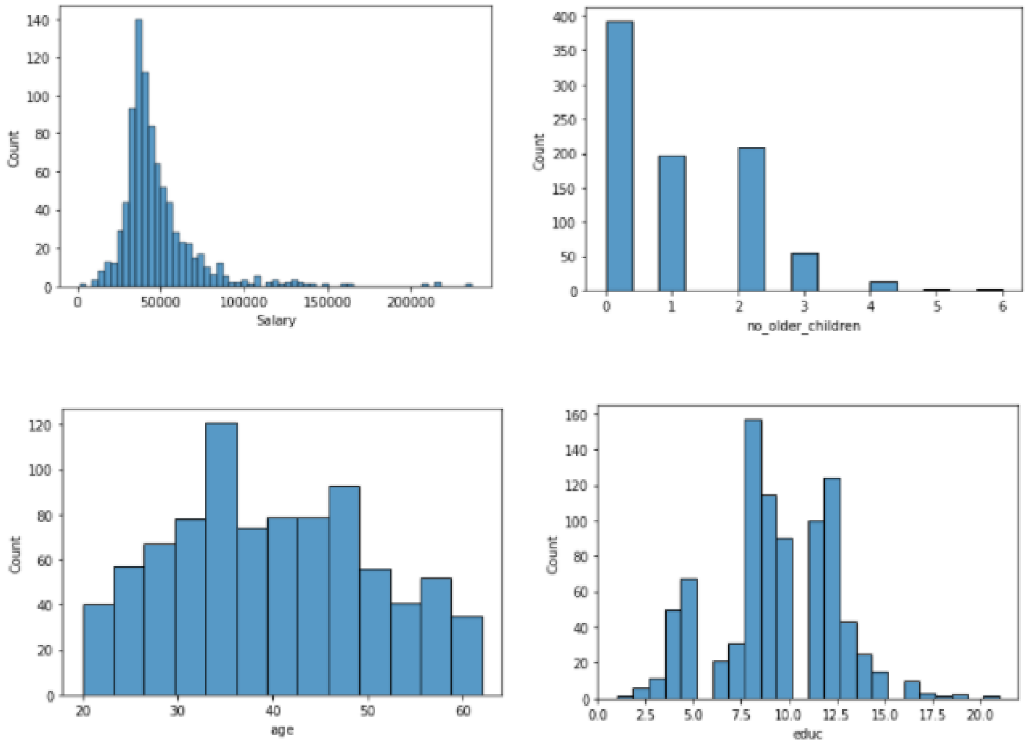
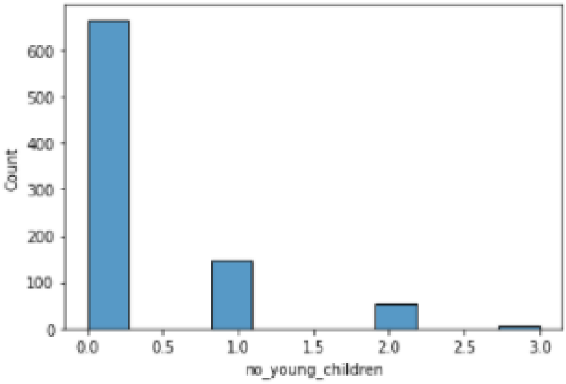


Fig.12 – Data distribution by plotting histogram



Let us check for outliers by plotting the box plot

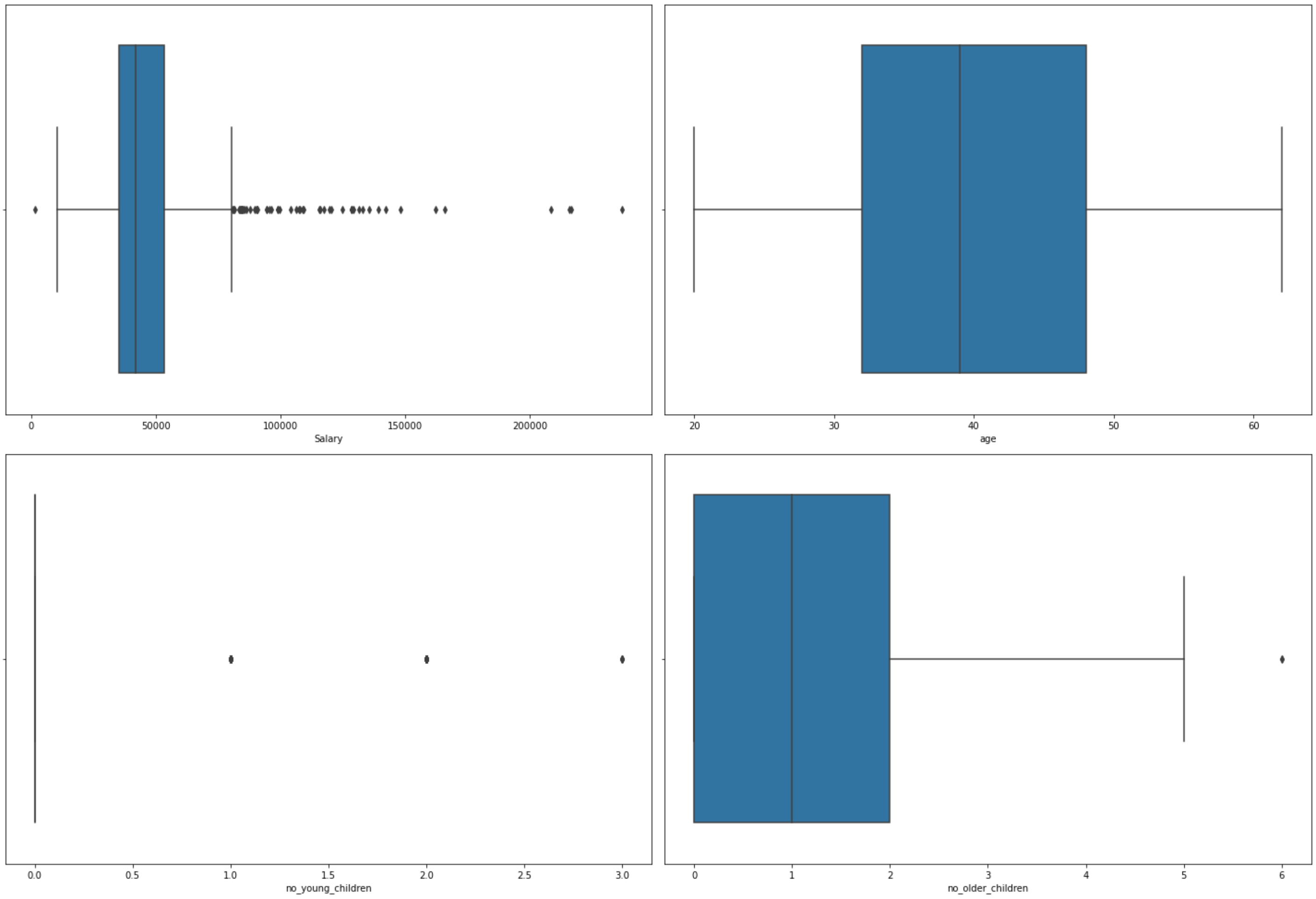


Fig.13 – Boxplot before outlier treatment (Problem2)

We can observer that there are significant outliers present in variable " Salary", however there are minimal outliers in other variables like 'educ', 'no. of young children' & 'no. of older children'. There are no outliers in variable 'age'. For Interpretation purpose we would need to study the variables such as no. of young children and no. of older children before outlier treatment. For this case study we have done outlier treatment for only salary & educ.

Treatment of outliers by IQR method (placed it here for comparison sake only)

Bi-Variate Analysis with Target variable

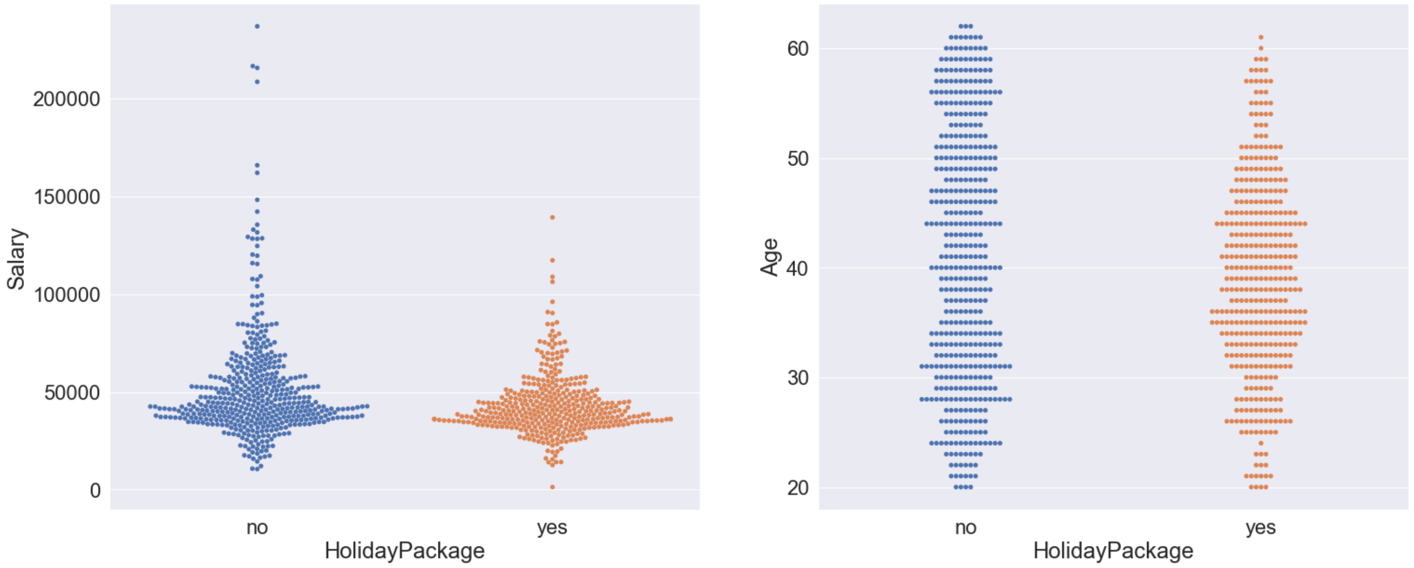


Fig.14 – Bi-Variate Analysis with Target variable (Problem-2)

We can see that:-

- As Salary increases to the max value, employees count increases for the not opting for the holiday package.
- As Age increases beyond 50 level, less employees opt for the holiday package

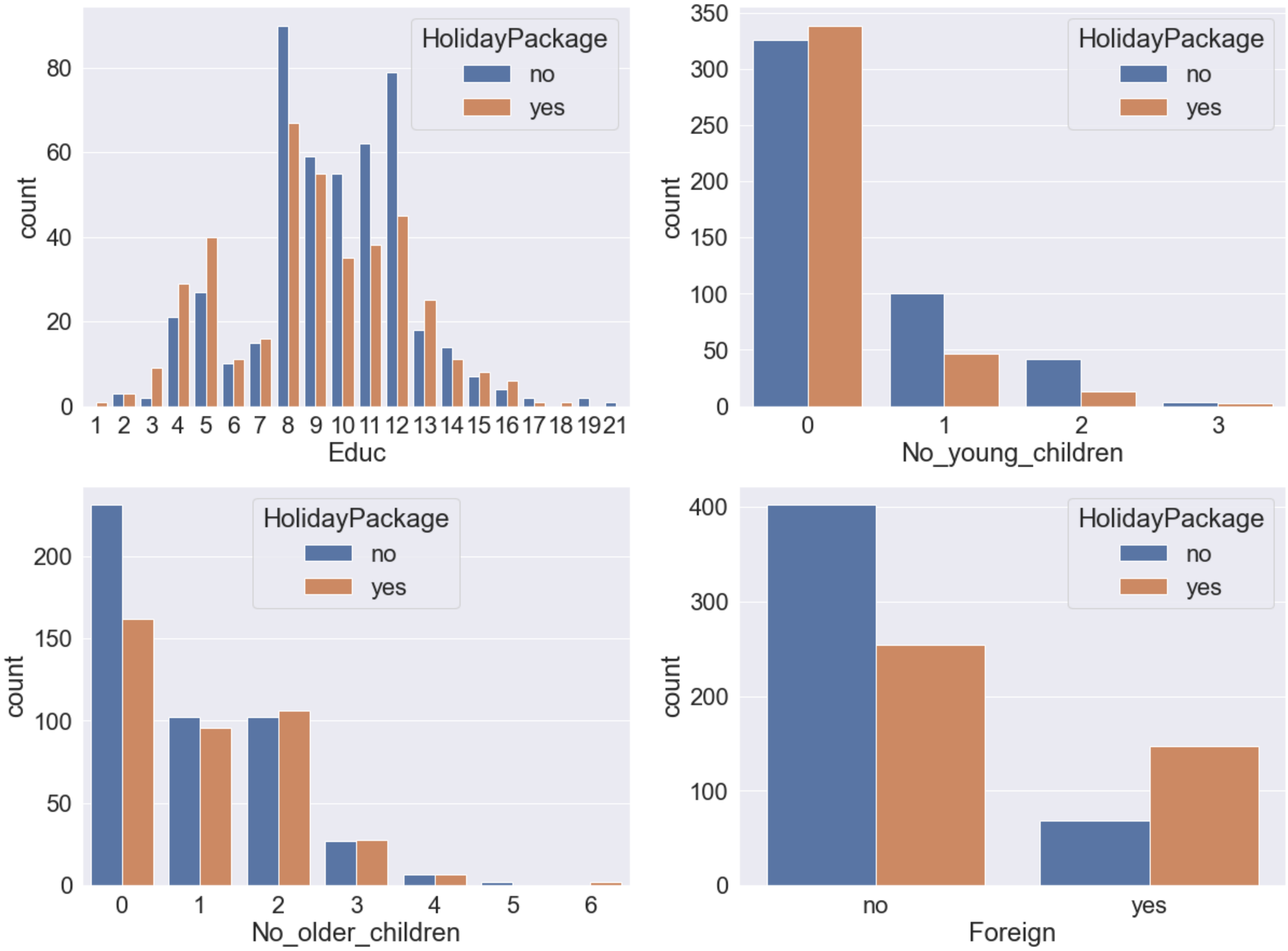
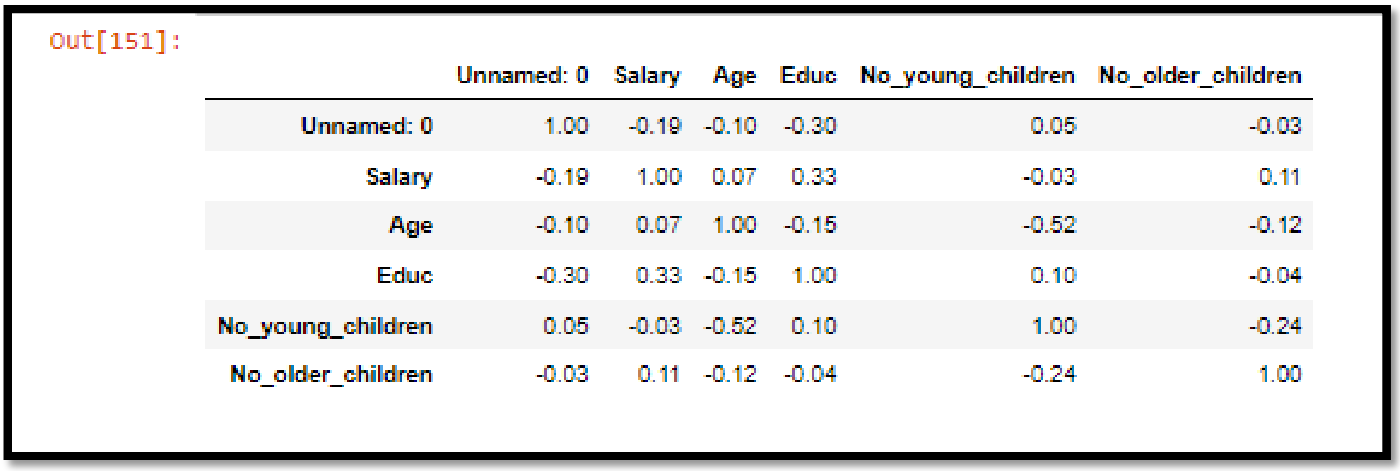


