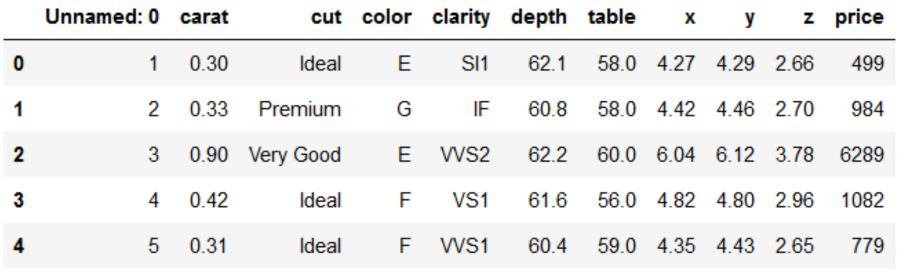
Predictive modelling

A. Cubic Zirconia dataset:

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

Answer: **EDA**

Table 1.1: EDA for Cubic Zirconia dataset

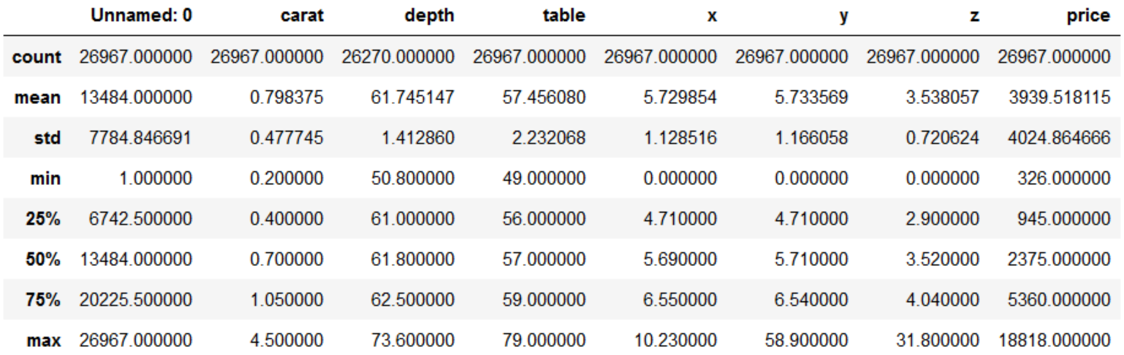


	Unnamed: 0	carat	cut	color	clarity	depth	table	x	y	z	price
0	1	0.30	Ideal	E	SI1	62.1	58.0	4.27	4.29	2.66	499
1	2	0.33	Premium	G	IF	60.8	58.0	4.42	4.46	2.70	984
2	3	0.90	Very Good	E	VVS2	62.2	60.0	6.04	6.12	3.78	6289
3	4	0.42	Ideal	F	VS1	61.6	56.0	4.82	4.80	2.96	1082
4	5	0.31	Ideal	F	VVS1	60.4	59.0	4.35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

Dataset have 11 variables which speaks about the carat, cut, colour and different varieties of gem stones. We have the dataset which contains the prices and attributes of almost 27000 cubic zirconia which is inexpensive diamonds.

Descriptive Statistics for the dataset

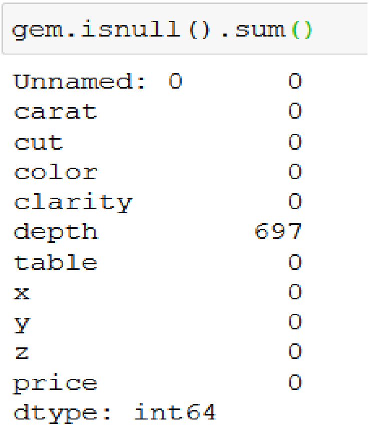
Table 1.2: Descriptive Statistics for Cubic Zirconia dataset



	Unnamed: 0	carat	depth	table	x	y	z	price
count	26967.000000	26967.000000	26270.000000	26967.000000	26967.000000	26967.000000	26967.000000	26967.000000
mean	13484.000000	0.798375	61.745147	57.456080	5.729854	5.733569	3.538057	3939.518115
std	7784.846691	0.477745	1.412860	2.232068	1.128516	1.166058	0.720624	4024.864666
min	1.000000	0.200000	50.800000	49.000000	0.000000	0.000000	0.000000	326.000000
25%	6742.500000	0.400000	61.000000	56.000000	4.710000	4.710000	2.900000	945.000000
50%	13484.000000	0.700000	61.800000	57.000000	5.690000	5.710000	3.520000	2375.000000
75%	20225.500000	1.050000	62.500000	59.000000	6.550000	6.540000	4.040000	5360.000000
max	26967.000000	4.500000	73.600000	79.000000	10.230000	58.900000	31.800000	18818.000000

Figure 1.1: Cubic Zirconia dataset information

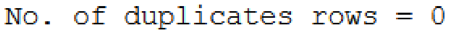
The above table shows there are 697 null values for depth variable. Other variables don't have any null values.



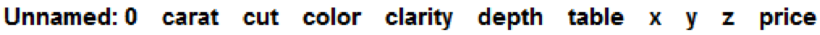
```
gem.isnull().sum()
```

```
Unnamed: 0      0
carat           0
cut             0
color           0
clarity         0
depth         697
table           0
x               0
y               0
z               0
price           0
dtype: int64
```

Check for Duplicate Values



```
No. of duplicates rows = 0
```



Unnamed: 0	carat	cut	color	clarity	depth	table	x	y	z	price

There are no duplicate values in the dataset.

Univariate Analysis

Histogram:

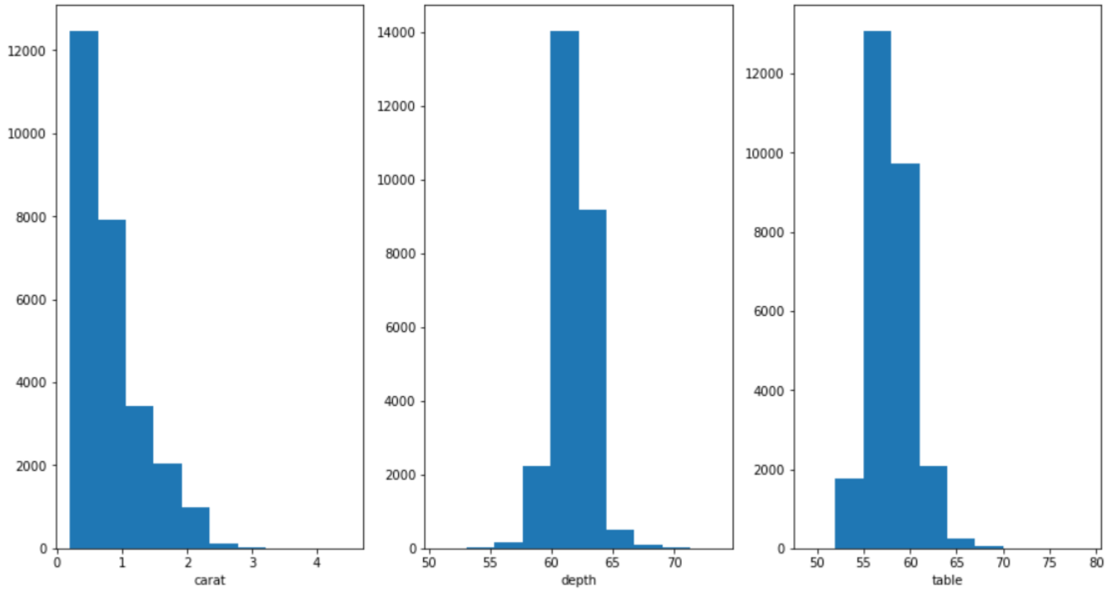


Figure 1.2: Histogram of carat, depth and table.

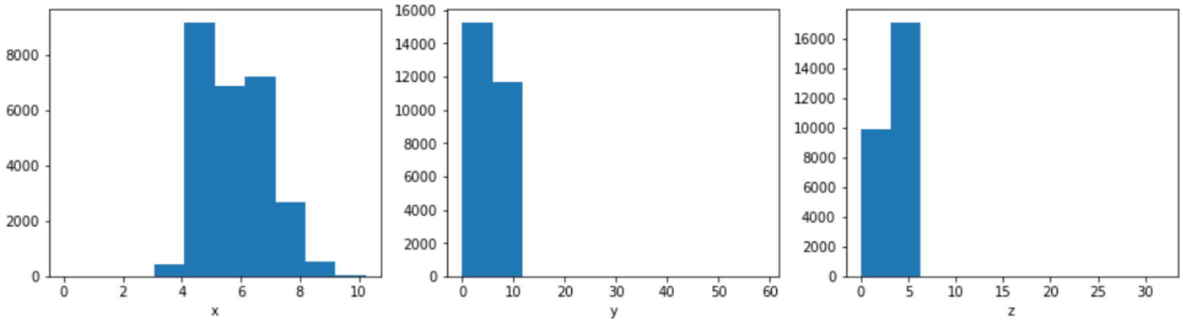


Figure 1.3: Histogram of x, y and z.

Box Plot:

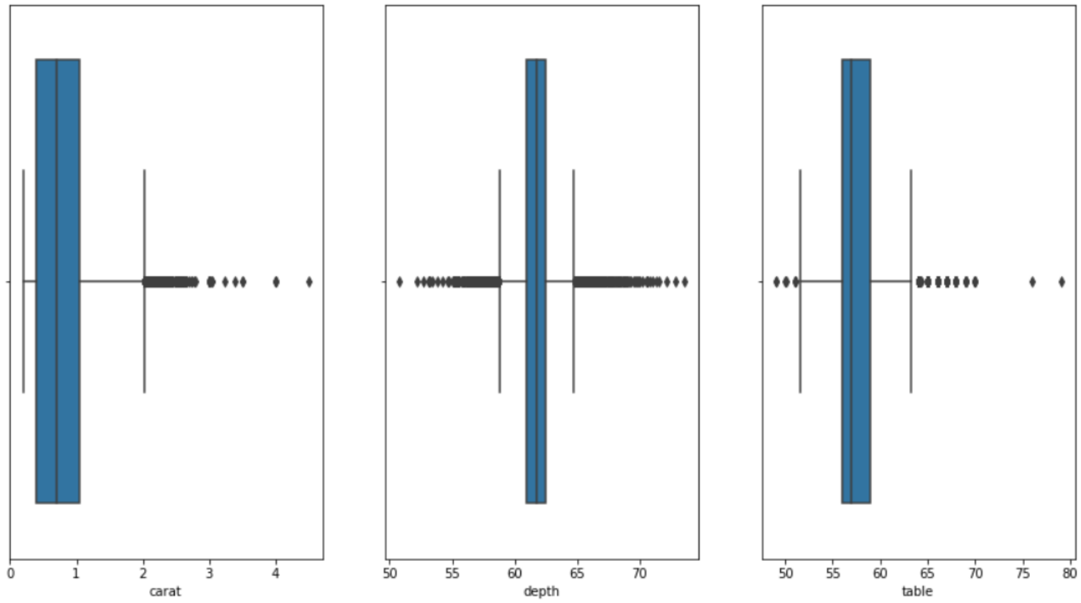


Figure 1.4: Box plots of carat, depth and table.

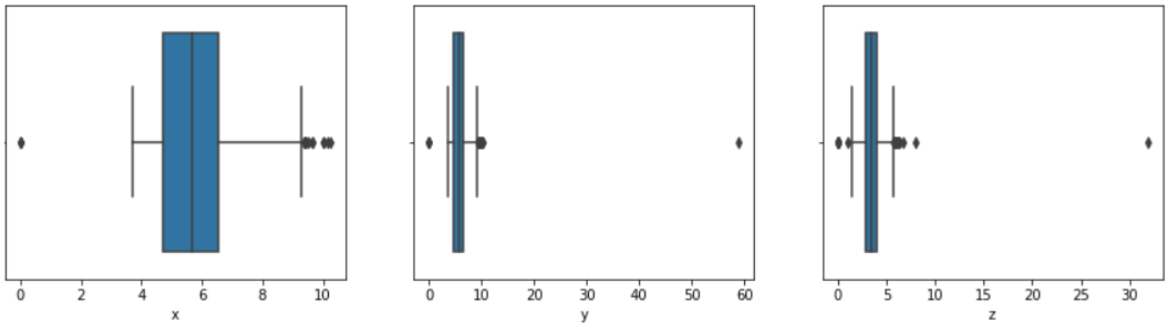


Figure 1.5: Box plots of x, y and z.

- From the given box plots we can see, all the variables have outliers and the number of outliers is huge in the dataset.

- Both the variables carat and depth have the maximum number or outliers

- Rest all variables have less numbers when compared to carat and depth.