Fig.15 – Historical View on Independent Variable vs Dependent Variable

We can see observe from above graph that:-

- More Employees opt for Tours if their education level is 3,4,5,6,7,13,14,15,16
- Employees don't opt for tours if they have young child
- Older children count doesn't appears to have much impact on tour opted by employees or not
- Foreigner employees tends to opt more for the tour

Correlation matrix



Out[151]:

	Unnamed: 0	Salary	Age	Educ	No_young_children	No_older_children
Unnamed: 0	1.00	-0.19	-0.10	-0.30	0.05	-0.03
Salary	-0.19	1.00	0.07	0.33	-0.03	0.11
Age	-0.10	0.07	1.00	-0.15	-0.52	-0.12
Educ	-0.30	0.33	-0.15	1.00	0.10	-0.04
No_young_children	0.05	-0.03	-0.52	0.10	1.00	-0.24
No_older_children	-0.03	0.11	-0.12	-0.04	-0.24	1.00

Heat Map

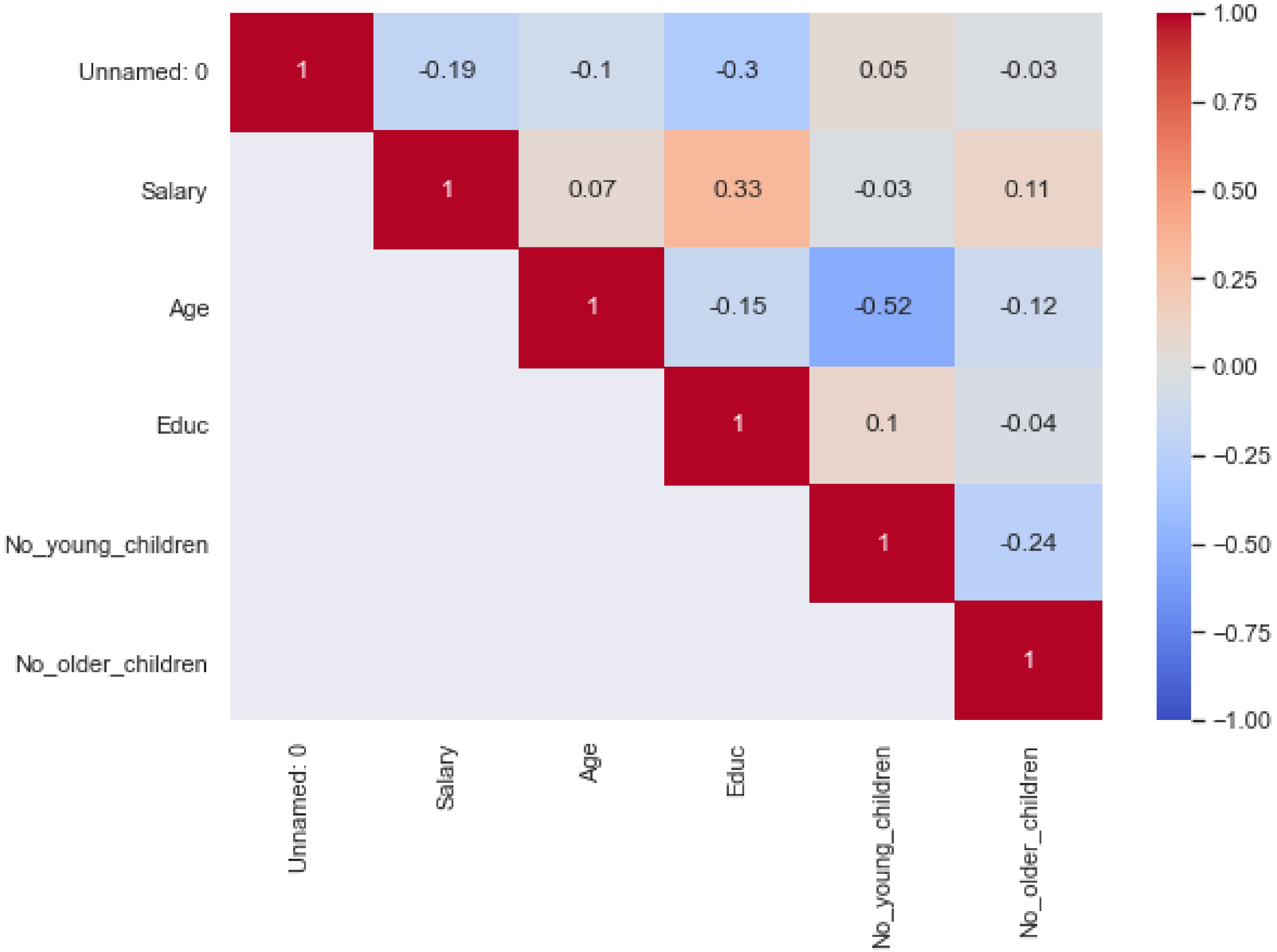


Fig.16 – HeatMap (Problem-2)

We can see in heatmap & correlation matrix that

- Salary has correlation with educ.
- Age is negatively correlated with No_young_children

Pairplots

To check the correlation of 2 columns in more detail we can draw pairplots/Scatter Diagram

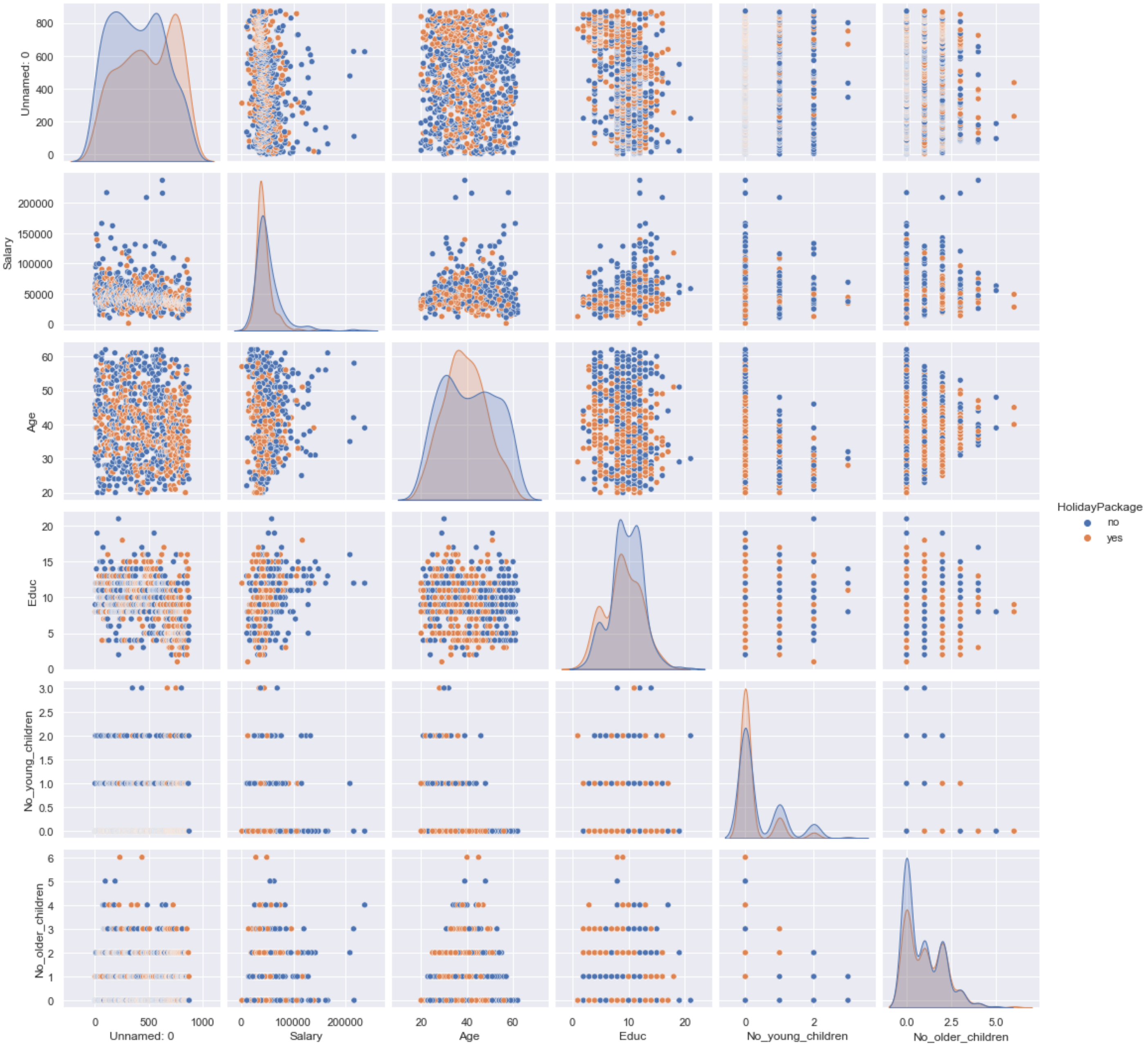


Fig.17 – Pairplots/Scatter Diagram (Problem-2)

As depicted in heat map of correlation matrix, we can see that no of young children negatively correlated with age.

Salary is slightly correlated with Educ

Q.2.2 Do not scale the data. Encode the data (having string values) for Modelling. Data Split: Split the data into train and test (70:30). Apply Logistic Regression and LDA (linear discriminant analysis).

Answer -

While Linear Regression helps us predicting continuous target variable, Logistic Regression helps us for predicting a discrete a target variable. Logistic Regression is one of the "white-box" algorithms which helps us in determining the probability values and the corresponding cut-offs. Logistic regression is used to solve such problem which gives us the corresponding probability outputs and then we can decide the appropriate cut-off points to get the target class outputs.

Precisely Logistic Regression is defined as a statistical approach, for calculating the probability outputs for the target labels. In its basic form, it is used to classify binary data. Logistic regression is very much similar to linear regression where the explanatory variables(X) are combined with weights to predict a target variable of binary class(y).

Evaluation of Logistic regression model- Performance measurement of classification algorithms are judge by confusion matrix which comprise the classification count values of actual and predicted labels.

Pros and cons of Logistics Regression:

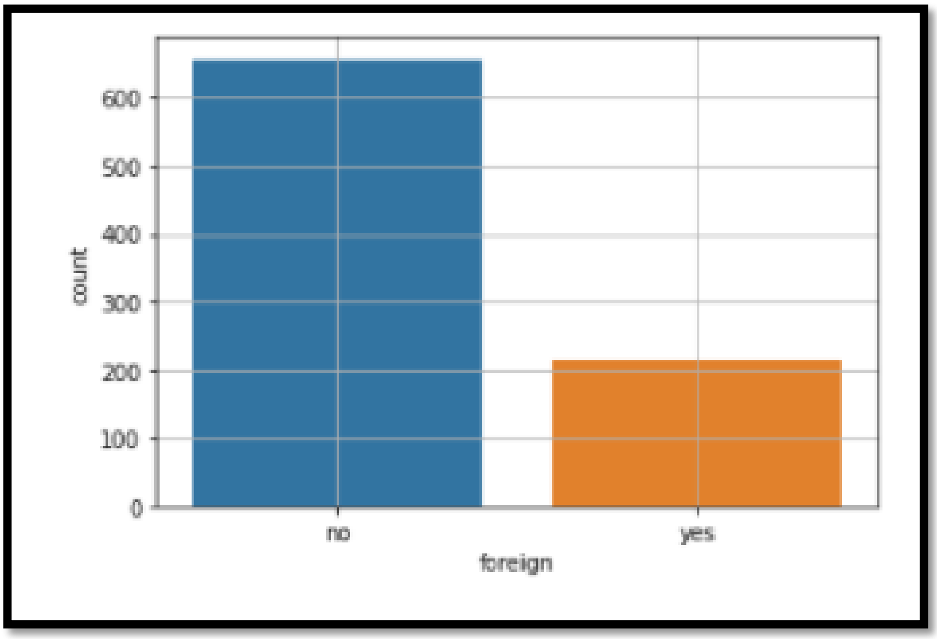
Pros- Logistic regression classification model is simple and easily scalable for multiple classes.