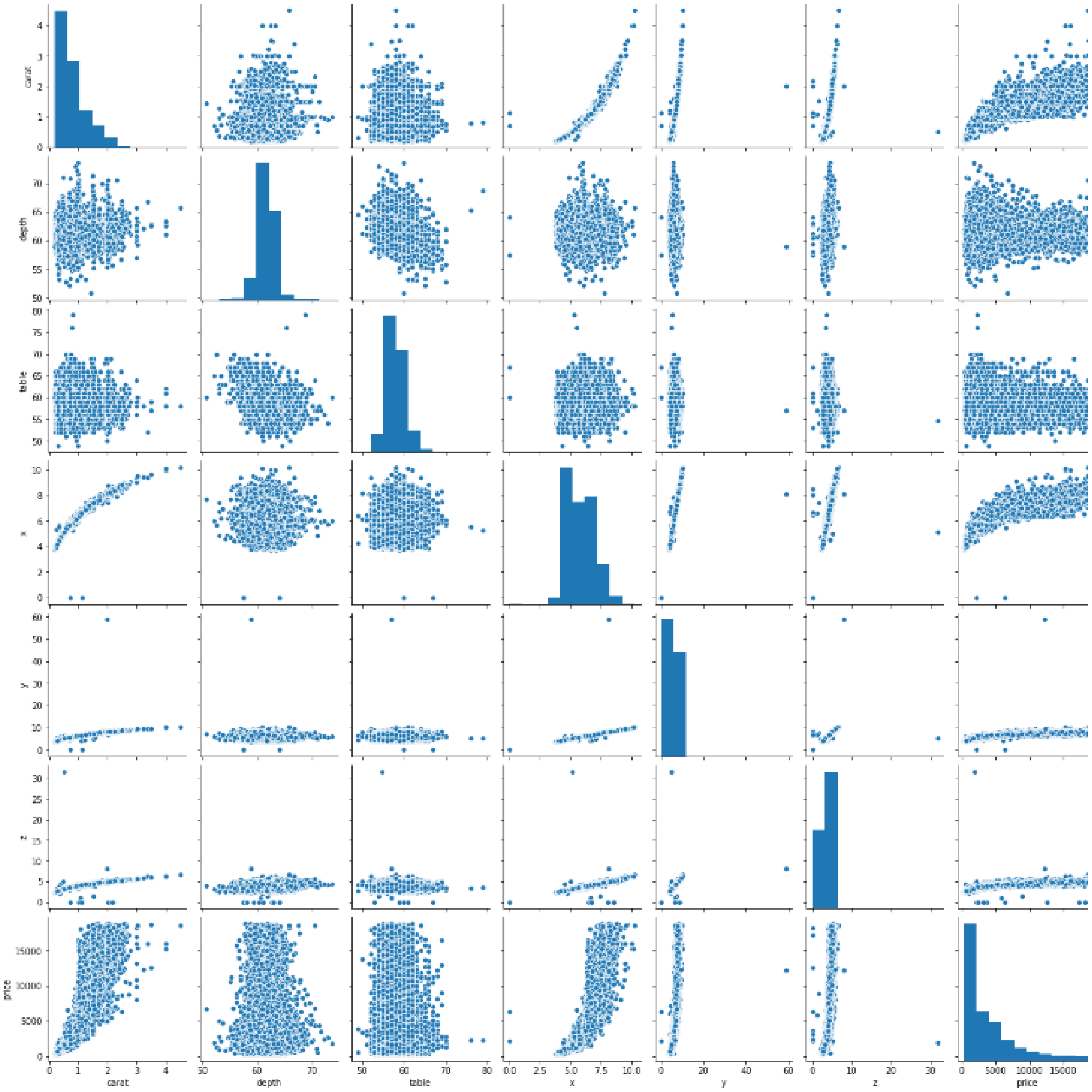
- The above histogram and box plots represent positive skewness.

- variable **x** has slightly less positive skewness when compared to the other variables, depth looks almost like normally distributed.

- variable **y** shows the highest positive skewness.

Multivariate Analysis

Pair plot:

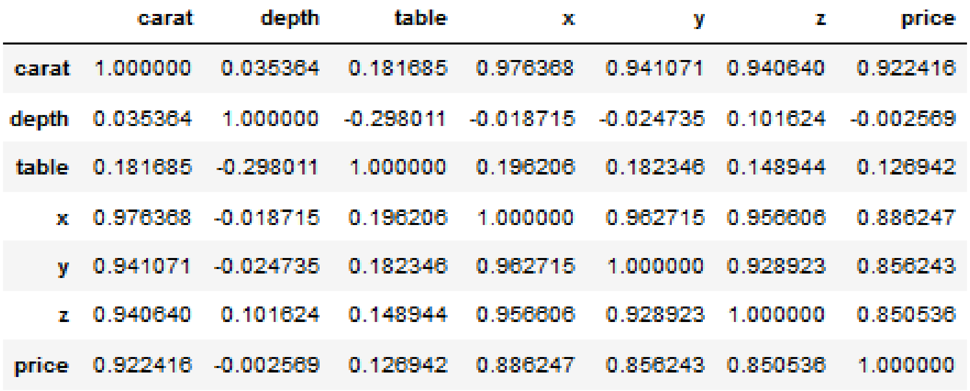


- In the above diagram scatter plots are plotted for all the numerical columns in the dataset. A scatter plot is a visual representation of the degree of correlation between any two columns. The pair plot function in seaborn makes it very easy to generate joint scatter plots for all the columns in the data.

- Pair plot shows scatter plot as well as the histogram between all the variables of the dataset.
- Pair plot makes it visually easier to understand if the data is highly co related to each other or not.

Correlation Matrix

Table 1.3: Correlation matrix of the Cubic Zirconia Dataset



	carat	depth	table	x	y	z	price
carat	1.000000	0.035364	0.181685	0.976368	0.941071	0.940640	0.922416
depth	0.035364	1.000000	-0.298011	-0.018715	-0.024735	0.101624	-0.002569
table	0.181685	-0.298011	1.000000	0.196206	0.182346	0.148944	0.126942
x	0.976368	-0.018715	0.196206	1.000000	0.962715	0.956606	0.886247
y	0.941071	-0.024735	0.182346	0.962715	1.000000	0.928923	0.856243
z	0.940640	0.101624	0.148944	0.956606	0.928923	1.000000	0.850536
price	0.922416	-0.002569	0.126942	0.886247	0.856243	0.850536	1.000000

Heat map:

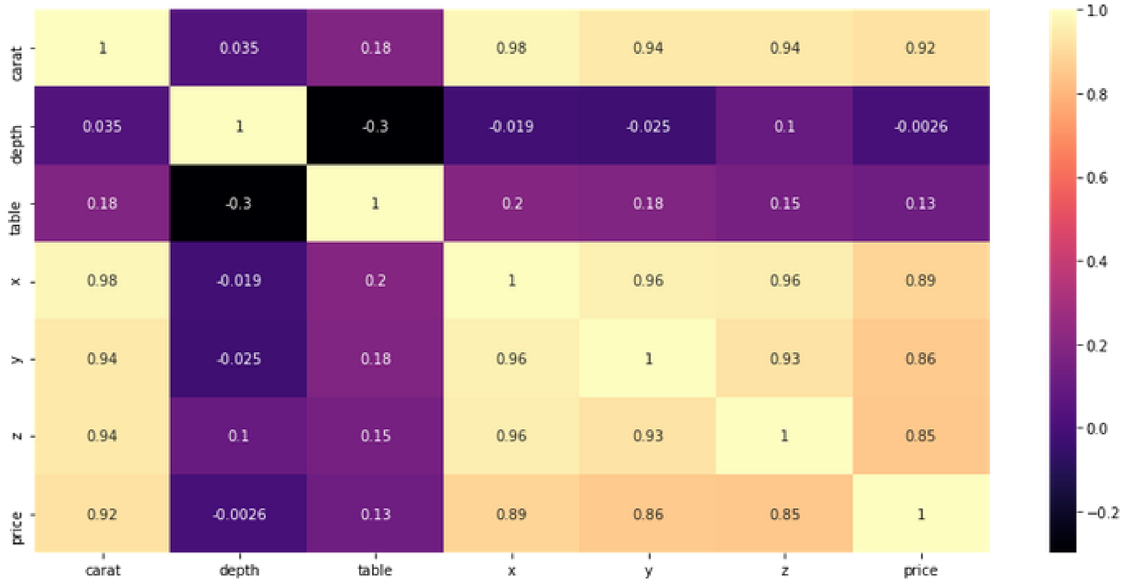


Figure 1.6: Heat map of the Cubic Zirconia Dataset

• We use the heatmap to check the correlation in visual and coloured manner. The colour helps with picking out the most correlated variables easily. The darker the colour, higher the correlation.

• x, y and z show the highest positive correlation of 0.98, length, width and height are equally co related to each other.

• carat and price show highest positive correlation of 0.92, higher the carat more the price.

• table and depth show the lowest positive correlation of -0.0026 and 0.13.

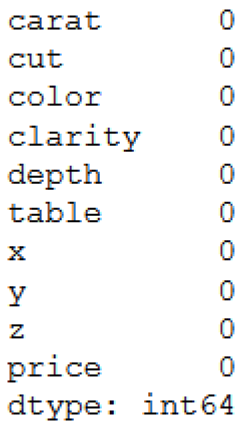
• depth show's negative correlation with all the variables.

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? Do you think scaling is necessary in this case?

Answer:

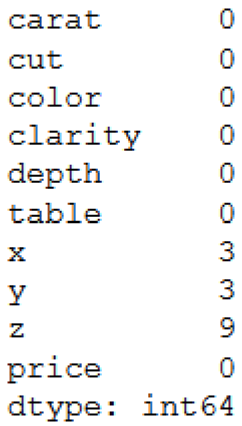
Data after imputing:

We have imputed the null values from the depth variable, as it was the only variable which had null values.



```
carat       0
cut         0
color       0
clarity     0
depth       0
table       0
x           0
y           0
z           0
price       0
dtype: int64
```

Checking for values which are equal to zero.



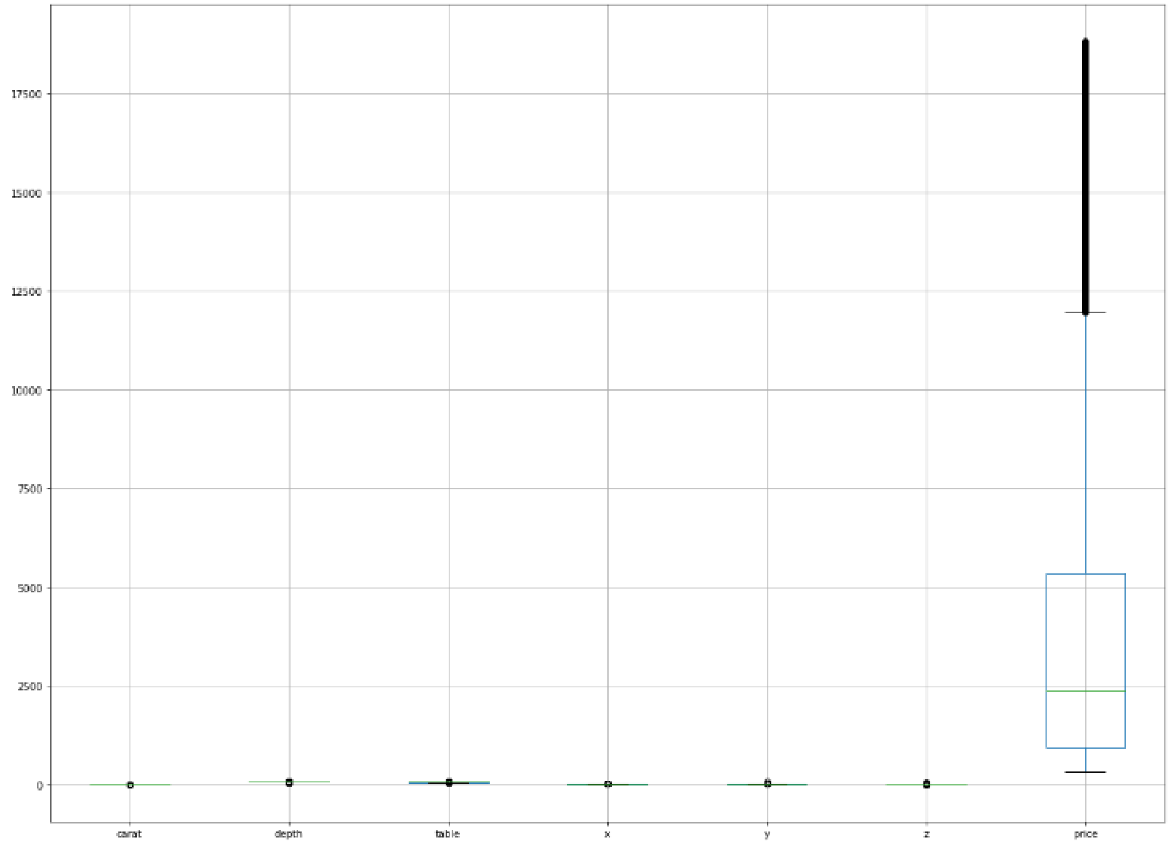
```
carat       0
cut         0
color       0
clarity     0
depth       0
table       0
x           3
y           3
z           9
price       0
dtype: int64
```

The variables x, y and z have values which are equal to zero / zero.
We need to drop those values to get a data set which is clean and the model is better.
The zero/ equal to zero values are not useful in the data. The expressions are split that way because of the relative precedence of the comma separator compared to the equality operator: Python sees a tuple containing two expressions, one of which happens to be an equality test, instead of an equality test between two tuples.
A tuple is a collection of values, and unlike an equality test, assignment has no value in Python. An assignment is not an expression, but a statement; it does not have a value that can be included into a tuple or any other surrounding expression.

After removing the zero values data count:

```
(26958, 10)
```

Removing Outliers:



We have removed the outliers from all the variables apart from the target variable price.

Having outliers in the target variable doesn’t make difference in the model. As price is the Y variable is what we are trying to conclude. Hence considering all the other variables as X, they are individual variables which will help us find y.

Why not scale the data.

We are not aware of the min/max hence scaling won’t be a step we need to use for this dataset.

As per the linear regression model scaling won’t impact the data. Hence we will choose not to scale the data.

If there is a vast difference in the range say few ranging in thousands and few ranging in the tens, and it makes the underlying assumption that higher ranging numbers have superiority of some sort. So these more significant number starts playing a more decisive role while training the model.

The machine learning algorithm works on numbers and does not know what that number represents. A weight of 10 grams and a price of 10 dollars represents completely two different things, which is understood for humans, but for a model as a feature, it treats both as same.

Suppose we have two features of weight and price, as in the below table. The "Weight" cannot have a meaningful comparison with the "Price." So the assumption algorithm makes that since "Weight" > "Price," thus "Weight," is more important than "Price."

1.3 Encode the data (having string values) for Modelling. Data Split: Split the data into test and train (70:30). Apply Linear regression. Performance Metrics: Check the performance of Predictions on Train and Test sets using R square, RMSE.

Answer:

Encoding the data using label encoding:

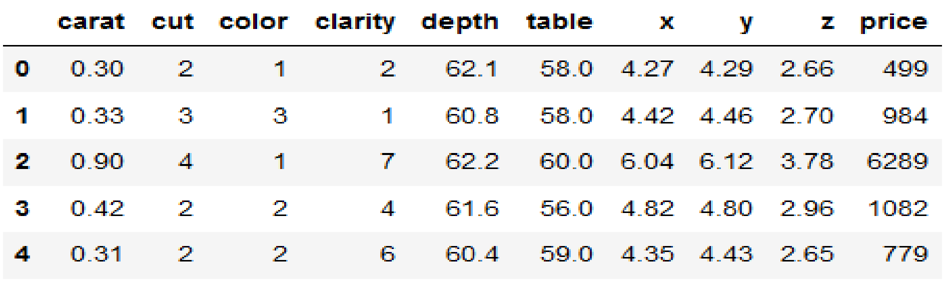
Typically, any structured dataset includes multiple columns a combination of numerical as well as categorical variables. A machine can only understand the numbers. It cannot understand the text. That's essentially the case with Machine Learning algorithms too.

That's primarily the reason we need to convert categorical columns to numerical columns so that a machine learning algorithm understands it.

- Label Encoding- The reason we use Label encoding as it is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.

  As you can see here, all the columns are categorical feature as it is represented by the object data type are numerical features as they are represented by *int64*.

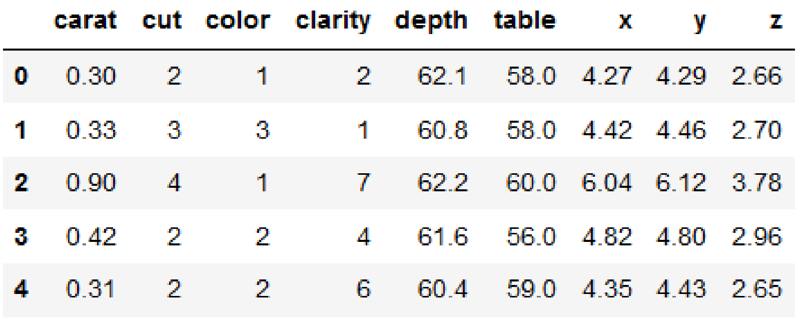
  Data after encoding:



	carat	cut	color	clarity	depth	table	x	y	z	price
0	0.30	2	1	2	62.1	58.0	4.27	4.29	2.66	499
1	0.33	3	3	1	60.8	58.0	4.42	4.46	2.70	984
2	0.90	4	1	7	62.2	60.0	6.04	6.12	3.78	6289
3	0.42	2	2	4	61.6	56.0	4.82	4.80	2.96	1082
4	0.31	2	2	6	60.4	59.0	4.35	4.43	2.65	779

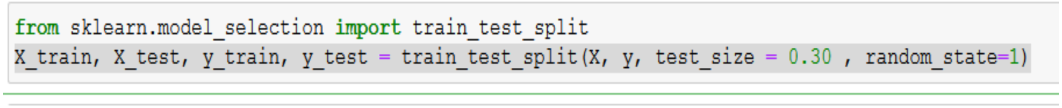
Splitting data into train and test data:

We are dropping the price target variable. Head after dropping price variable.



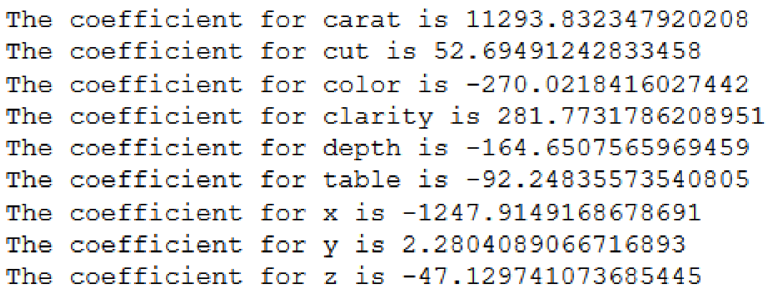
	carat	cut	color	clarity	depth	table	x	y	z
0	0.30	2	1	2	62.1	58.0	4.27	4.29	2.66
1	0.33	3	3	1	60.8	58.0	4.42	4.46	2.70
2	0.90	4	1	7	62.2	60.0	6.04	6.12	3.78
3	0.42	2	2	4	61.6	56.0	4.82	4.80	2.96
4	0.31	2	2	6	60.4	59.0	4.35	4.43	2.65

Splitting X and y into training and test set in 70:30 ratio



```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30 , random_state=1)
```

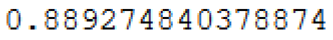
Finding coefficients



```
The coefficient for carat is 11293.832347920208
The coefficient for cut is 52.69491242833458
The coefficient for color is -270.0218416027442
The coefficient for clarity is 281.7731786208951
The coefficient for depth is -164.6507565969459
The coefficient for table is -92.24835573540805
The coefficient for x is -1247.9149168678691
The coefficient for y is 2.2804089066716893
The coefficient for z is -47.12974107368544
```

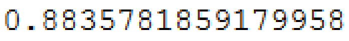
R square on training and testing data:

Train Data for R square:



```
0.889274840378874
```

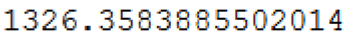
Test Data for R square:



```
0.8835781859179958
```

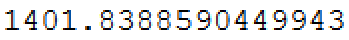
RMSE on train and test data

Predicted Train:



```
1326.3583885502014
```

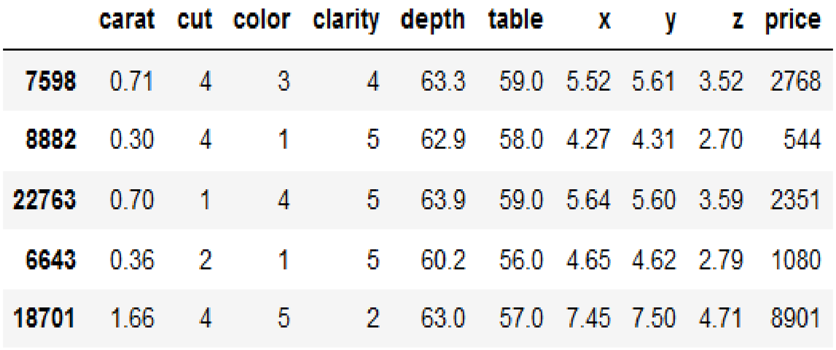
Predicted Test:



```
1401.8388590449943
```

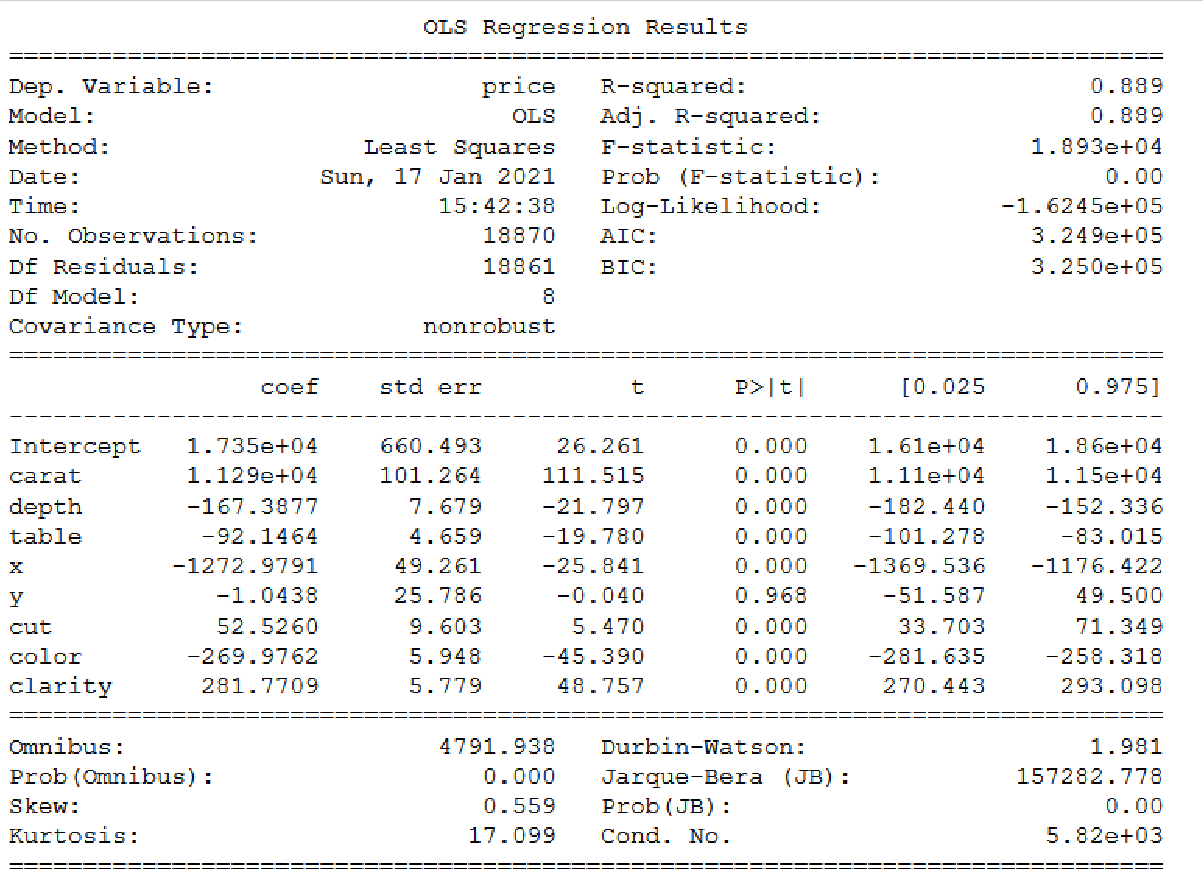
Linear Regression using stats models

Linear regression is one of the fundamental statistical and machine learning techniques. Whether you want to do statistics, machine learning, or scientific computing, there are good chances that you'll need it. It's advisable to learn it first and then proceed towards more complex methods.