Cons- Classifier constructs linear boundaries and the interpretation of coefficients value is difficult.

In the given dataset, the target variable – Holliday Package and an independent variable – Foreign are object variables. Let us study them one at a time.

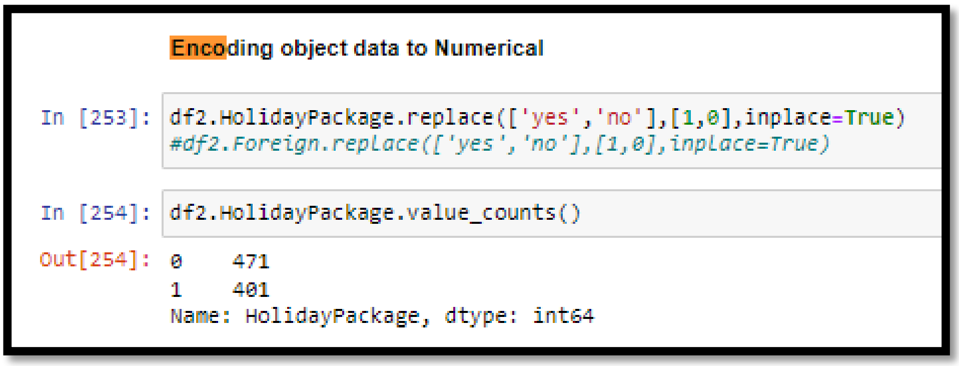
Holliday_Package: The distribution seems to be fine, with 54% for no and 46% for yes.

Foreign: The data is imbalanced with more skewed towards no and relatively a smaller shared for yes.



Both the variables can be encoded into numerical values for model creation analytical purposes.

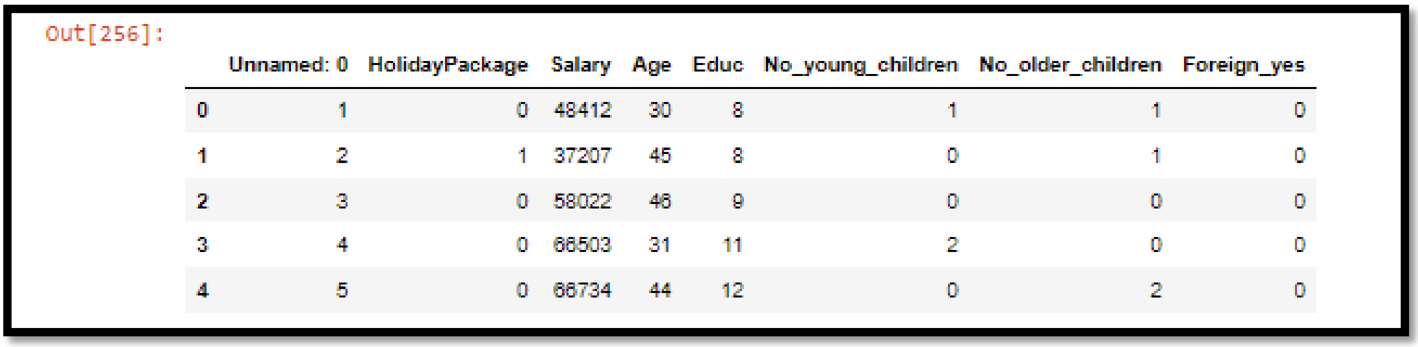
Table 2.2.1: Encoding the string values



Encoding object data to Numerical

```
In [253]: df2.HolidayPackage.replace(['yes','no'],[1,0],inplace=True)
          #df2.Foreign.replace(['yes','no'],[1,0],inplace=True)

In [254]: df2.HolidayPackage.value_counts()

Out[254]: 0    471
          1    401
          Name: HolidayPackage, dtype: int64
```

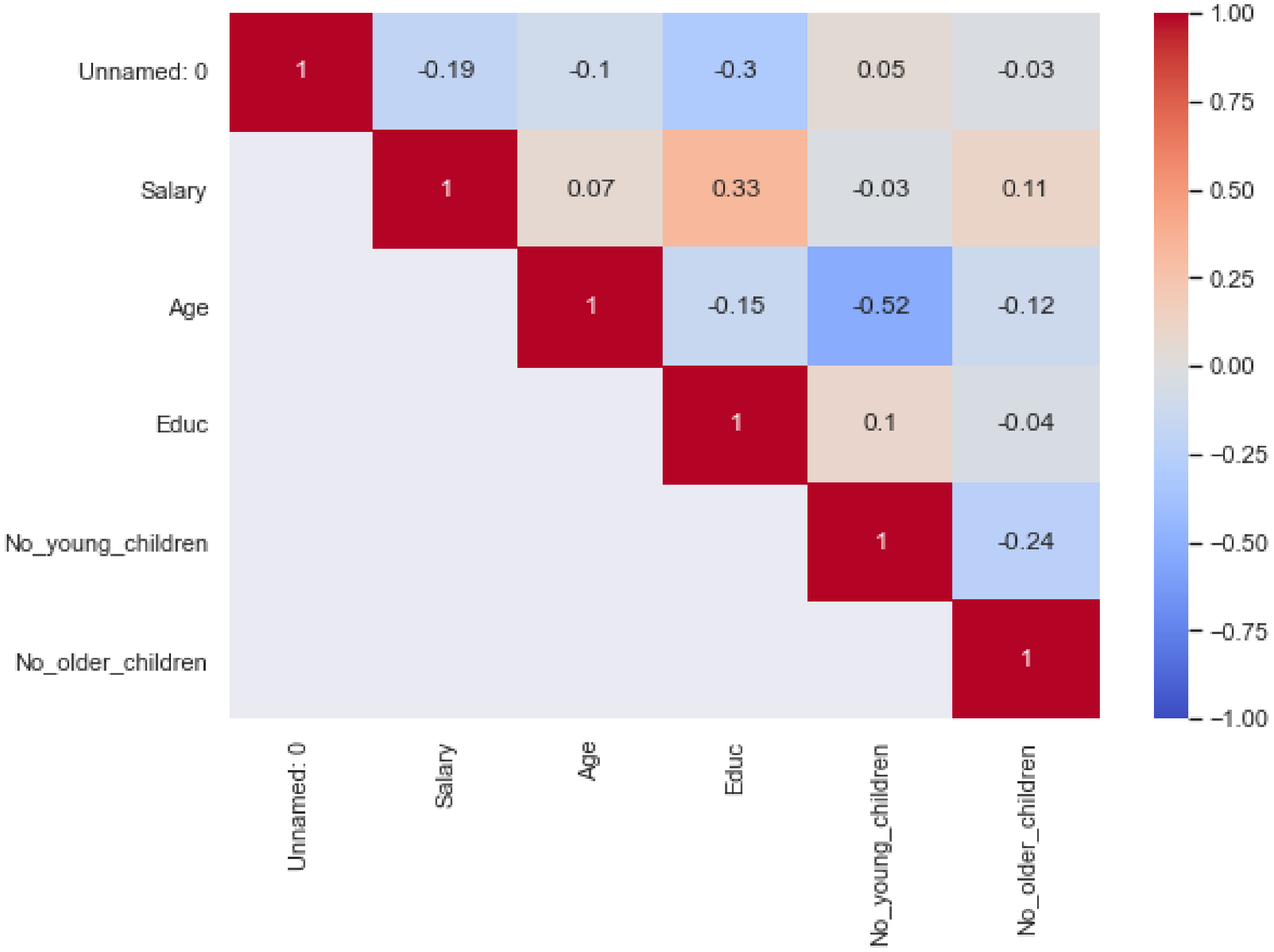
Table 2.2.2: Checking if the values are converted into numeric using the head()



Out[256]:

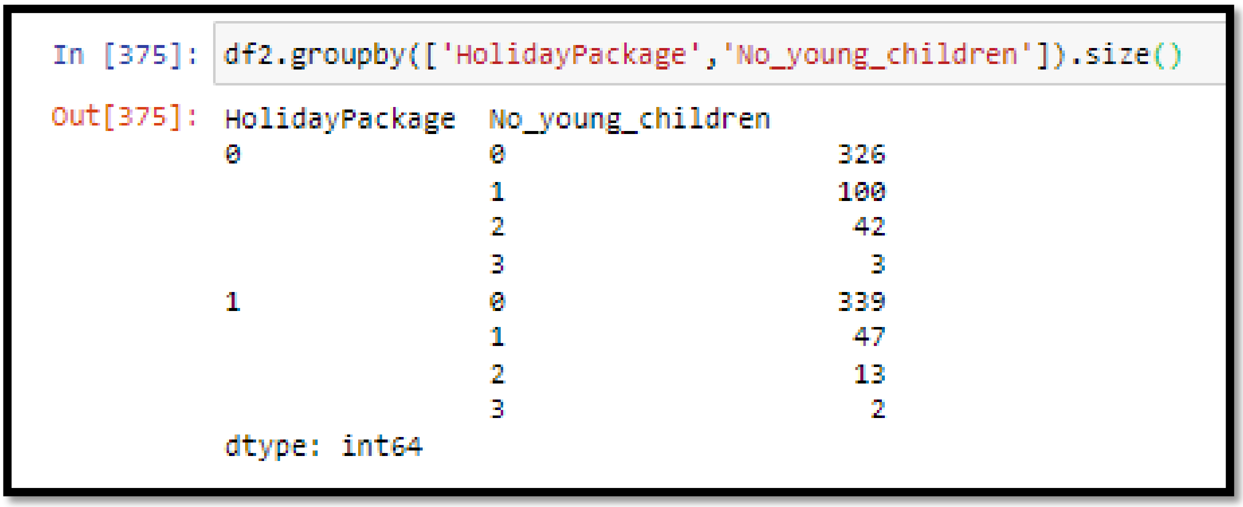
	Unnamed: 0	HolidayPackage	Salary	Age	Educ	No_young_children	No_older_children	Foreign_yes
0	1	0	48412	30	8	1	1	0
1	2	1	37207	45	8	0	1	0
2	3	0	58022	46	9	0	0	0
3	4	0	66503	31	11	2	0	0
4	5	0	66734	44	12	0	2	0

Before we proceed with the model creation, let us read the other part of the data to see how the numerical data also impacts the model. I will first look at the data correlation to quickly identify the variable importance using the heatmap



We can relate there isn't any strong correlation between any variables. Salary and education display moderate corelation and no_older_children is somewhat correlated with salary variable. However, there are no strong correlation in the data set.

Looking at the no_young_children variable:



```
In [375]: df2.groupby(['HolidayPackage','No_young_children']).size()

Out[375]: HolidayPackage  No_young_children
          0               0                    326
                          1                    100
                          2                     42
                          3                      3
          1               0                    339
                          1                     47
                          2                     13
                          3                      2
          dtype: int64
```

As next steps, we will initiate the LogisticRegression function and will then fit the Logistic Regression model. There after we will predict on the training and test data set.

Building Logistic Regression Model

Initially we built the logistic regression model using sklearn as per the following hyper parameters :-

- solver='newton-cg',
- max_iter=10000,
- penalty='none',
- verbose=True,
- n_jobs=2

Model Evaluation

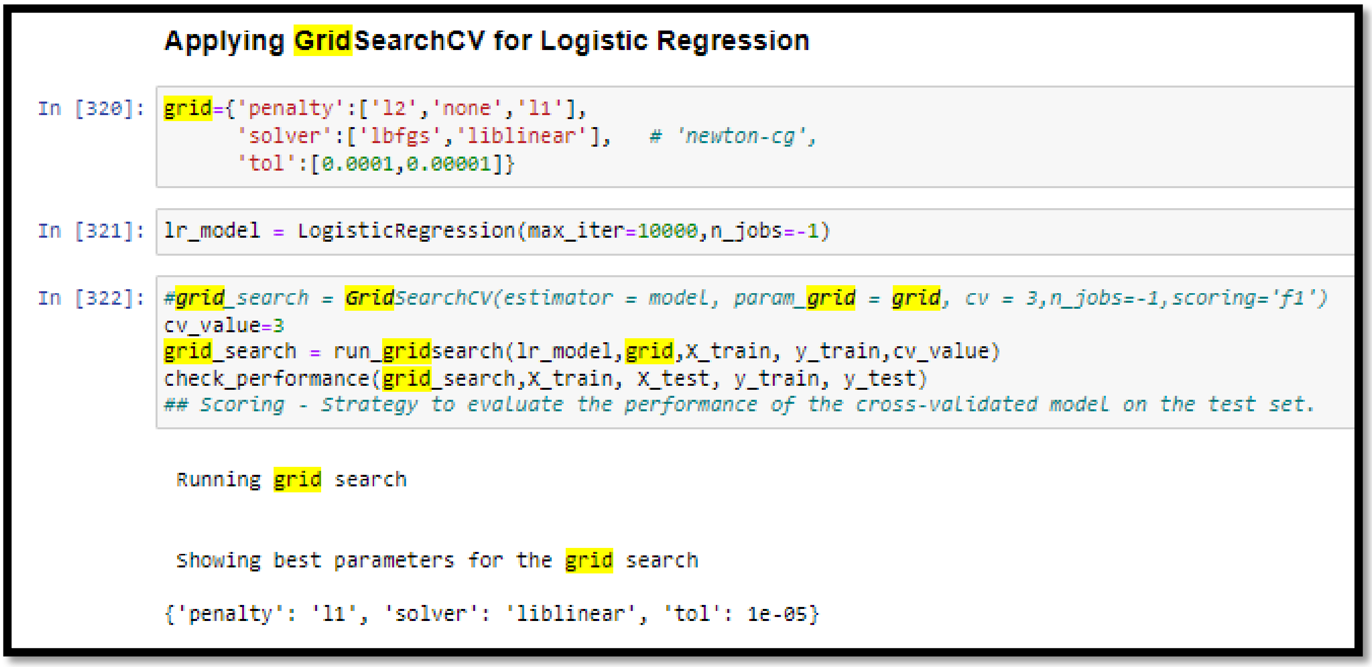
Accuracy score for train and test data set is as shown below:-

```
Model score for training dataset 0.6655737704918033
Model score for training dataset 0.6679389312977099
```

The accuracy scores aren't too different and can be considered as right fit models avoiding the scenarios of underfit and overfit models.