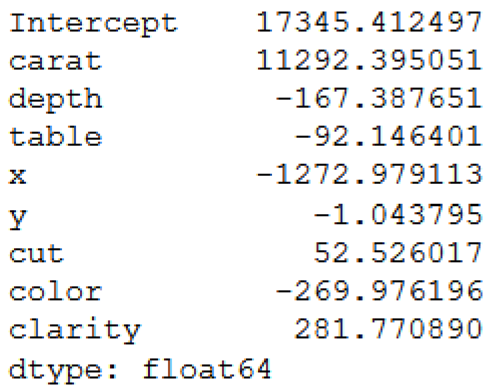


	carat	cut	color	clarity	depth	table	x	y	z	price
7598	0.71	4	3	4	63.3	59.0	5.52	5.61	3.52	2768
8882	0.30	4	1	5	62.9	58.0	4.27	4.31	2.70	544
22763	0.70	1	4	5	63.9	59.0	5.64	5.60	3.59	2351
6643	0.36	2	1	5	60.2	56.0	4.65	4.62	2.79	1080
18701	1.66	4	5	2	63.0	57.0	7.45	7.50	4.71	8901

Summary of the stats model:



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.889
Model:                            OLS   Adj. R-squared:                  0.889
Method:                 Least Squares   F-statistic:                 1.893e+04
Date:                Sun, 17 Jan 2021   Prob (F-statistic):               0.00
Time:                        15:42:38   Log-Likelihood:            -1.6245e+05
No. Observations:               18870   AIC:                         3.249e+05
Df Residuals:                   18861   BIC:                         3.250e+05
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   1.735e+04    660.493     26.261      0.000    1.61e+04    1.86e+04
carat       1.129e+04    101.264    111.515      0.000    1.11e+04    1.15e+04
depth       -167.3877      7.679    -21.797      0.000    -182.440    -152.336
table        -92.1464      4.659    -19.780      0.000    -101.278     -83.015
x          -1272.9791     49.261    -25.841      0.000   -1369.536   -1176.422
y             -1.0438     25.786     -0.040      0.968     -51.587      49.500
cut           52.5260      9.603      5.470      0.000      33.703      71.349
color       -269.9762      5.948    -45.390      0.000    -281.635    -258.318
clarity      281.7709      5.779     48.757      0.000     270.443     293.098
==============================================================================
Omnibus:                     4791.938   Durbin-Watson:                   1.981
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           157282.778
Skew:                           0.559   Prob(JB):                         0.00
Kurtosis:                      17.099   Cond. No.                     5.82e+03
==============================================================================
```



```
Intercept     17345.412497
carat         11292.395051
depth          -167.387651
table           -92.146401
x             -1272.979113
y                -1.043795
cut              52.526017
color          -269.976196
clarity         281.770890
dtype: float64
```

In this article, we will use Python's **stats model's** module to implement Ordinary Least Squares **(**OLS**)** method of linear regression.

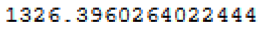
A linear regression model establishes the relation between a dependent variable(**y**) and at least one independent variable(**x**) as

Let's talk about the summary of OLS regression. We have intercept of 17345.41, carat 11292.39, depth, table, x, y, color is displaying negative regression of -167.38, -92.14 ,-1272.97,-1.04,-269.97 respectively. cut and clarity have 52.52 and 281.77 positive regression.

Standard Errors assume that the covariance matrix of the errors is correctly specified.

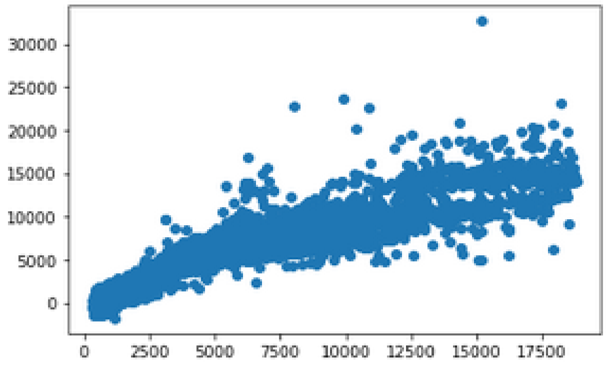
In *OLS* method, we have to choose the values of and such that, the total sum of squares of the difference between the calculated and observed values of y, is minimised.

MSE Square root.



```
1326.3960264022444
```

Scatter plot for test data price variable Y Predict.



The scatter plot represents the price has huge number of outliers, as price can vary depending on the diamond height, weight and size.

Scatter plot is the best representation to check if the data is proportionately divided or not.

The Sqrt of Mean square error is 1326.39 which is comparatively high.

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

Answer:

The business insights and recommendations for the data set cubic zirconia we understand the price and carat are equally proportional to each other, which is positively co related.

Diamonds are a rare and naturally occurring mineral that are comprised of carbon.

The variable clarity is also positively co related to price.

Hence business should focus on the carat and clarity of the diamond. Which will help the customers purchase the diamonds more often when compared to buying just gold.

Having bigger/larger carat diamond will give the business more profits. As jewellery is something not often purchased company should focus on attracting more customers by having better carat and clarity products, if we look deeper the clarity refers to the cut which is positively corelated to the price as well.

Business needs to understand the quality of the diamond depends on the carat, clarity and cut. These are the three aspects business should be always focused to make sure the sales are high.

The jewellery designs should be made available in smaller and larger designs or sets which in turns gives better options to the customer entering the store to purchase a necklace matching with earrings and a larger sale to the jewellery store in turns the business.

The higher profitable stones will be the once which are good in carat, color, clarity and cut. However, we need to focused on the one more parameter which are negatively co related in the data set. But play a very important role when comes to sale or profits the two variables are depth this variable play major role because the higher the cubic zirconia diamond size the expensive it is. Hence the diameter plays a major role.

I would like to conclude my findings by stating the business should be more focused on the following five parameters carat, color, cut , clarity and depth of the diamonds to understand the customer they have who often visit, will and buy from there store.

B. Holiday Package data set

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:

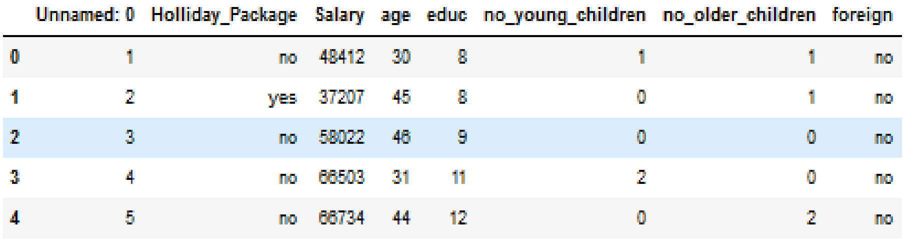
For building the models first we need to split the dataset into training and testing dataset. We will keep the training size as 70% and testing data size as 30 %. Keep random state as 1.

- We will drop the un named 0 column as it doesn’t have any use during the eda.
- Holiday package is the dependent variable and rest of the variables as independent variables.

Table 1.1: EDA for Holiday Package data set

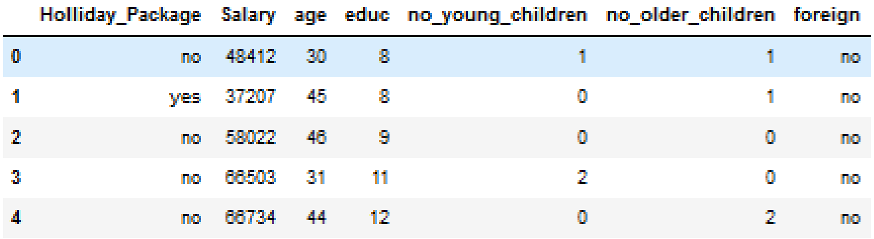
Checking if the data is loaded fine and head reads all the variables from the actual data.

EDA:



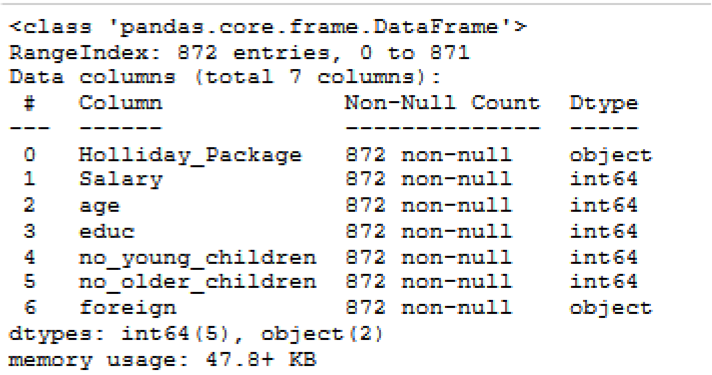
	Unnamed: 0	Holliday_Package	Salary	age	educ	no_young_children	no_older_children	foreign
0	1	no	48412	30	8	1	1	no
1	2	yes	37207	45	8	0	1	no
2	3	no	58022	46	9	0	0	no
3	4	no	66503	31	11	2	0	no
4	5	no	66734	44	12	0	2	no

Dropping the variable unnamed:0 as it doesn't carry any good for EDA.



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

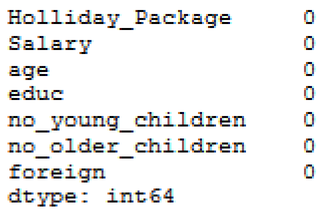
Data Info



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 872 entries, 0 to 871
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Holliday_Package   872 non-null    object
 1   Salary             872 non-null    int64 
 2   age                872 non-null    int64 
 3   educ               872 non-null    int64 
 4   no_young_children  872 non-null    int64 
 5   no_older_children  872 non-null    int64 
 6   foreign            872 non-null    object
dtypes: int64(5), object(2)
memory usage: 47.8+ KB
```

As the data has object, we need to convert them to integer type.

Finding null values:



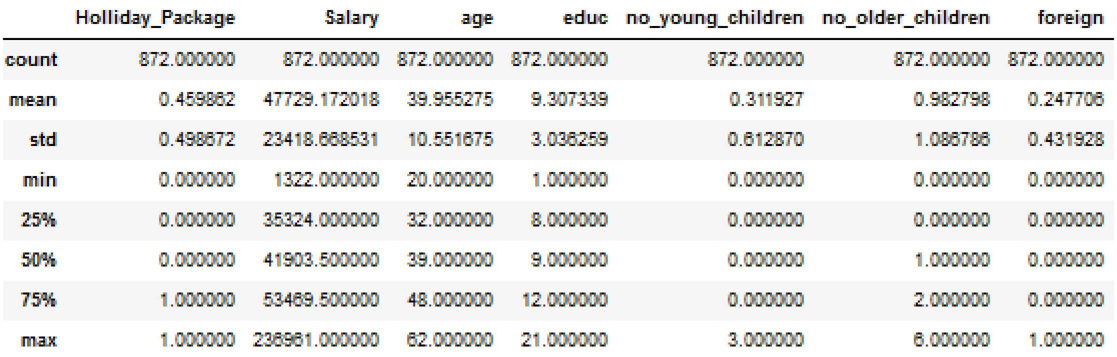
```
Holliday_Package      0
Salary                0
age                   0
educ                  0
no_young_children     0
no_older_children     0
foreign               0
dtype: int64
```

Check for Duplicate Values

```
Number of duplicate rows = 0
```

Descriptive Statistics for the dataset

Table 1.2: Descriptive Statistics for Holiday package dataset



	Holliday_Package	Salary	age	educ	no_young_children	no_older_children	foreign
count	872.000000	872.000000	872.000000	872.000000	872.000000	872.000000	872.000000
mean	0.459862	47729.172018	39.955275	9.307339	0.311927	0.982798	0.247706
std	0.498672	23418.668531	10.551675	3.036259	0.612870	1.086786	0.431928
min	0.000000	1322.000000	20.000000	1.000000	0.000000	0.000000	0.000000
25%	0.000000	35324.000000	32.000000	8.000000	0.000000	0.000000	0.000000
50%	0.000000	41903.500000	39.000000	9.000000	0.000000	1.000000	0.000000
75%	1.000000	53469.500000	48.000000	12.000000	0.000000	2.000000	0.000000
max	1.000000	236961.000000	62.000000	21.000000	3.000000	6.000000	1.000000

Univariate Analysis

- From Histogram we see that all the continuous variables are positively skewed. Age somehow looks like normally distributed.
- Salary shows highest positively skewed distribution whereas educ has a little negative skewness.

Histogram:

A histogram is used to summarize discrete or continuous data. In other words, it provides a visual interpretation. This requires focusing on the main points, facts of numerical data by showing the number of data points that fall within a specified range of values (called “bins”). It is similar to a vertical bar graph.

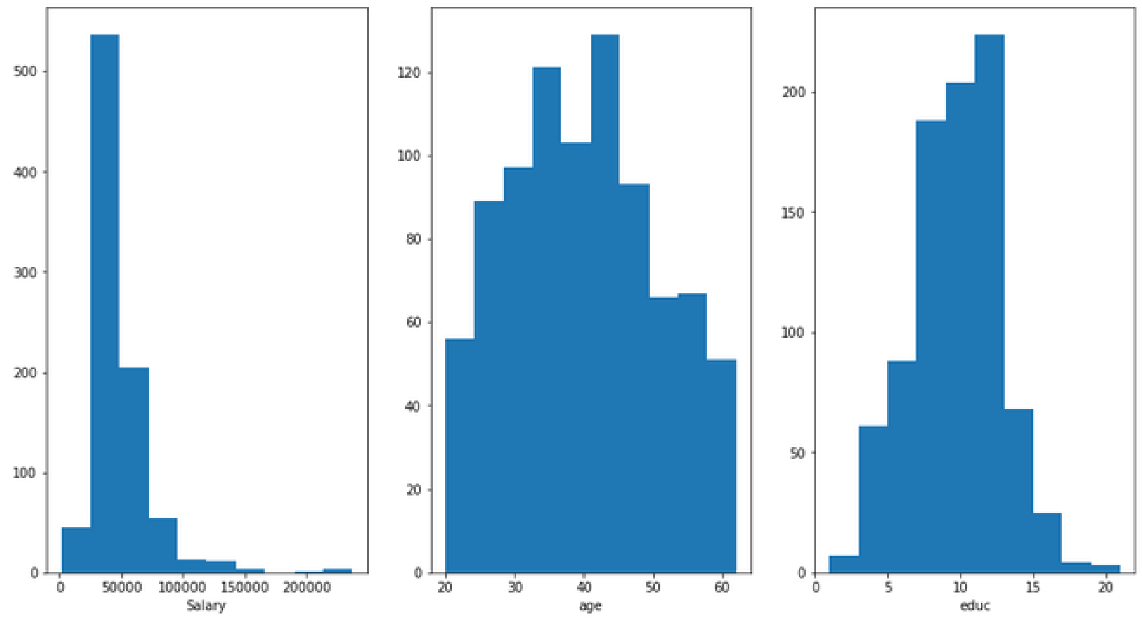


Figure 1.2: Histogram of salary, age and educ.

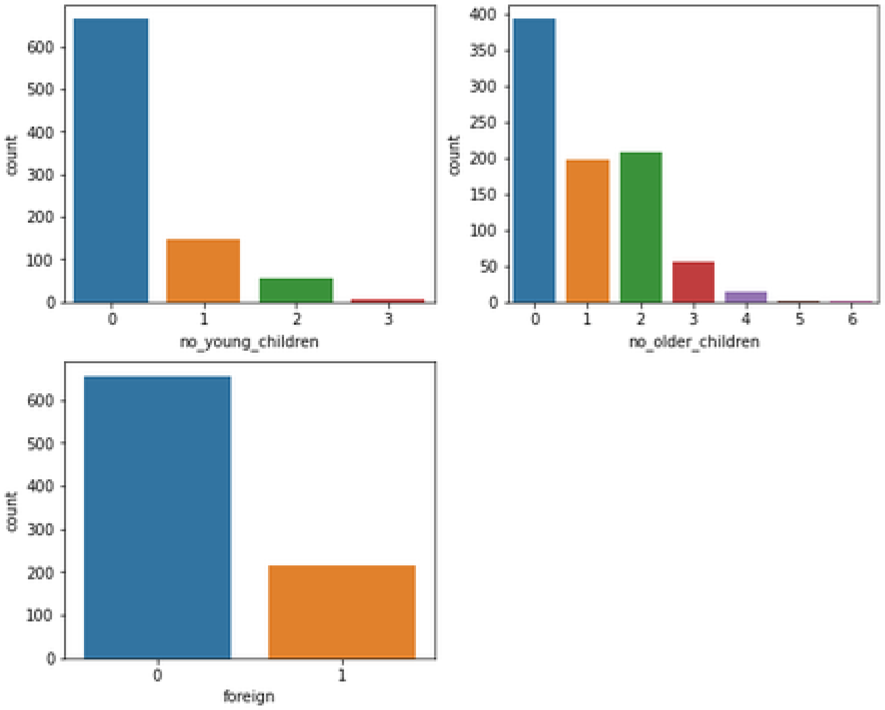


Figure 1.3: Histogram of no_young_children, no_older_children and foreign.

Box Plot:

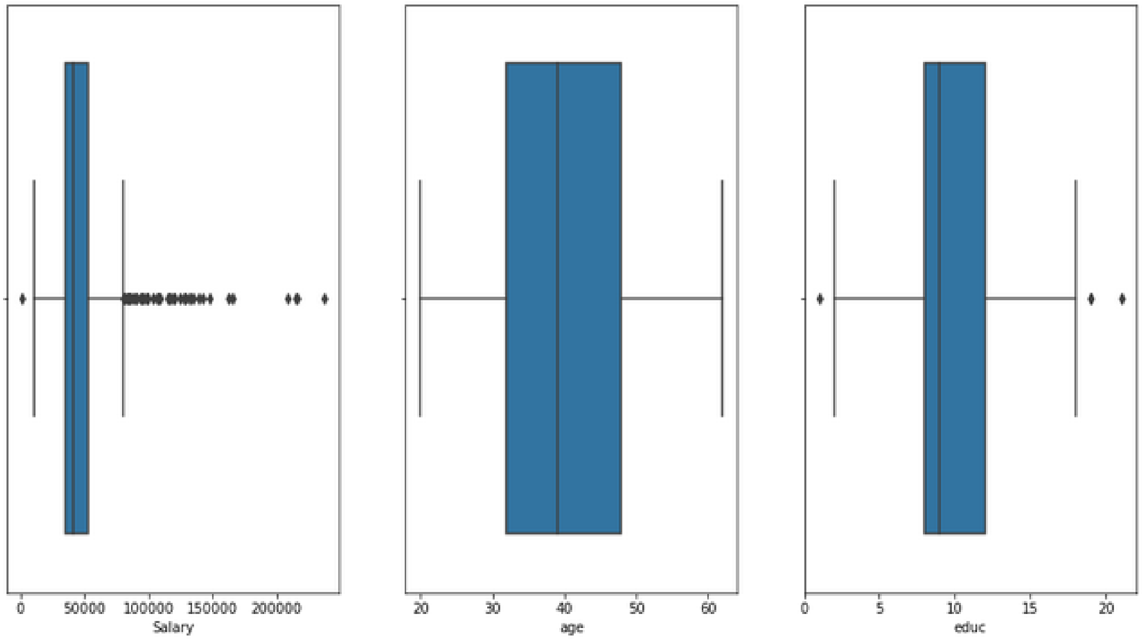


Figure 1.4: Box plots of salary, age and educ.

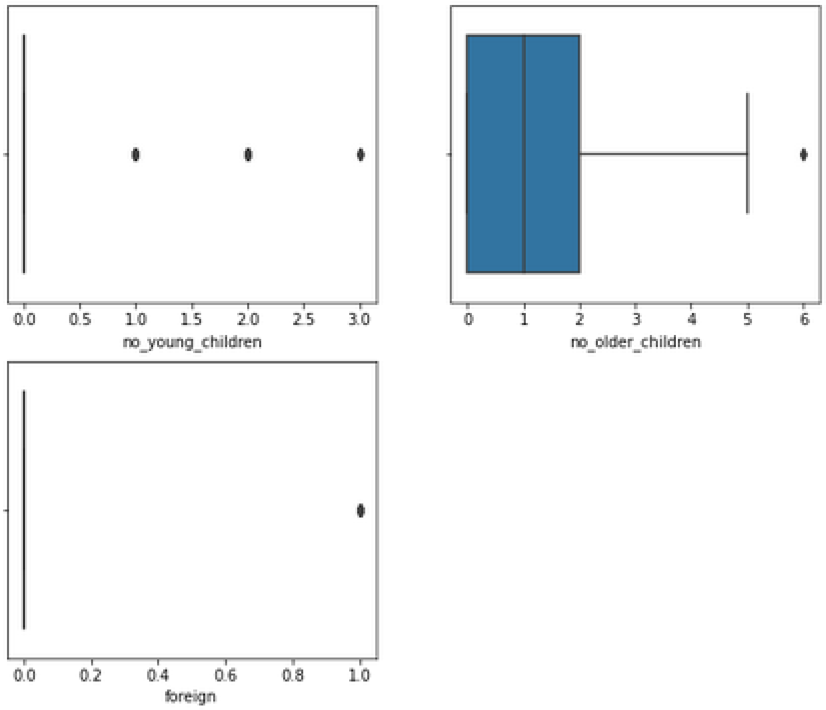
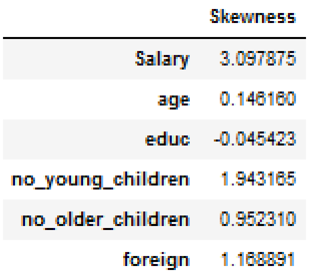


Figure 1.5: Box plots of no_young_children, no_older_children and foreign.

- From the box plot we can see all the variables have outliers.
- Variable age has no outliers.
- Variable educ, no_older_children and foreign has least number of outliers.
- From the histogram and box plots we can see that, salary, educ, no_young_children, no_older_children and foreign. Have positive skewness.
- age shows least positive skewness or negative skewness and looks almost like normally distributed.
- no_young_children and foreign shows the highest positive skewness.

Skewness:

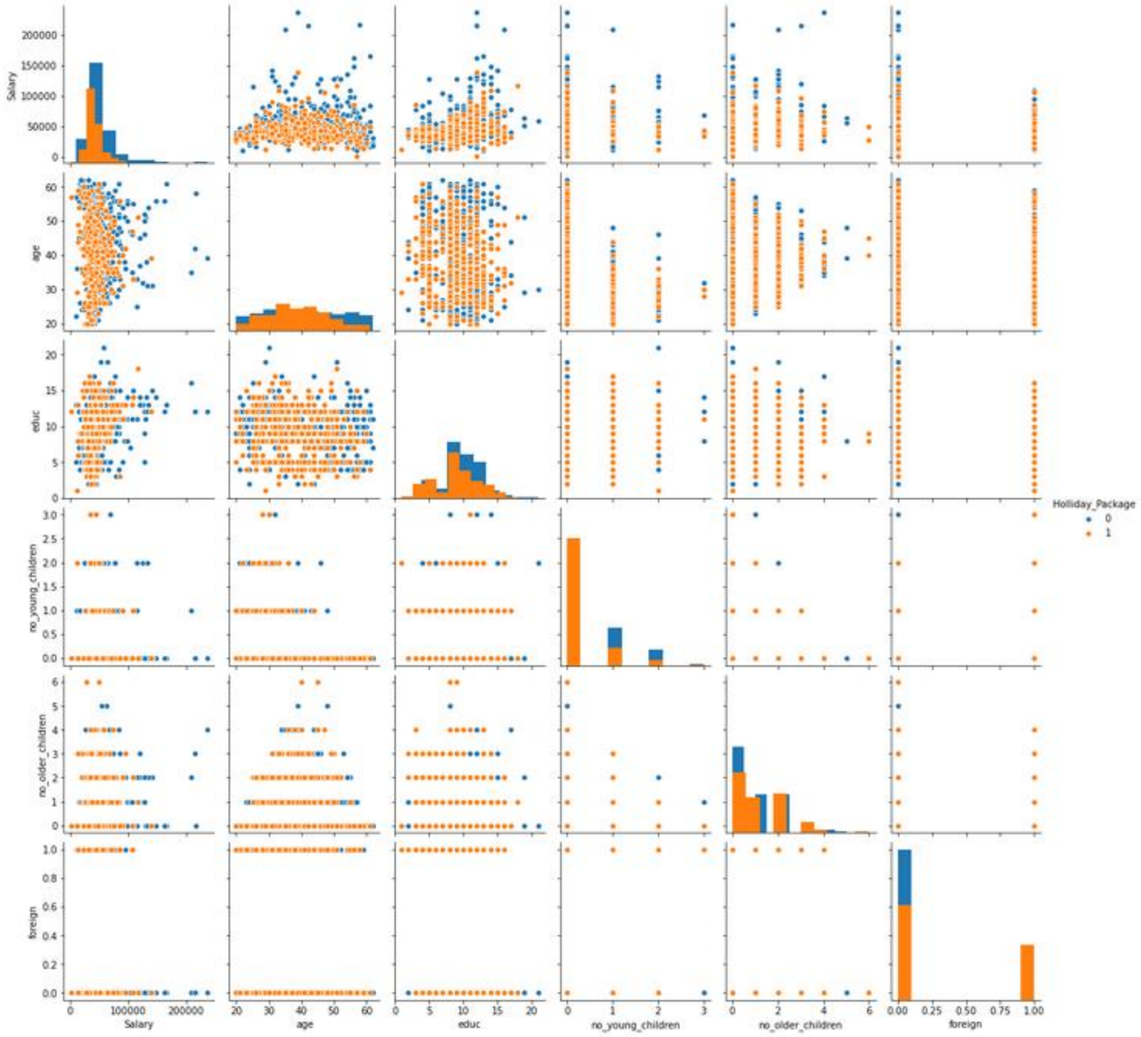


	Skewness
Salary	3.097875
age	0.146160
educ	-0.045423
no_young_children	1.943165
no_older_children	0.952310
foreign	1.168891