We can apply for GridSearchCV here to finetune the model further to see if helps improve the results. GridSearchCV is a brute force iterative method of obtaining the best model based on a scoring metric provided by us and the parameters provided.

We can pass on the parameters to see what the best set as prescribed by the model is. A glimpse of the steps taken are shared below:



Applying GridSearchCV for Logistic Regression

```
In [320]: grid={'penalty':['l2','none','l1'],
                'solver':['lbfgs','liblinear'],    # 'newton-cg',
                'tol':[0.0001,0.00001]}

In [321]: lr_model = LogisticRegression(max_iter=10000,n_jobs=-1)

In [322]: #grid_search = GridSearchCV(estimator = model, param_grid = grid, cv = 3,n_jobs=-1,scoring='f1')
          cv_value=3
          grid_search = run_gridsearch(lr_model,grid,X_train, y_train,cv_value)
          check_performance(grid_search,X_train, X_test, y_train, y_test)
          ## Scoring - Strategy to evaluate the performance of the cross-validated model on the test set.

           Running grid search


           Showing best parameters for the grid search

          {'penalty': 'l1', 'solver': 'liblinear', 'tol': 1e-05}
```

AUC and ROC for the training data & test data

```
AUC for Train dataset: 0.734
AUC for test dataset: 0.734
```

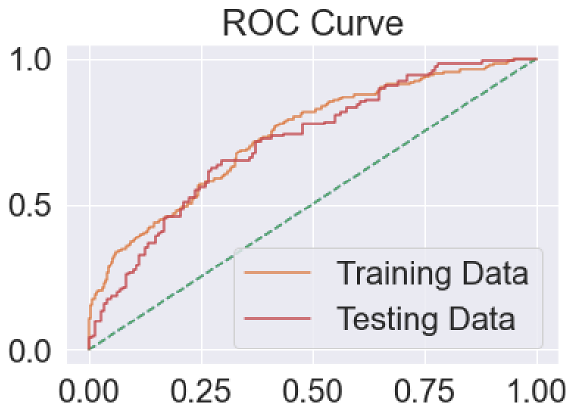


Fig.18 – ROC Curve on Training & Testing Data

LDA takes the help of prior probabilities to predict the corresponding target probabilities. Prior probabilities is the probability of y (say equal to 1) without taking into account any other data or variables. The corresponding updated probabilities when the covariates (Xs) are available is called the posterior probabilities. We want to find P(Y=1|X). Thus, a Linear Discriminant Analysis (LDA) discriminates between the two classes by looking at the features (Xs)
Like Logistics regression, here also we will make a model and fit it. Post which, we can evaluate the accuracy scores.

Confusion Matrix for the training data and testing data

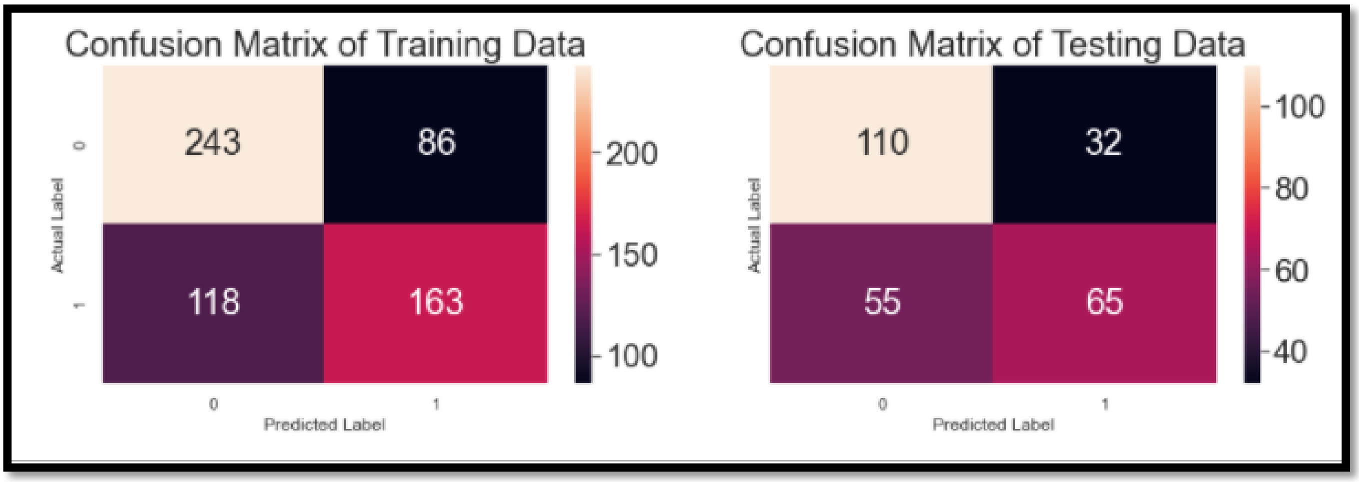
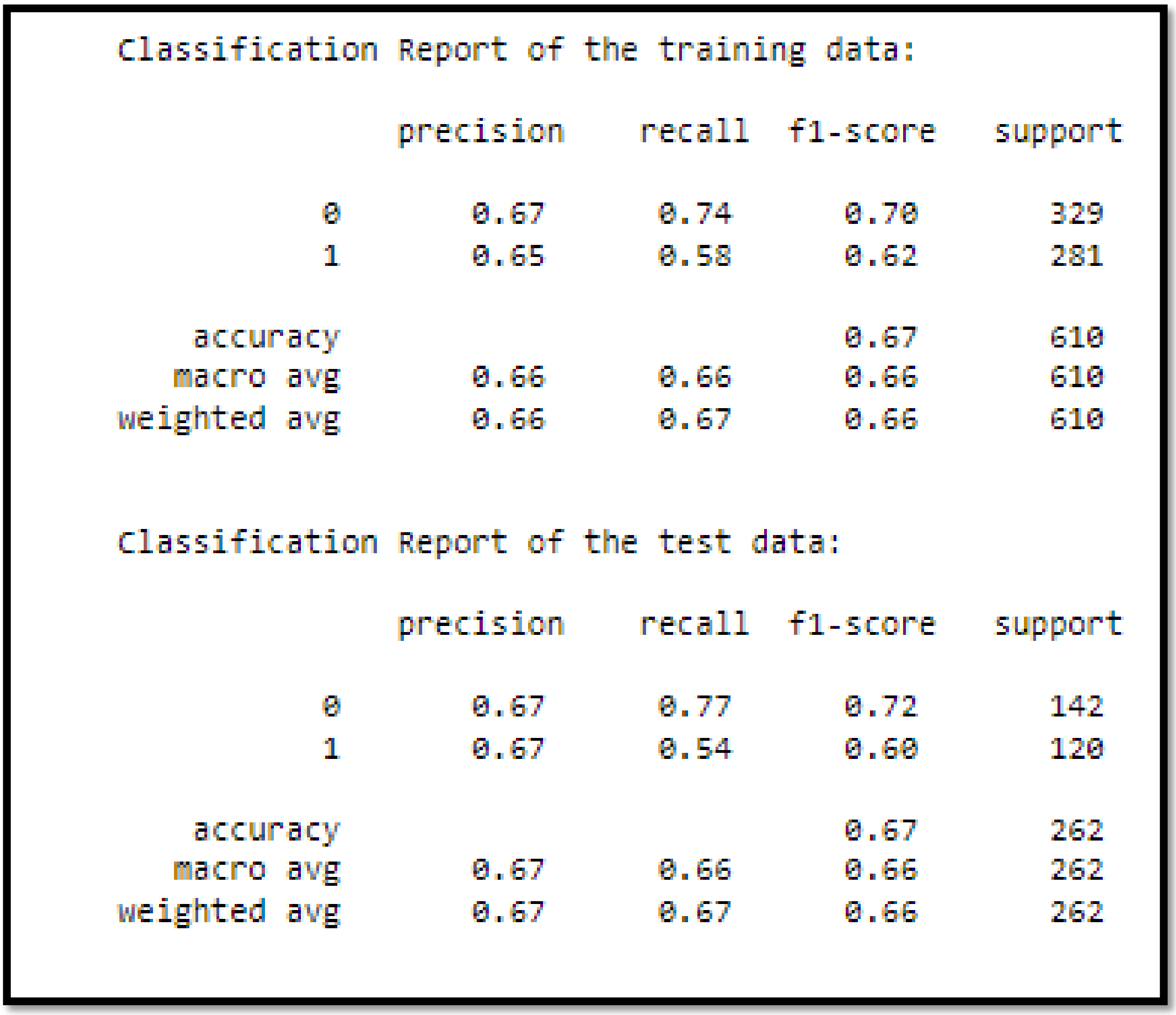


Fig.19 – Confusion Matrix Training & Testing Data

Training Data and Test Data Classification Report Comparison



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0       0.67      0.74      0.70       329
           1       0.65      0.58      0.62       281

    accuracy                           0.67       610
   macro avg       0.66      0.66      0.66       610
weighted avg       0.66      0.67      0.66       610


Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.67      0.77      0.72       142
           1       0.67      0.54      0.60       120

    accuracy                           0.67       262
   macro avg       0.67      0.66      0.66       262
weighted avg       0.67      0.67      0.66       262
```

Applying GridSearchCV for Logistic Regression

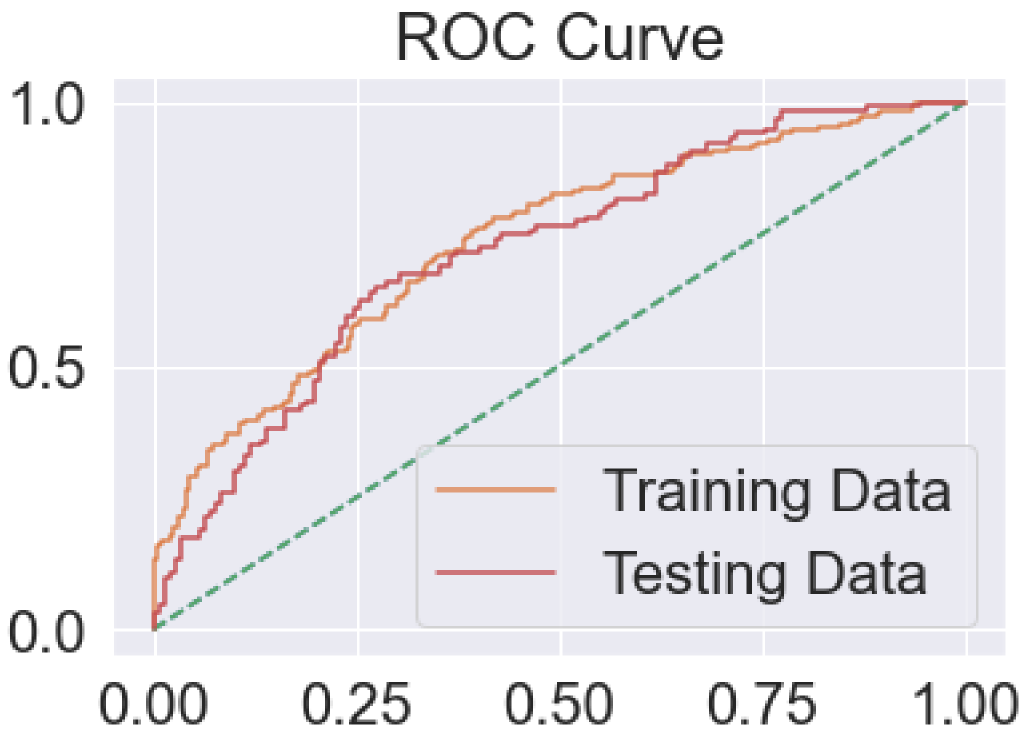
```
Showing best parameters for the grid search