Multivariate Analysis

Pair plot:

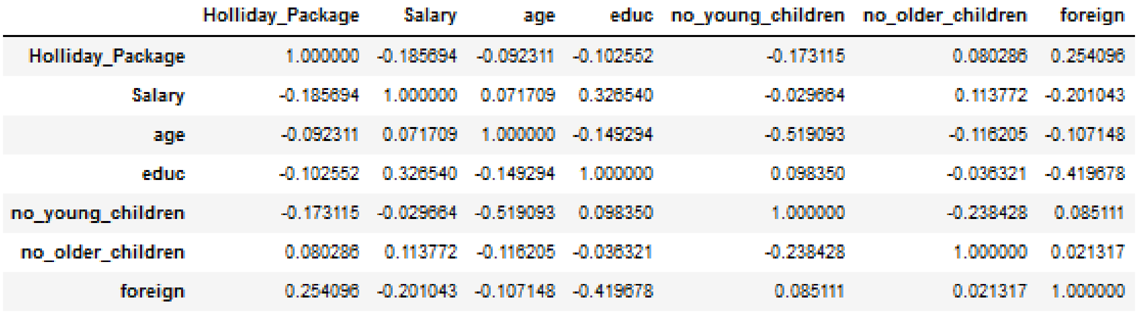
Figure 1.6: Pair plot for the data set



- In the above plot scatter diagrams are plotted for all the numerical columns in the dataset. A scatter plot is a visual representation of the degree of correlation between any two columns.
- The pair plot function in seaborn makes it very easy to generate joint scatter plots for all the columns in the data.
- Pair plot shows scatter plot as well as the histogram between all the variables of the dataset.

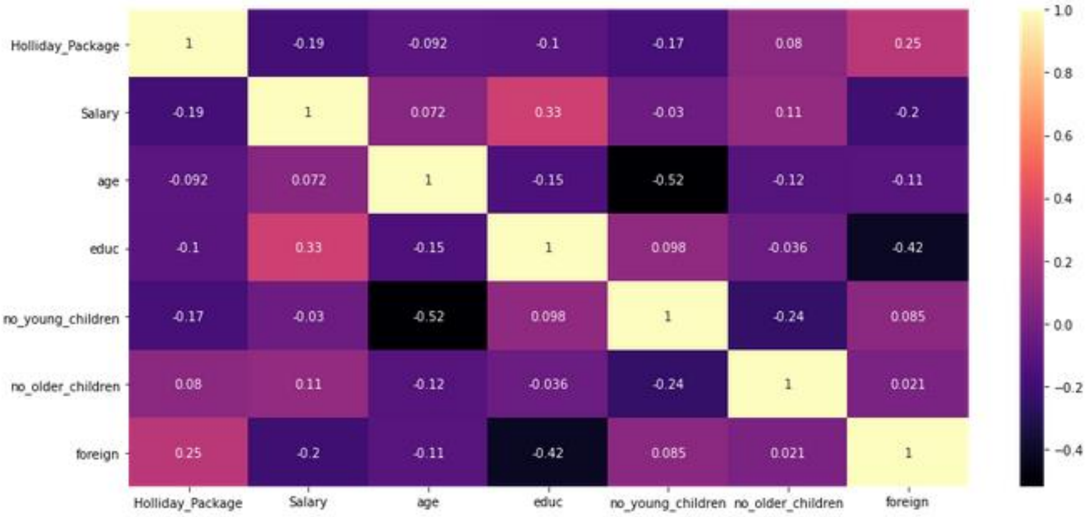
Correlation Matrix

Table 1.4: Correlation matrix of the Holiday Package Dataset



	Holliday_Package	Salary	age	educ	no_young_children	no_older_children	foreign
Holliday_Package	1.000000	-0.185694	-0.092311	-0.102552	-0.173115	0.080286	0.254096
Salary	-0.185694	1.000000	0.071709	0.326540	-0.029664	0.113772	-0.201043
age	-0.092311	0.071709	1.000000	-0.149294	-0.519093	-0.116205	-0.107148
educ	-0.102552	0.326540	-0.149294	1.000000	0.098350	-0.036321	-0.419678
no_young_children	-0.173115	-0.029664	-0.519093	0.098350	1.000000	-0.238428	0.085111
no_older_children	0.080286	0.113772	-0.116205	-0.036321	-0.238428	1.000000	0.021317
foreign	0.254096	-0.201043	-0.107148	-0.419678	0.085111	0.021317	1.000000

Figure 1.7: Heat map of the Holiday package Dataset



- We use the heatmap to check the correlation in visual and coloured manner. The colour helps with picking out the most correlated easily. The darker the colour, higher the correlation.

- Age, educ, no_young_children and foreign shows highest positive correlation of 0.92, they are highly co related to each other. So depending on the age, education number of children and foreign trips, employees are the once who opt for the package.
- Age and educ shows positive correlation of 0.098, hence the company should focus on these two variables.
- No of young children, age and educ shows positive correlation of 0.098 which shows employees with younger the children are more likely to opt for the holiday package.

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:

Logistic Regression:

Logistic regression is a fundamental classification technique. It belongs to the group of linear classifiers and is somewhat similar to polynomial and **linear regression**. Logistic regression is fast and relatively uncomplicated, and it's convenient for you to interpret the results. Although it's essentially a method for binary classification, it can also be applied to multiclass problems.

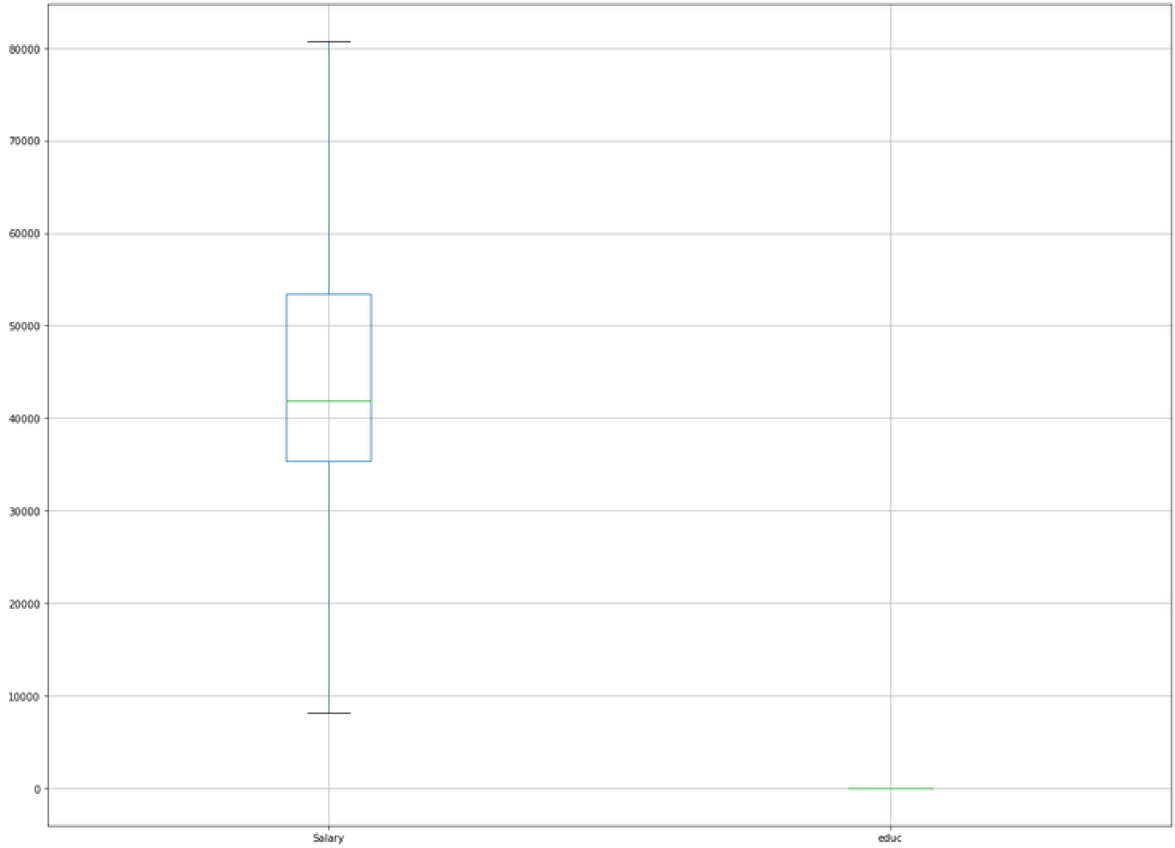
Classification in regression model:

Classification is a very important area of supervised machine learning. A large number of important machine learning problems fall within this area. There are many classification methods, and logistic regression is one of them.

Removing outliers:

- The variable salary and educ had outliers which are removed in the below figure. Now we have data without outliers.
- Other variables don't have any outliers.
- Data outliers can deceive the training process resulting in longer training times and less accurate models. Outliers are defined as samples that are significantly different from the remaining data.
- We are not scaling the dare as informed in the question.

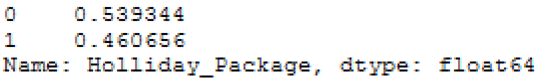
- After imputing, we will predict the train dependent variable and test dependent variable using train independent variables and test independent variables.
- After this we will check the performance metrics like classification report, AUC score and ROC Score.
- We will also be doing the Grid Search CV , where CV stands for Cross validation. This step is required to get the best parameter result for the model.



Train and Test split: Logistics regression:

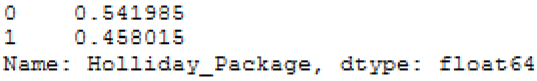
- Here we are splitting the train and test data to fit the logistic regression model and LDA model.
  The train model is performing 0.53 for 0 and 0.46 for 1.
- The test mode is performing 0.54 for 0 and 0.45 for 1.

Train data:



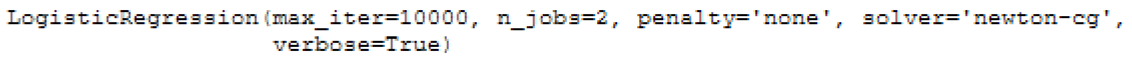
```
0    0.539344
1    0.460656
Name: Holliday_Package, dtype: float64
```

Test data:



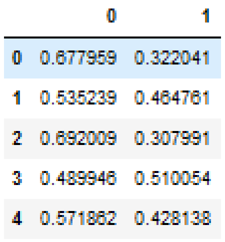
```
0    0.541985
1    0.458015
Name: Holliday_Package, dtype: float64
```

Fit the Logistic Regression model:



```
LogisticRegression(max_iter=10000, n_jobs=2, penalty='none', solver='newton-cg',
                   verbose=True)
```

Getting the Predicted Classes and Probs:



	0	1
0	0.677959	0.322041
1	0.535239	0.464761
2	0.692009	0.307991
3	0.489946	0.510054
4	0.571862	0.428138

The predicted classes and probs are derived in the above table which varies from 0.67 to 0.48.

LDA:

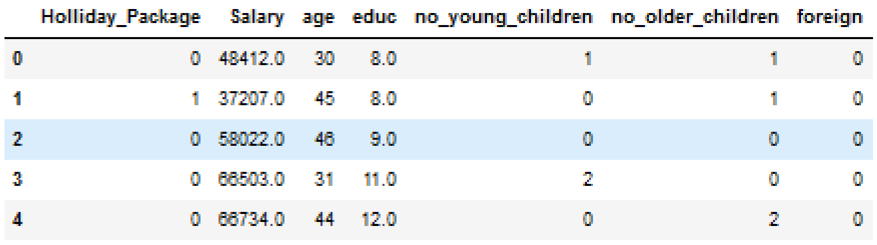
Linear Discriminant Analysis

A classifier with a linear decision boundary, generated by fitting class conditional densities to the data and using Bayes' rule.

The model fits a Gaussian density to each class, assuming that all classes share the same covariance matrix.