{'penalty': 'l1', 'solver': 'liblinear', 'tol': 1e-05}
```

Model Evaluation

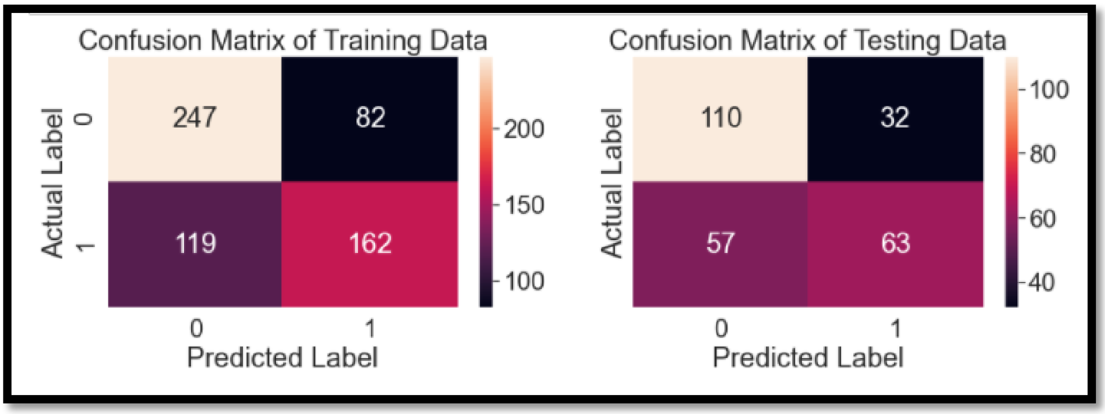
```
Model score for training dataset 0.6704918032786885
Model score for training dataset 0.6603053435114504
```

AUC and ROC for the training data & test data

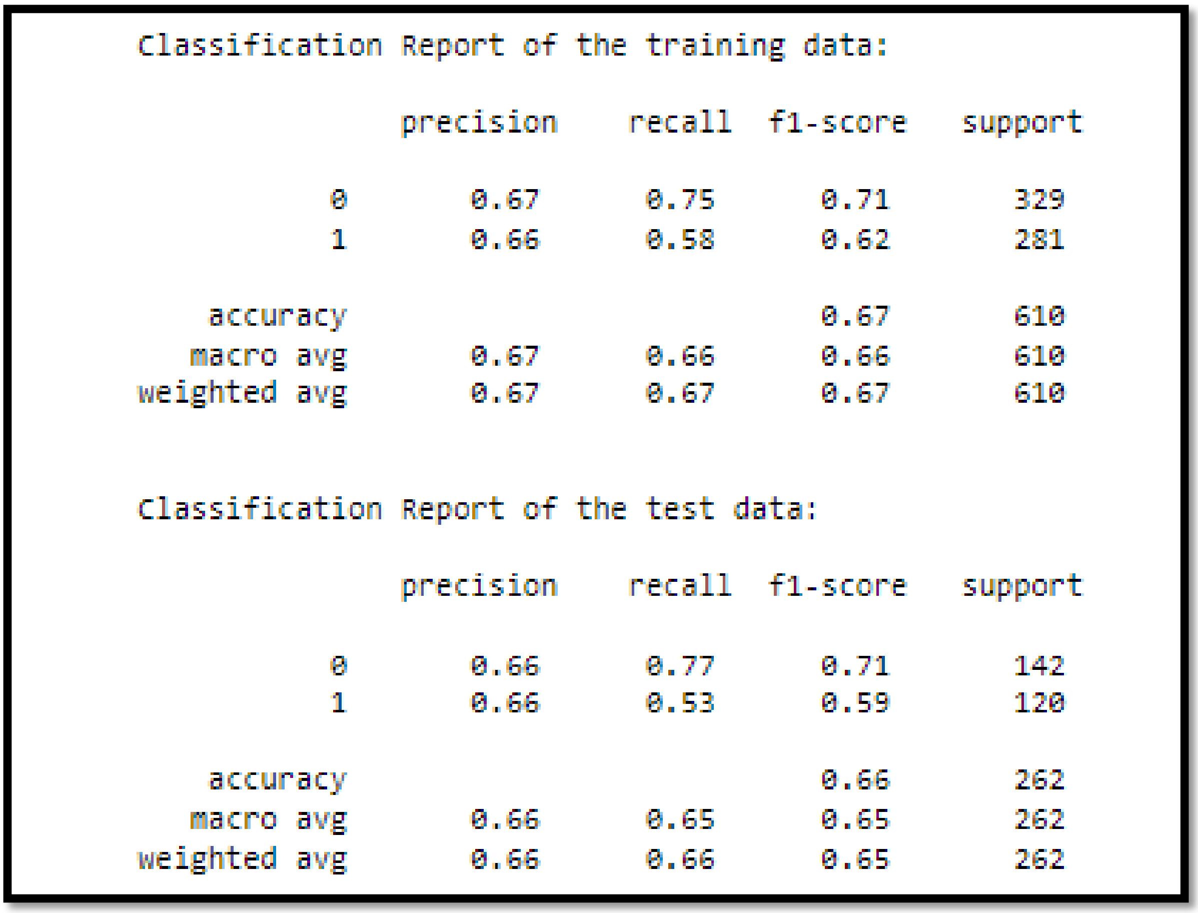
```
AUC for Train dataset: 0.735
AUC for test dataset: 0.735
```



Confusion Matrix for the training data and testing data



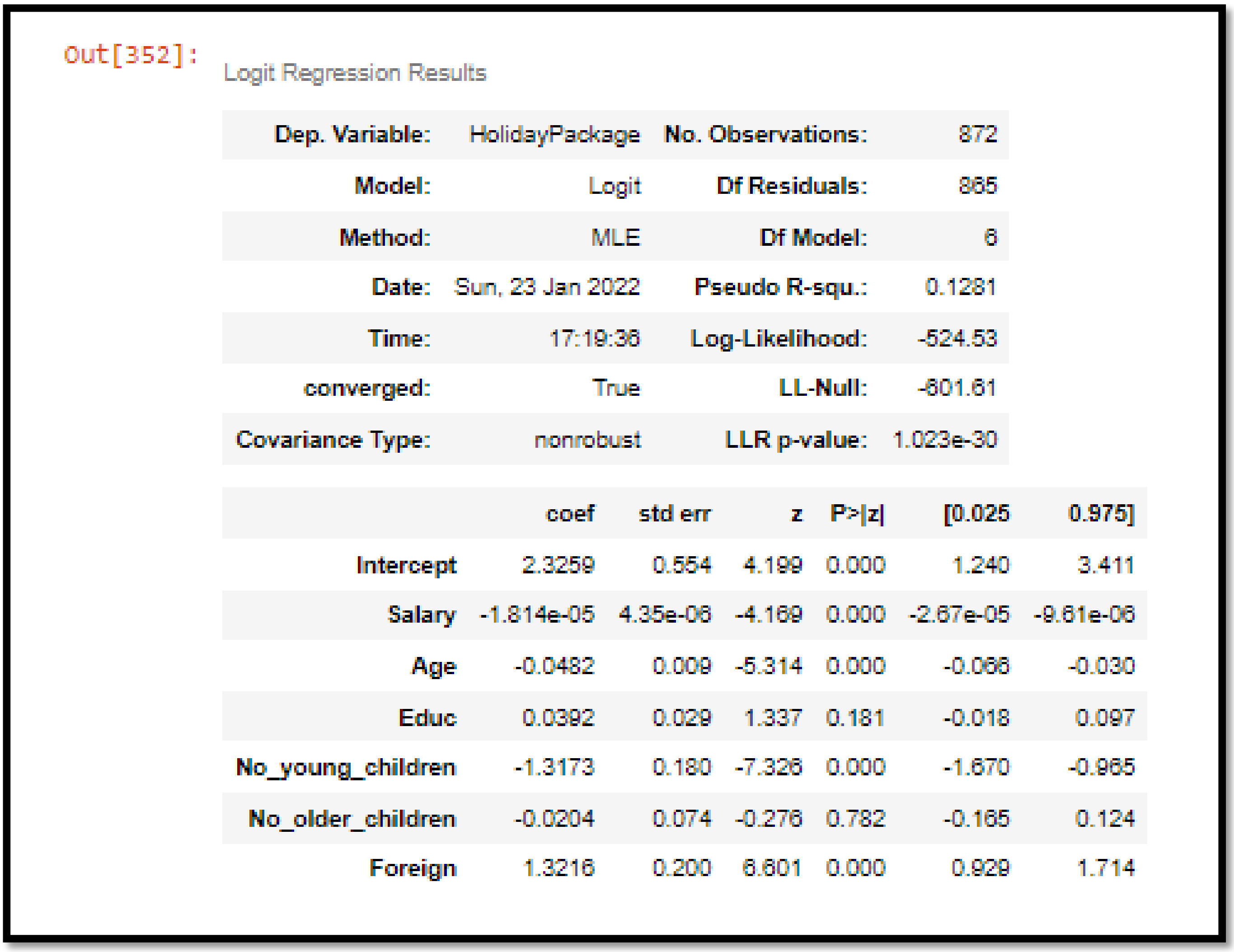
Training Data and Test Data Classification Report Comparison



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0       0.67      0.75      0.71       329
           1       0.66      0.58      0.62       281

    accuracy                           0.67       610
   macro avg       0.67      0.66      0.66       610
weighted avg       0.67      0.67      0.67       610


Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.66      0.77      0.71       142
           1       0.66      0.53      0.59       120

    accuracy                           0.66       262
   macro avg       0.66      0.65      0.65       262
weighted avg       0.66      0.66      0.65       262
```

Getting the equation

Using statsmodel, we can find the equation of log odds and we can find which coefficient has the more weightage in deciding the target response variable



Out[352]:

Logit Regression Results

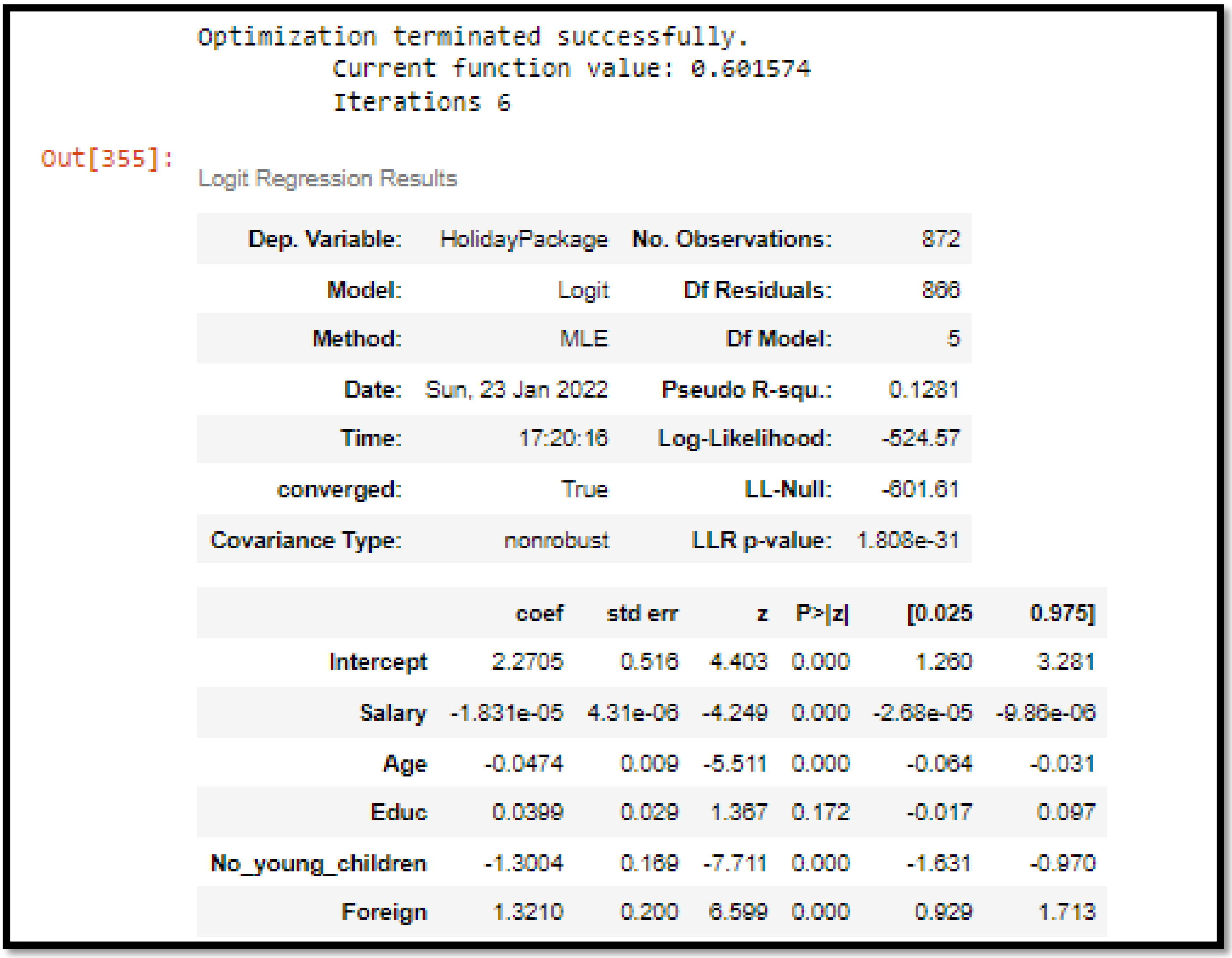
Dep. Variable:	HolidayPackage	No. Observations:	872
Model:	Logit	Df Residuals:	865
Method:	MLE	Df Model:	6
Date:	Sun, 23 Jan 2022	Pseudo R-squ.:	0.1281
Time:	17:19:36	Log-Likelihood:	-524.53
converged:	True	LL-Null:	-601.61
Covariance Type:	nonrobust	LLR p-value:	1.023e-30

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	2.3259	0.554	4.199	0.000	1.240	3.411
Salary	-1.814e-05	4.35e-06	-4.169	0.000	-2.67e-05	-9.61e-06
Age	-0.0482	0.009	-5.314	0.000	-0.066	-0.030
Educ	0.0392	0.029	1.337	0.181	-0.018	0.097
No_young_children	-1.3173	0.180	-7.326	0.000	-1.670	-0.965
No_older_children	-0.0204	0.074	-0.276	0.782	-0.165	0.124
Foreign	1.3216	0.200	6.601	0.000	0.929	1.714

We can see that the p value of No_older_children is the highest (.733) , followed by Educ(.241) and it is greator than 0.05.

Hence it confirms that No_older_children and Educ attribute has no impact on dependent variable HolidayPackage

Removing these 2 columns, we ran the model again to get the following report :-



```
Optimization terminated successfully.
         Current function value: 0.601574
         Iterations 6
```

Out[355]:

Logit Regression Results

Dep. Variable:	HolidayPackage	No. Observations:	872
Model:	Logit	Df Residuals:	866
Method:	MLE	Df Model:	5
Date:	Sun, 23 Jan 2022	Pseudo R-squ.:	0.1281
Time:	17:20:16	Log-Likelihood:	-524.57
converged:	True	LL-Null:	-601.61
Covariance Type:	nonrobust	LLR p-value:	1.808e-31

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	2.2705	0.516	4.403	0.000	1.260	3.281
Salary	-1.831e-05	4.31e-06	-4.249	0.000	-2.68e-05	-9.86e-06
Age	-0.0474	0.009	-5.511	0.000	-0.064	-0.031
Educ	0.0399	0.029	1.367	0.172	-0.017	0.097
No_young_children	-1.3004	0.169	-7.711	0.000	-1.631	-0.970
Foreign	1.3210	0.200	6.599	0.000	0.929	1.713

Fig.20 – Logit Regression Results

Now all p values are less than 0.05. Hence all these attributes and their coeffficients have importance in deciding the target variable HolidayPackage.
Also we can see that coef value is highest for No_young_children followed by foreign, Age and salary
Salary coefficient value is very low i.e -00001932. So its impact is almost 0 on dependent variable

Logistics Regression Conclusion

Train Data:

```
AUC: 73%

Accuracy: 67%

Precision: 65%

f1-Score: 62%

Recall: 58%
```

Test Data:

```
AUC: 73%

Accuracy: 66%

Precision: 66%

f1-Score: 59%

Recall: 53%
```

Train and Test dataset have similar statistics, hence model is giving similar result for test and train data set.

With accuracy of 66% and recall rate of 52%, model is only able to predict 52% of total tours which were actually claimed as claimed.

Precision is 68% of test data which means, out of total employees predicted by model as opt for tour, 68% employees actually opted for the tour

F1-score is the harmonic mean of precision and recall, it takes into the effect of both the scores and this value is low if any of these 2 value is low.

Since we are building a model to predict if whether employee will opt for tour or not, for practical purposes, we will be more interested in correctly classifying 1 (employees opting for tour) than 0(employees not opting for tour).

If an employee not opting for tour is incorrectly predicted to be "opted for tour" by the model, then the impact on cost for the travel company would be bare minimum. But if an employee opted for tour is incorrectly predicted to be not opted by the model, then the cost impact would be very high for the tour and travel company. It’s a loss of potential lead for the company. Hence recall rate (actual data point identified as True by model) is very important in this scenario.

As Recall rate of test dataset is very poor around 52% thus this doesnt looks good enough for classification

Logistic regression equation is as shown below :-

Log (odd) = (2.81) + (-0.0) * Salary + (-0.05) * Age + (-1.3) * No_young_children + (1.21) * Foreign

We can see that salary coefficient is very small , this it can be removed. So our equation would become :-

Log (odd) = (2.67) * Intercept + (-0.0) * Salary + (-0.05) * Age + (-1.29) * No_young_children + (1.21) * Foreign

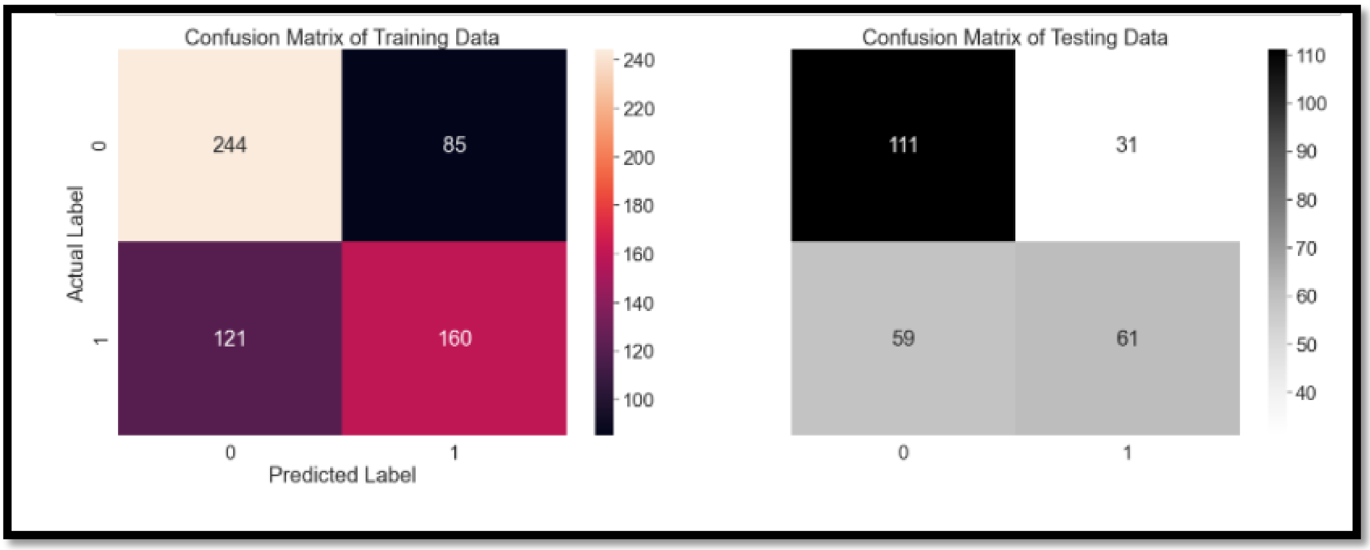
Most important attribute here is No of young children followed by Foreign and age

LDA Model

We have built the model using sklearn LinearDiscriminantAnalysis method

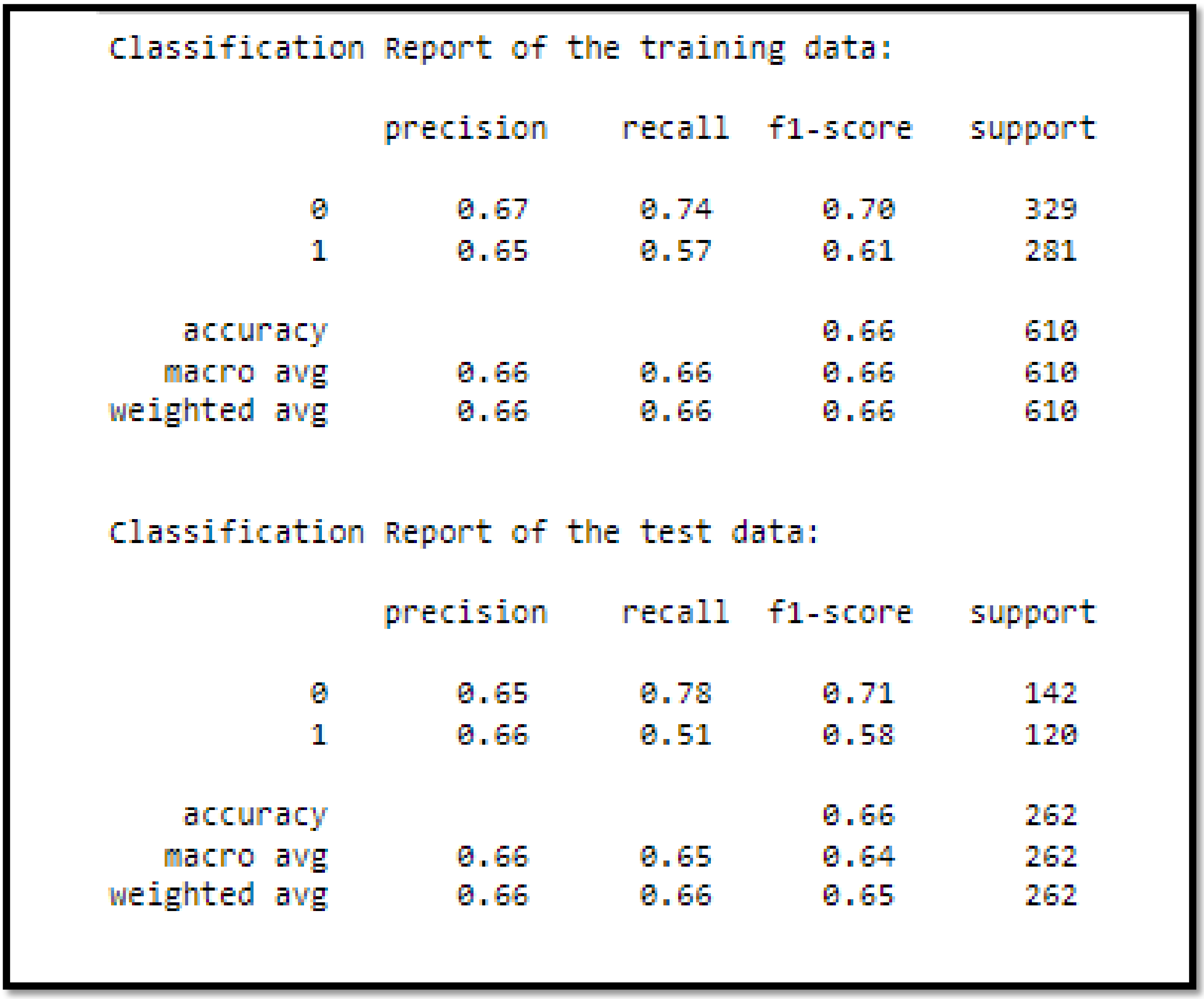
Model Evaluation

Training Data and Test Data Confusion Matrix Comparison



Training Data and Test Data Classification Report Comparison

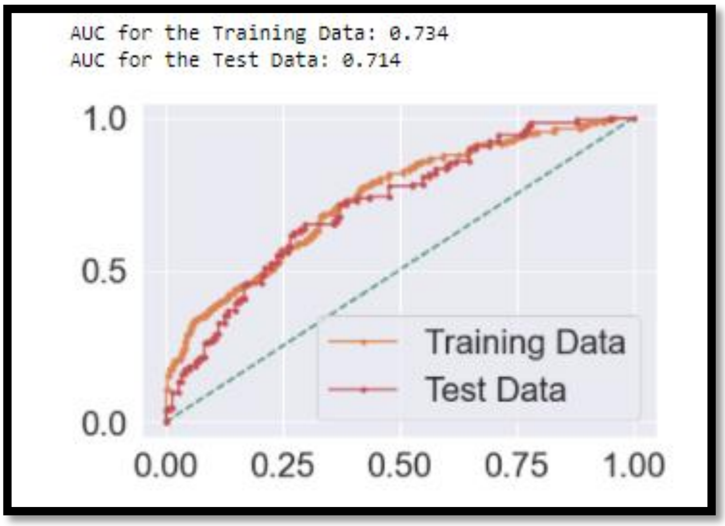
```
Classification Report of the training data:
```



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0       0.67      0.74      0.70       329
           1       0.65      0.57      0.61       281

    accuracy                           0.66       610
   macro avg       0.66      0.66      0.66       610
weighted avg       0.66      0.66      0.66       610


Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.65      0.78      0.71       142
           1       0.66      0.51      0.58       120

    accuracy                           0.66       262
   macro avg       0.66      0.65      0.64       262
weighted avg       0.66      0.66      0.65       262
```

AUC and ROC for the training data & test data