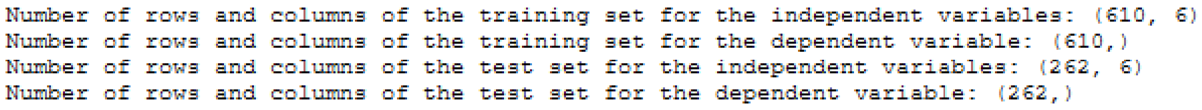
The fitted model can also be used to reduce the dimensionality of the input by projecting it to the most discriminative directions, using the transform method.

Data head



	Holliday_Package	Salary	age	educ	no_young_children	no_older_children	foreign
0	0	48412.0	30	8.0	1	1	0
1	1	37207.0	45	8.0	0	1	0
2	0	58022.0	46	9.0	0	0	0
3	0	66503.0	31	11.0	2	0	0
4	0	66734.0	44	12.0	0	2	0

Train test split:

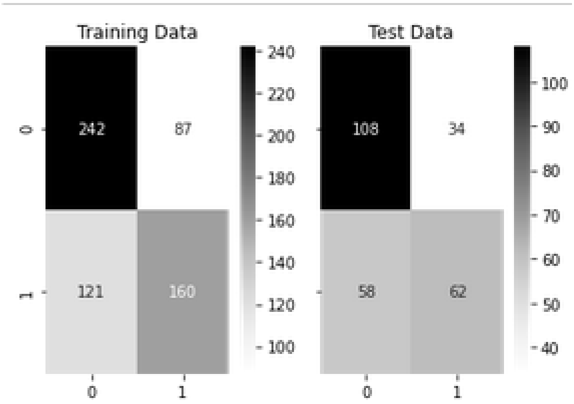


```
Number of rows and columns of the training set for the independent variables: (610, 6)
Number of rows and columns of the training set for the dependent variable: (610,)
Number of rows and columns of the test set for the independent variables: (262, 6)
Number of rows and columns of the test set for the dependent variable: (262,)
```

Performing LDA on the data:

Test Data Probability Prediction

Figure. 1.7 Heatmap for Linear discriminant analysis.



Training Data Probability Prediction:

Testing Data Probability Prediction:

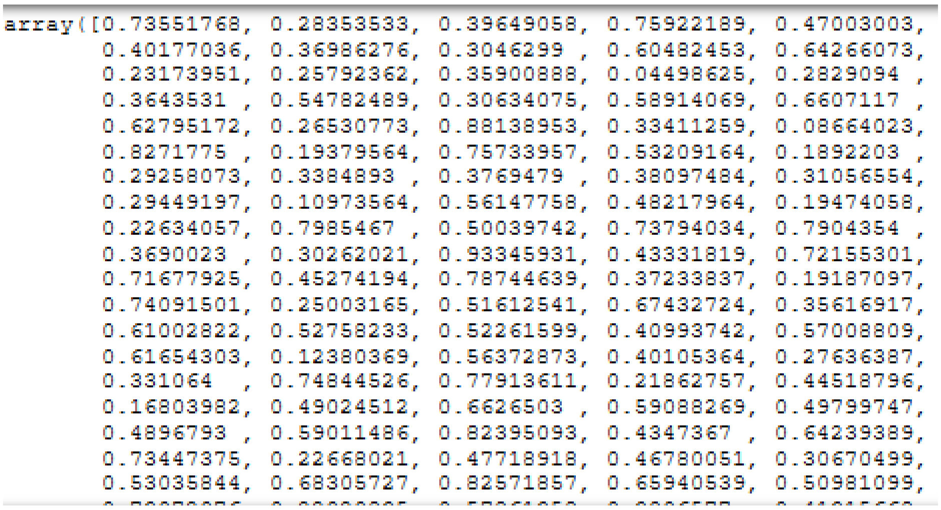
- The training and testing probability are shown in the above heatmap for linear discriminant analysis.
- Training data heatmap is predicting the values at 242 and 87 for 0.

  121 and 160 for 1.

- Testing data heatmap is predicting the values at 108 and 34 for 0.

  58 and 62 for for 1.

Predicting Probability Train data:

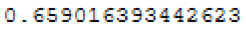


```
array([0.73551768, 0.28353533, 0.39649058, 0.75922189, 0.47003003,
       0.40177036, 0.36986276, 0.3046299 , 0.60482453, 0.64266073,
       0.23173951, 0.25792362, 0.35900888, 0.04498625, 0.2829094 ,
       0.3643531 , 0.54782489, 0.30634075, 0.58914069, 0.6607117 ,
       0.62795172, 0.26530773, 0.88138953, 0.33411259, 0.08664023,
       0.8271775 , 0.19379564, 0.75733957, 0.53209164, 0.1892203 ,
       0.29258073, 0.3384893 , 0.3769479 , 0.38097484, 0.31056554,
       0.29449197, 0.10973564, 0.56147758, 0.48217964, 0.19474058,
       0.22634057, 0.7985467 , 0.50039742, 0.73794034, 0.7904354 ,
       0.3690023 , 0.30262021, 0.93345931, 0.43331819, 0.72155301,
       0.71677925, 0.45274194, 0.78744639, 0.37233837, 0.19187097,
       0.74091501, 0.25003165, 0.51612541, 0.67432724, 0.35616917,
       0.61002822, 0.52758233, 0.52261599, 0.40993742, 0.57008809,
       0.61654303, 0.12380369, 0.56372873, 0.40105364, 0.27636387,
       0.331064  , 0.74844526, 0.77913611, 0.21862757, 0.44518796,
       0.16803982, 0.49024512, 0.6626503 , 0.59088269, 0.49799747,
       0.4896793 , 0.59011486, 0.82395093, 0.4347367 , 0.64239389,
       0.73447375, 0.22668021, 0.47718918, 0.46780051, 0.30670499,
       0.53035844, 0.68305727, 0.82571857, 0.65940539, 0.50981099,
```

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.

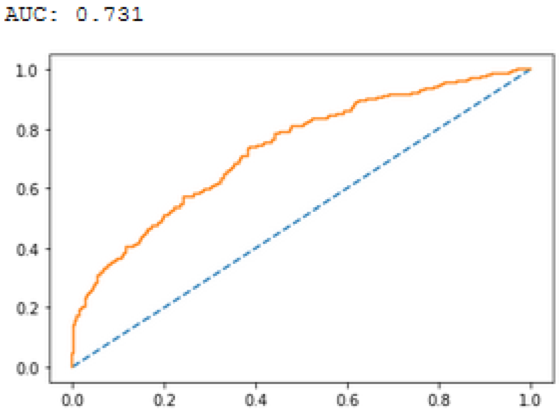
Answer:

Accuracy - Training Data model score and AUC



```
0.659016393442623
```

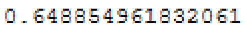
Figure 1.8 AUC score for the train data.



```
AUC: 0.731
```

- AUC score for Training model is **0.73.**
- Accuracy for training data is **0.65**. This shows the model is not performing great.

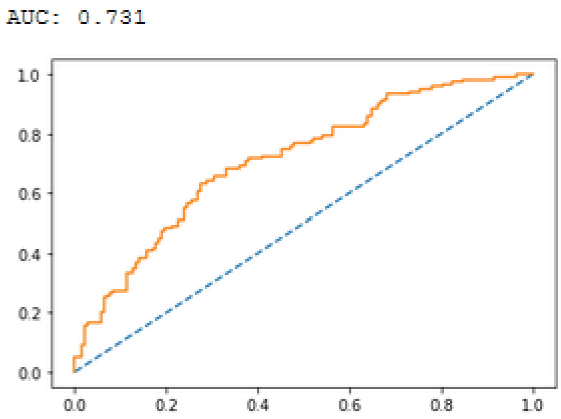
Accuracy - Testing Data model score and AUC



```
0.648854961832061
```

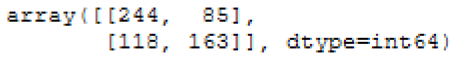
Figure 1.8 AUC score for the test data.

- AUC score for Testing model is **0.73.**
- Accuracy for Testing data is **0.64**. This shows the model is not performing very well.



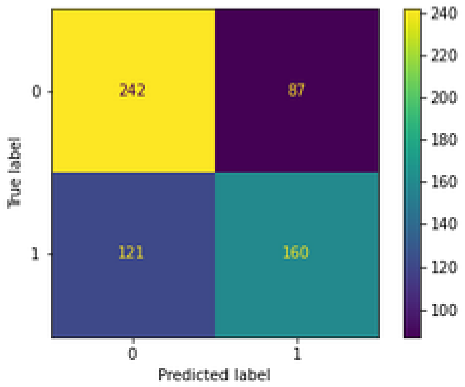
```
AUC: 0.731
```

Confusion matrix for y_train and ytrain_Predict

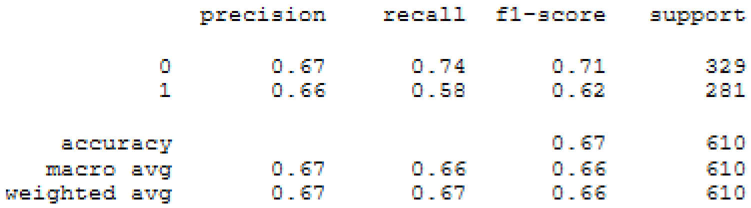


```
array([[244,  85],
       [118, 163]], dtype=int64)
```

Figure 1.9 Confusion matrix x_train and y_train data

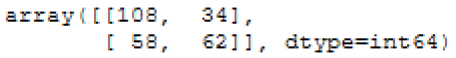


Classification report y_train and ytrain_predict



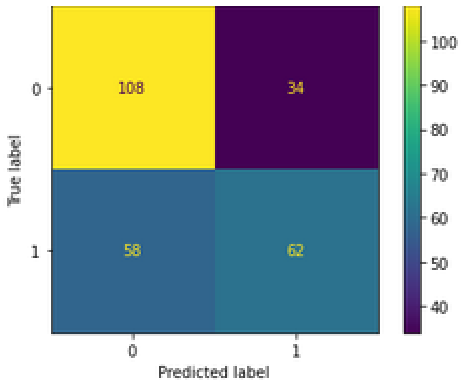
```
              precision    recall  f1-score   support

           0       0.67      0.74      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      0.66       610
```

Confusion matrix for y_test and ytest_predict

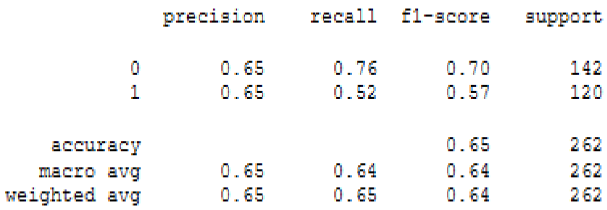


```
array([[108,  34],
       [ 58,  62]], dtype=int64)
```

Figure 1.9 Confusion matrix y_test and ytest_predict



Classification report y_test and ytest_predict

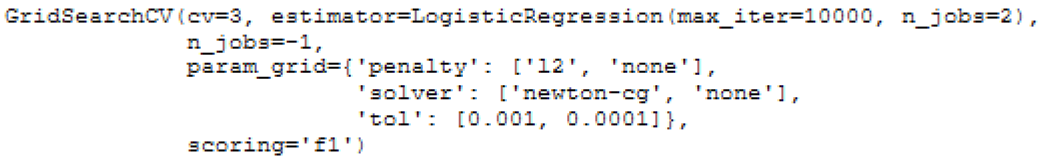


```
              precision    recall  f1-score   support

           0       0.65      0.76      0.70       142
           1       0.65      0.52      0.57       120

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

Confusion matrix **x_train , y_train and y_test ,ytest_predict**

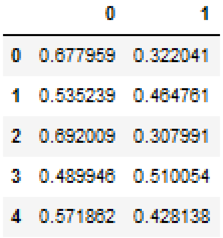
- Accuracy for the above train data set is 0.65.
- Macro avg of precision 0.65, recall 0.64 and F1 score of 0.64.
- Weighted average of precision 0.65, recall 0.64 and F1 score of 0.64.
- Performance metrics are determined using best parameter values prediction that we got because recall and precision are low.

GridSearchCV for X_train and y_train