```
AUC for the Training Data: 0.734
AUC for the Test Data: 0.714
```



```
AUC for the Training Data: 0.734
AUC for the Test Data: 0.714
```

LDA Conclusion

Train Data:

AUC: 73%

Accuracy: 66%

Precision: 65%

f1-Score: 61%

Recall: 57%

Test Data:

AUC: 71%

Accuracy: 65%

Precision: 65%

f1-Score: 57%

Recall: 52%

Train and Test dataset have similar statistics, hence model is giving similar result for test and train data set.

With accuracy of 65% and recall rate of 52%, model is only able to predict 52% of total tours which were actually claimed as claimed.

Precision is 65% of test data which means, out of total employees predicted by model as opt for tour, 65% employees actually opted for the tour

F1-score is the harmonic mean of precision and recall, it takes into the effect of both the scores and this value is low if any of these 2 value is low.

Since we are building a model to predict if whether employee will opt for tour or not, for practical purposes, we will be more interested in correctly classifying 1 (employees opting for tour) than 0(employees not opting for tour).

If a employee not opting for tour is incorrectly predicted to be "opted for tour" by the model, then the impact on cost for the travel company would be bare minimum. But if am employee opted for tour is incorrectly predicted to be not opted by the model, then the cost impact would be very high for the tour and travel company. Its a loss of potential lead for the company. Hence recall rate (actual data point identified as True by model) is very important in this scenario.

2.3 Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model Final Model: Compare Both the models and write inference which model is best/optimized.

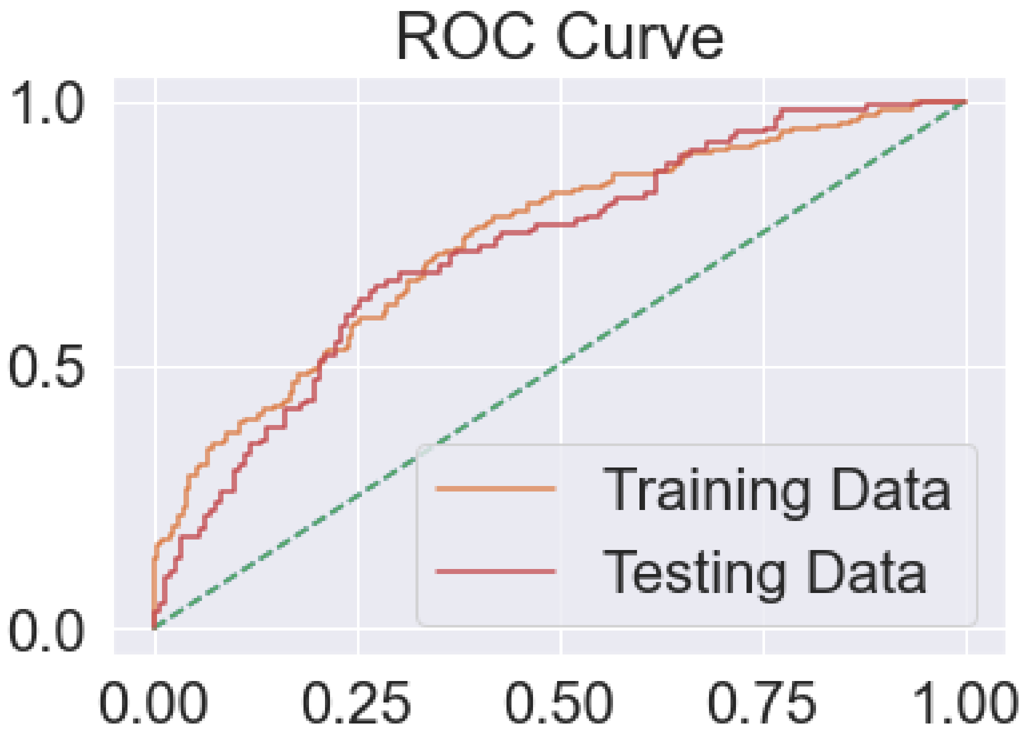
Confusion matrix cells are populated by the terms:
True Positive(TP)- The values which are predicted as True and are actually True.
True Negative(TN)- The values which are predicted as False and are actually False.
False Positive(FP)- The values which are predicted as True but are actually False.
False Negative(FN)- The values which are predicted as False but are actually True.

ROC Curve- Receiver Operating Characteristic(ROC) measures the performance of models by evaluating the trade-offs between sensitivity (true positive rate) and false (1- specificity) or false positive rate.
AUC - The area under curve (AUC) is another measure for classification models is based on ROC. It is the measure of accuracy judged by the area under the curve for ROC.

Performance Matrix of Logistics Regression model:

Train Data

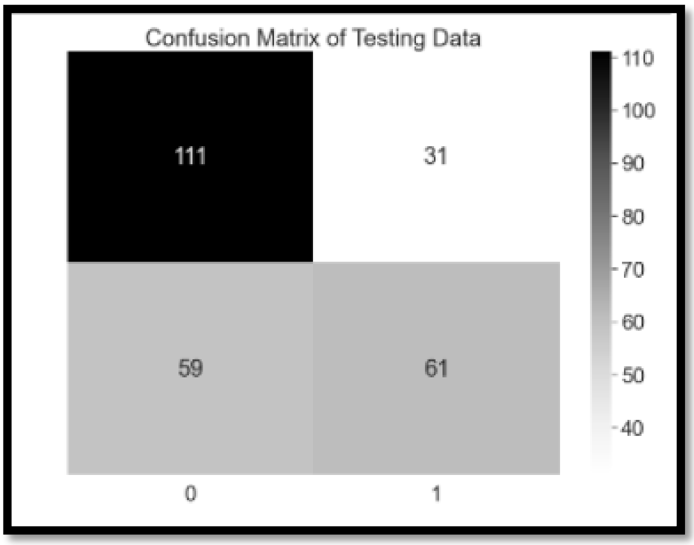
AUC score is 0.741 or 74.1%



Classification report for Train data:

```
Classification Report of the training data:

              precision    recall  f1-score   support

           0       0.67      0.75      0.71       329
           1       0.66      0.58      0.62       281

    accuracy                           0.67       610
   macro avg       0.67      0.66      0.66       610
weighted avg       0.67      0.67      0.67       610
```

Confusion Matrix for Test Data



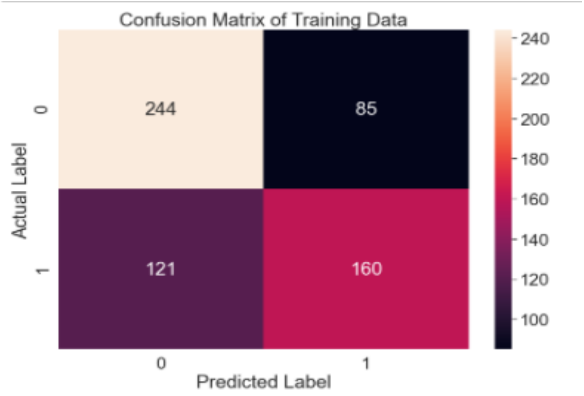
Classification report for Test Data

```
Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.65      0.78      0.71       142
           1       0.66      0.51      0.58       120

    accuracy                           0.66       262
   macro avg       0.66      0.65      0.64       262
weighted avg       0.66      0.66      0.65       262
```

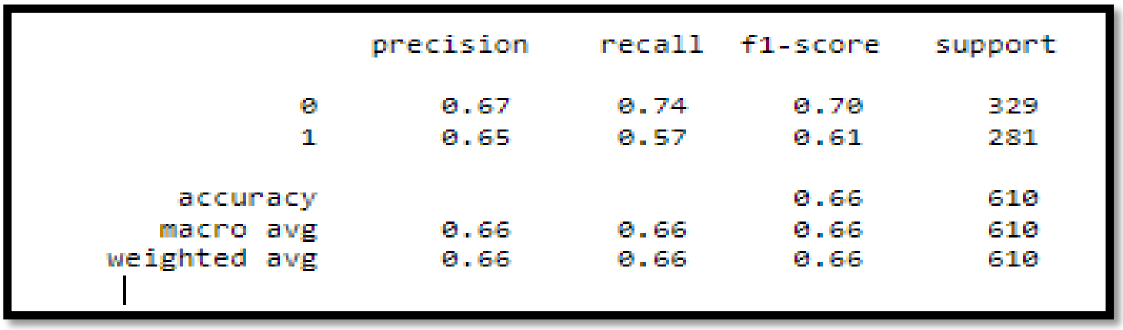
While looking the metrices for both training and the test data, it seems the accuracy scores are same on both models at 66%. Our model is close enough to be treated as a right fit model. The current model is not struggling with being a over fit model or an under fit model.
The AUC scores for both the training and test data is also same at 74%.
The model performance is good on F1 score as well with training data performing better at 62% while the test data gave a F1 score of 58%.

Let us see if these numbers change if we try to further refine the model using the GridSearchCV.

Applying GridSearchCV on Logistic Regression:
Confusion Matrix on the Train data:

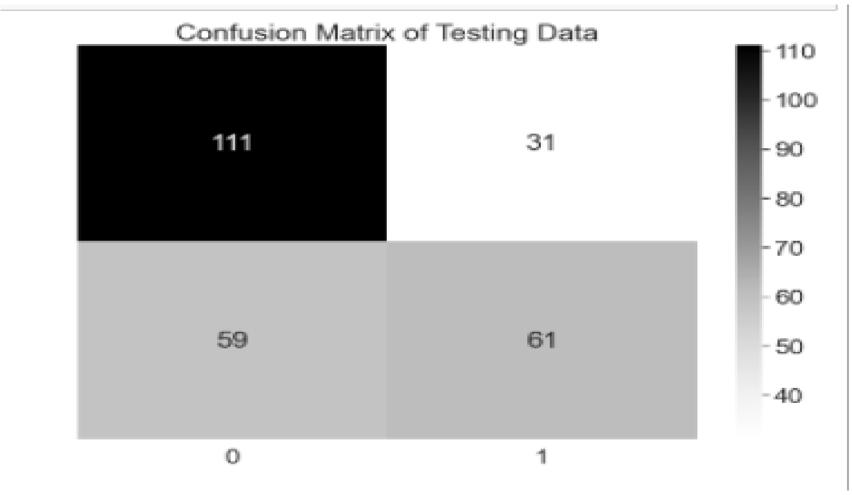


Classification report on Train data:

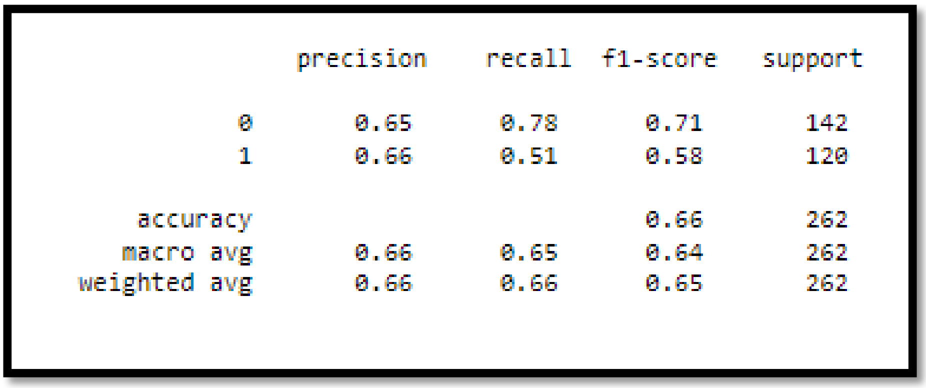


```
              precision    recall  f1-score   support

           0       0.67      0.74      0.70       329
           1       0.65      0.57      0.61       281

    accuracy                           0.66       610
   macro avg       0.66      0.66      0.66       610
weighted avg       0.66      0.66      0.66       610
```

Confusion Matrix on the Test data:



Classification report on Test data:



```
              precision    recall  f1-score   support

           0       0.65      0.78      0.71       142
           1       0.66      0.51      0.58       120

    accuracy                           0.66       262
   macro avg       0.66      0.65      0.64       262
weighted avg       0.66      0.66      0.65       262
```

Even after applying the best parameters, my model performance has been similar with almost same accuracy scores when compared with before putting the model through GridSearchCV step.

Accuracy score on Train data: 66%

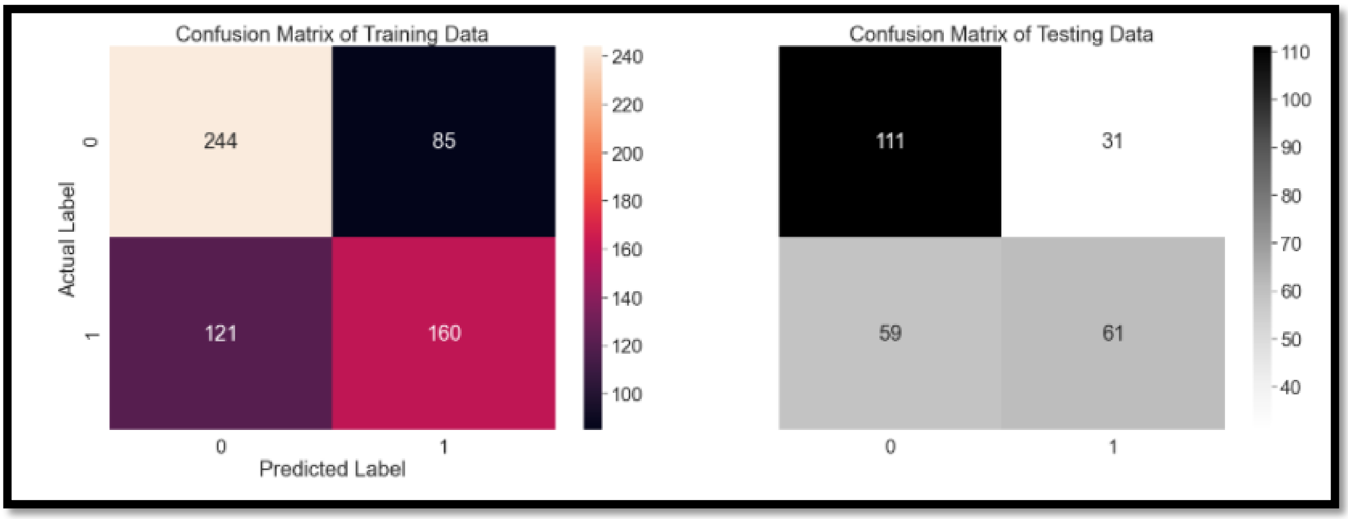
Accuracy score on test data: 66%

Also, the F1 scores have been similar when compared with the default model with 62% and 57% for Train and Test data, respectively.

To summarize, our model scores are not very good but seems to be a right fit model and seems to be avoiding the underfit and overfit scenarios. The training data seems to be performing a bit better when compared with test data though the difference is not much. The point to note is that F1 score seems to perform better on Training data and dips slightly in the test data. We can study the data further for data engineering to improve the scores, as needed for the business case.

LDA Model:

Confusion Matrix on Training data

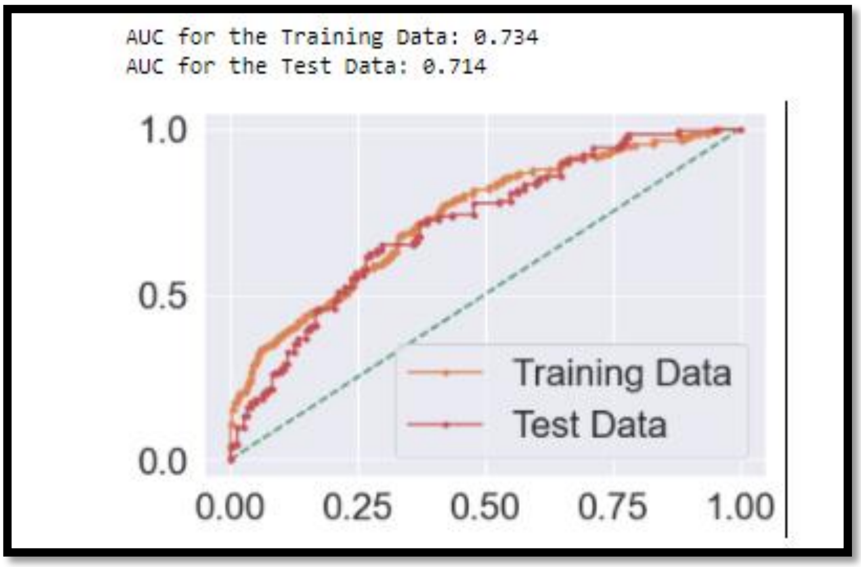


Classification report for both training and testing data:

```
        Classification Report of the training data:

               precision    recall  f1-score   support

           0       0.67      0.74      0.70       329
           1       0.65      0.57      0.61       281

    accuracy                           0.66       610
   macro avg       0.66      0.66      0.66       610
weighted avg       0.66      0.66      0.66       610


Classification Report of the test data:

               precision    recall  f1-score   support

           0       0.65      0.78      0.71       142
           1       0.66      0.51      0.58       120

    accuracy                           0.66       262
   macro avg       0.66      0.65      0.64       262
weighted avg       0.66      0.66      0.65       262
```

The accuracy score of the training data and test data is same at 66%. This is almost similar to the Logistic Regression model result so far. The AUC scores are marginally lower for the test data, else they are also almost similar to the Logistic Regression model. F1 scores are 61% and 57% for train and test data, respectively, which again is almost close to the logistic regression model.

```
AUC for the Training Data: 0.734
AUC for the Test Data: 0.714
```



```
AUC for the Training Data: 0.734
AUC for the Test Data: 0.714
```

Overall, the model seems to be a right fit model and is staying away from being referred as under fit or over fit model. Let us see if we can refine the results further and improve on the F1 score of the test data specifically.

As stated above, both the models – Logistics and LDA offers almost similar results. While LDA offers flexibility to control or change the important metrices such as precision, recall and F1 score by changing the custom cut-off. Like in this case study, the moment we changed the cut off to 40%, we were able to improve our precision, recall and F1 scores considerably. Further, this is up to the business if they would allow the play with the custom cut off values or no.
Though for this case study, I have chosen to proceed with Logistics Regression as its is easier to implement, interpret, and very efficient to train. Also, our dependent variable is following a binary classification of classes, and hence it is ideal for us to rely on the logistic regression model to study the test case at hand. Logistic regression is a classification algorithm used to find the probability of event success and event failure.

Q.2.4 Inference: Basis on these predictions, what are the insights and recommendations.

We started this test case with looking at the data correlation to identify early trends and patterns. At one stage, Salary and education seems to be important parameters which might have played out as an important predictor.

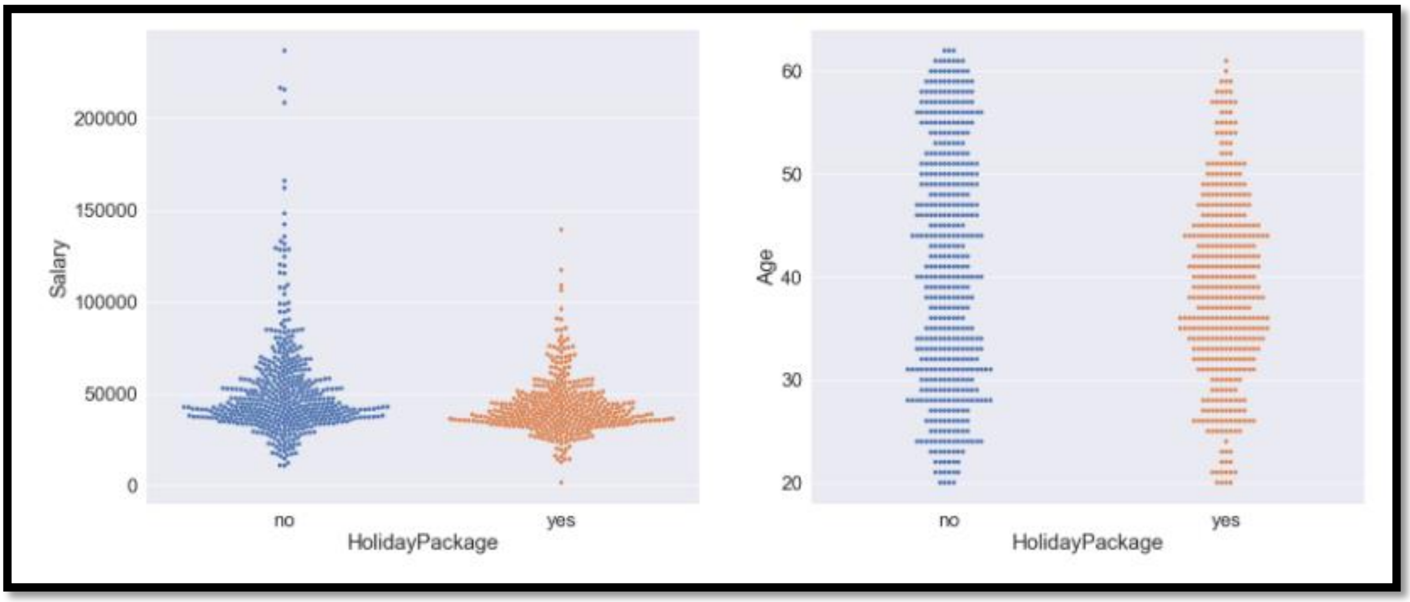


Fig.21 – Salary & Age variable vs HolidayPackage

Holiday Package vs Salary-

While performing the bivariate analysis we observe that Salary for employees opting for holiday package and for not opting for holiday package is similar in nature. However, the distribution is fairly spread out for people not opting for holiday packages.

Holiday Package vs Age –
The distribution of data for age variable with holiday package is also similar in nature. The range of age for p eople not opting for holliday package is more spread out when compared with people opting for yes.

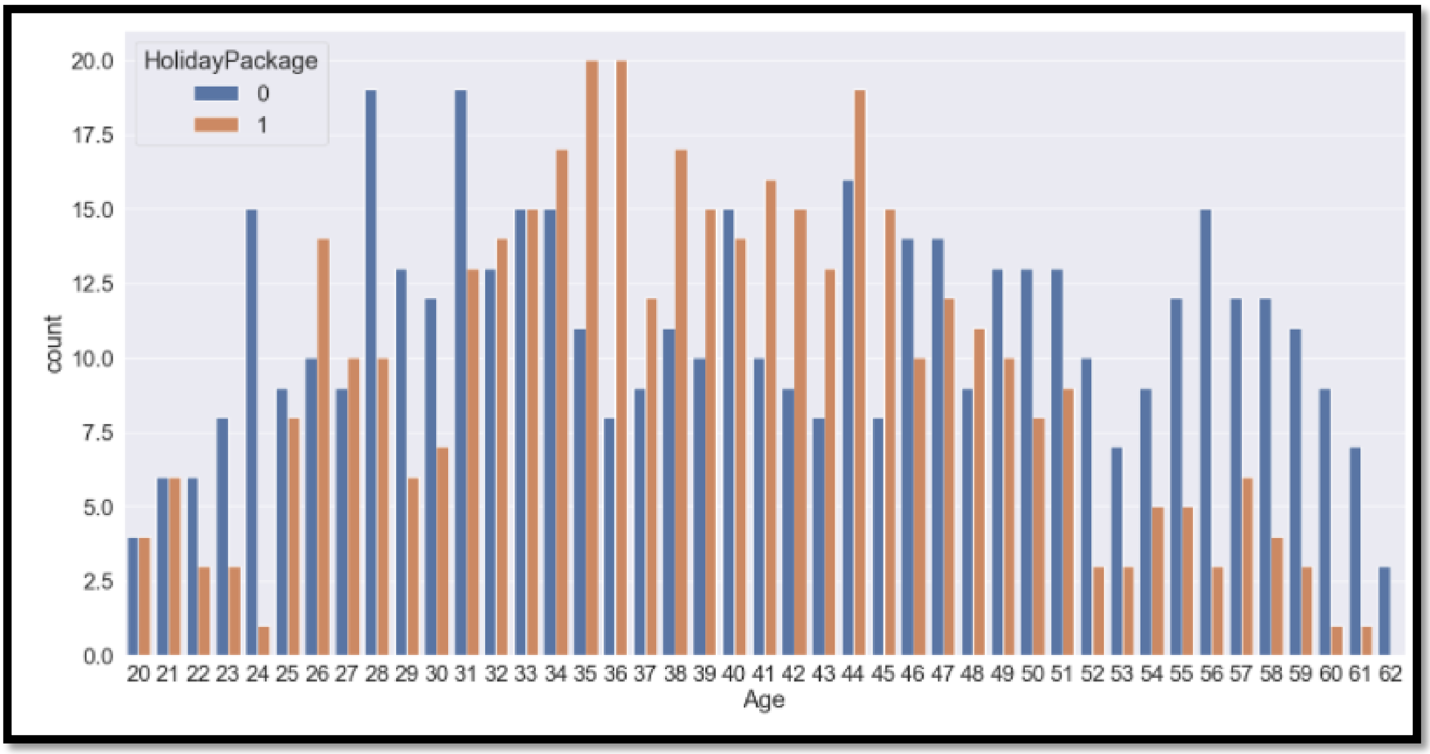


Fig.22 – Countplot on Holiday Package vs Age

We can clearly see that employees in middle range (34 to 45 years) are going for holiday package as compared to older and younger employees

However, almost similar distribution here for salary and age is indicating that they might not come out as strong predictors after the model is created. Let’s carry on with more data exploration and check

There is a significant difference in employees with younger children who are opting for holiday package and employees who are not opting for holiday package

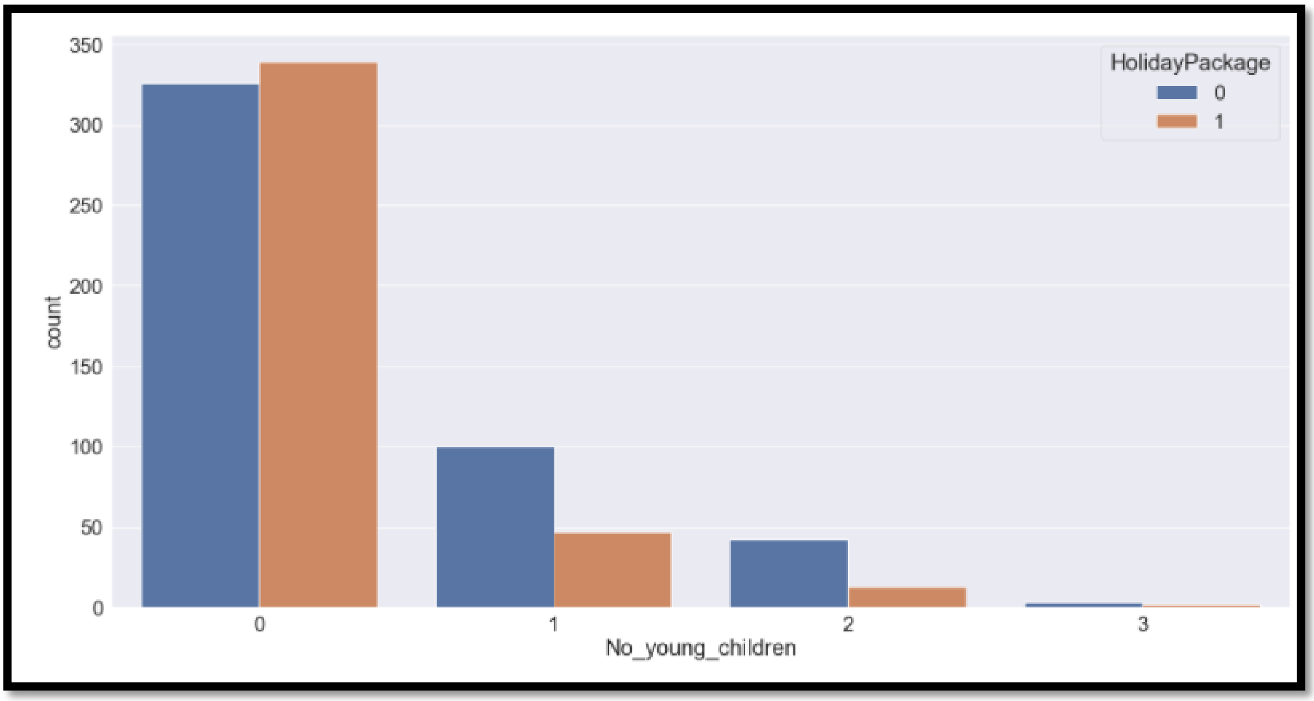


Fig.23 – Countplot on Holiday Package vs No_young_children

We can clearly see that people with younger children are not opting for holiday packages
We identify that employees with number of younger children has a varied distribution and might end up playing an important role in our model building process.

Employees with older children has almost similar distribution for opting and not opting for holiday packages across the number of children levels and hence I don’t think it will be an important predictor at all for my model and I did not include this specific variable for my model building process.

As interpretation,

1) There is no plausible effect of salary, age, and education on the prediction for Holliday_packages. These variables don’t seem to impact the decision to opt for holiday packages as we couldn’t establish a strong relation of these variables with the target variable

2) Foreign has emerged as a strong predictor with a positive coefficient value. The log likelihood or likelihood of a foreigner opting for a holiday package is high.

3) no_young_children variable is negating the probability for opting for holiday packages, especially for couple with number of young children at 2.

The company can try to bin salary ranges to see if they can derive some more meaningful interpretations out of that variable. May be club the salary or age in different buckets and see if there is some plausible impact on the predictor variable. OR else, the business can use some different model techniques to do a deep dive.

Recommendation:

1) The company should really focus on foreigners to drive the sales of their holiday packages as that's where the majority of conversions are going to come in.

2) The company can try to direct their marketing efforts or offers toward foreigners for a better conversion opting for holiday packages

3) The company should also stay away from targeting parents with younger children. The chances of selling to parents with 2 younger children is probably the lowest. This also gels with the fact that parents try and avoid visiting with younger children.

4) If the firm wants to target parents with older children, that still might end up giving favourable return for their marketing efforts then spent on couples with younger children.

Thank You!

☺