```
GridSearchCV(cv=3, estimator=LogisticRegression(max_iter=10000, n_jobs=2),
             n_jobs=-1,
             param_grid={'penalty': ['l2', 'none'],
                         'solver': ['newton-cg', 'none'],
                         'tol': [0.001, 0.0001]},
             scoring='f1')
```

Prediction on the training and testing data set:



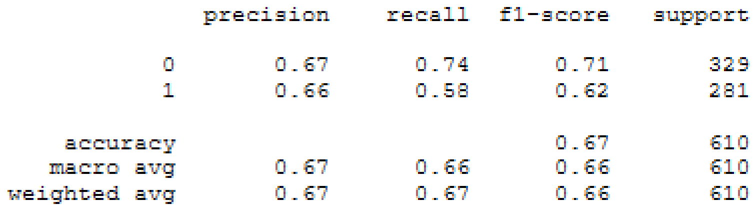
	0	1
0	0.677959	0.322041
1	0.535239	0.464761
2	0.692009	0.307991
3	0.489946	0.510054
4	0.571862	0.428138

Prediction of the training and testing data set after grid search cv is in between 0.69 to 0.57 for 0

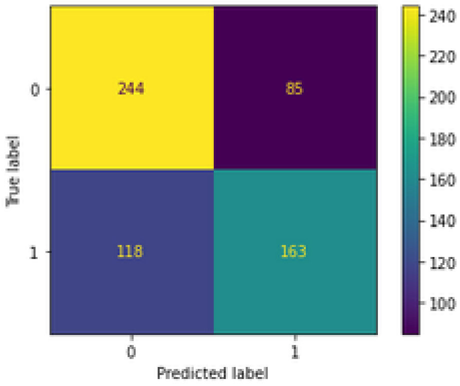
And 0.30 to 0.51 for 1.

Confusion matrix on the training and testing data:

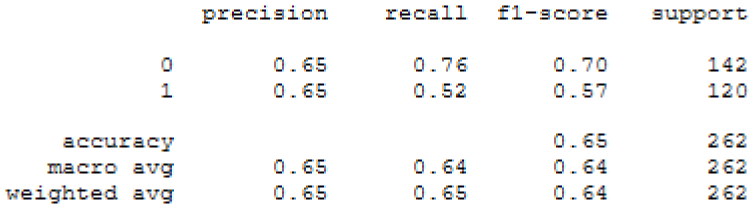
X_train,y_train



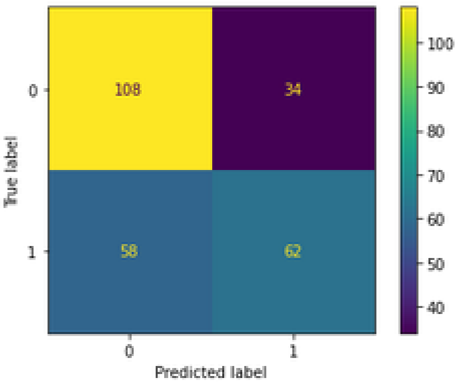
```
              precision    recall  f1-score   support

           0       0.67      0.74      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      0.66       610
```



X_test,y_test



```
              precision    recall  f1-score   support

           0       0.65      0.76      0.70       142
           1       0.65      0.52      0.57       120

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

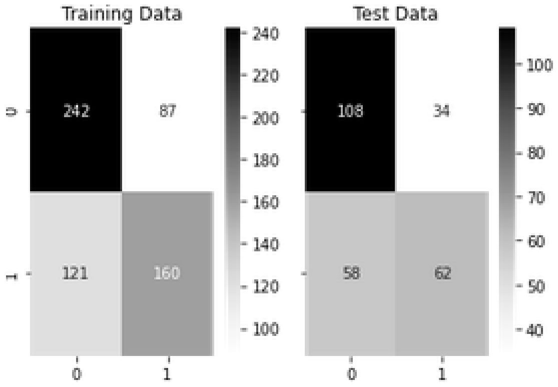


Accuracy for x train and y train is **0.67**

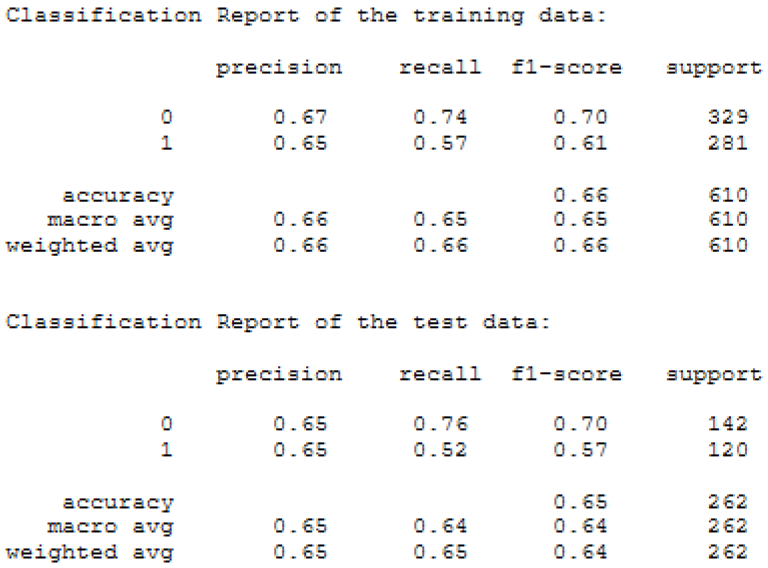
Accuracy of x test and y test is **0.65**

LDA model evaluation

Plotting confusion matrix for the different models for the Training Data



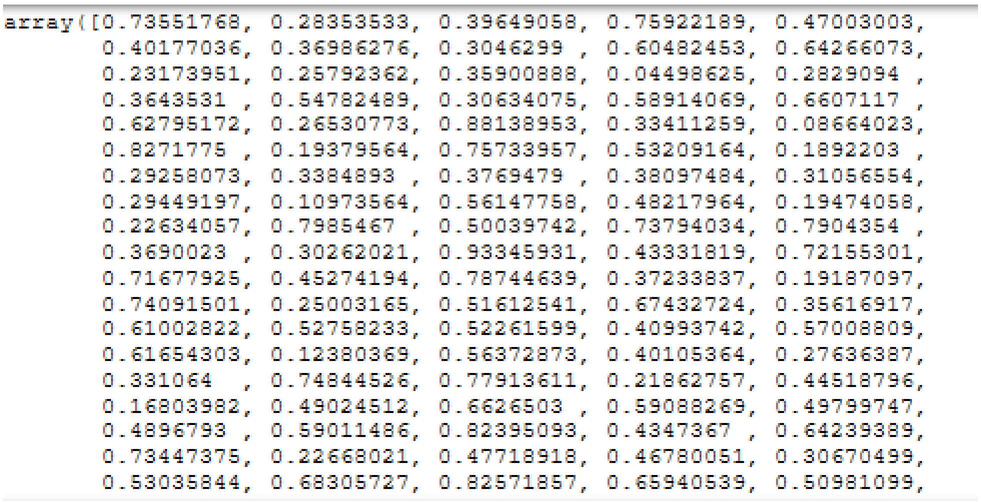
Classification report of train and test data:



```
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      0.65      0.65       610
weighted avg       0.66      0.66      0.66       610


Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.65      0.76      0.70       142
           1       0.65      0.52      0.57       120

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

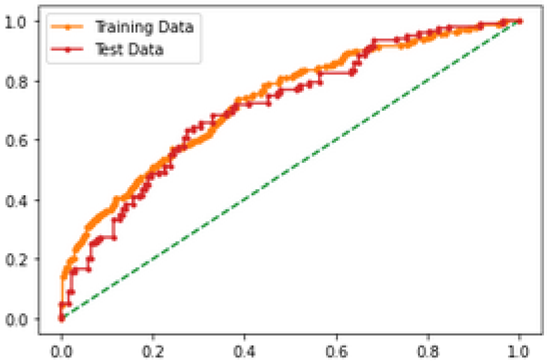
Training and Testing Data Probability Prediction

- The Linear Discriminant Analysis is a simple linear machine learning algorithm for classification.
- How to fit, evaluate, and make predictions with the Linear Discriminant Analysis model with Scikit-Learn.
- How to tune the hyperparameters of the Linear Discriminant Analysis algorithm on a given dataset.
- Accuracy of 0.66 for training data.
- Accuracy of 0.65 for test data.



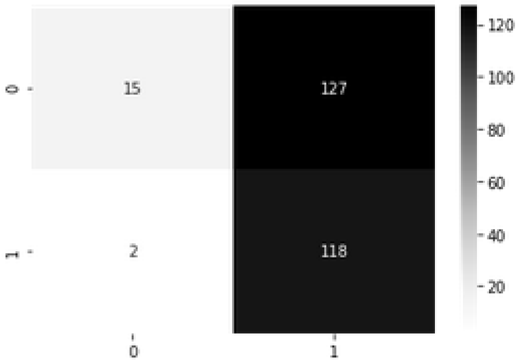
```
array([0.73551768, 0.28353533, 0.39649058, 0.75922189, 0.47003003,
       0.40177036, 0.36986276, 0.3046299 , 0.60482453, 0.64266073,
       0.23173951, 0.25792362, 0.35900888, 0.04498625, 0.2829094 ,
       0.3643531 , 0.54782489, 0.30634075, 0.58914069, 0.6607117 ,
       0.62795172, 0.26530773, 0.88138953, 0.33411259, 0.08664023,
       0.8271775 , 0.19379564, 0.75733957, 0.53209164, 0.1892203 ,
       0.29258073, 0.3384893 , 0.3769479 , 0.38097484, 0.31056554,
       0.29449197, 0.10973564, 0.56147758, 0.48217964, 0.19474058,
       0.22634057, 0.7985467 , 0.50039742, 0.73794034, 0.7904354 ,
       0.3690023 , 0.30262021, 0.93345931, 0.43331819, 0.72155301,
       0.71677925, 0.45274194, 0.78744639, 0.37233837, 0.19187097,
       0.74091501, 0.25003165, 0.51612541, 0.67432724, 0.35616917,
       0.61002822, 0.52758233, 0.52261599, 0.40993742, 0.57008809,
       0.61654303, 0.12380369, 0.56372873, 0.40105364, 0.27636387,
       0.331064  , 0.74844526, 0.77913611, 0.21862757, 0.44518796,
       0.16803982, 0.49024512, 0.6626503 , 0.59088269, 0.49799747,
       0.4896793 , 0.59011486, 0.82395093, 0.4347367 , 0.64239389,
       0.73447375, 0.22668021, 0.47718918, 0.46780051, 0.30670499,
       0.53035844, 0.68305727, 0.82571857, 0.65940539, 0.50981099,
```

```
AUC for the Training Data: 0.731
AUC for the Test Data: 0.713
```

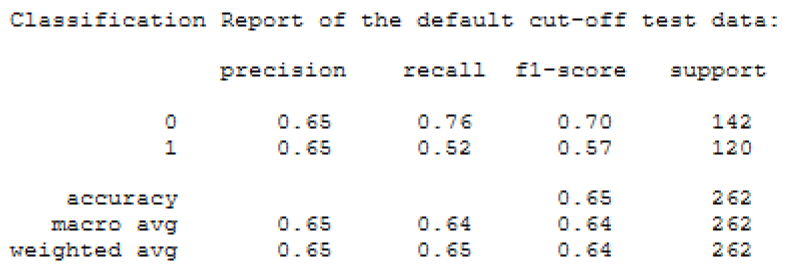


Predicting the classes on the test data

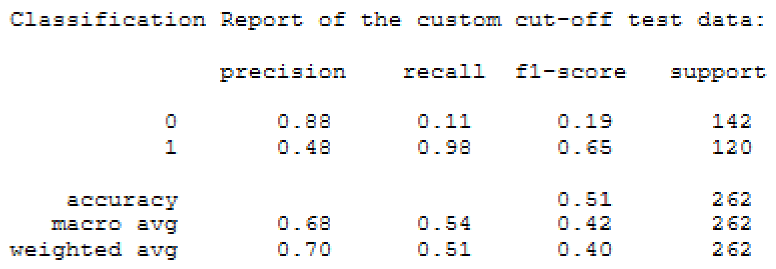
Heatmap for all the confusion matrix.



Classification report for dafult cut off test and custom cut- off train data:



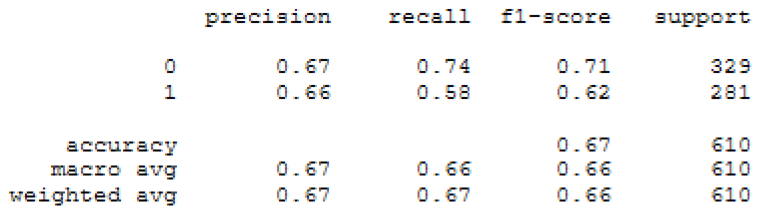
```
Classification Report of the default cut-off test data:

              precision    recall  f1-score   support

           0       0.65      0.76      0.70       142
           1       0.65      0.52      0.57       120

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



```
Classification Report of the custom cut-off test data:

              precision    recall  f1-score   support

           0       0.88      0.11      0.19       142
           1       0.48      0.98      0.65       120

    accuracy                           0.51       262
   macro avg       0.68      0.54      0.42       262
weighted avg       0.70      0.51      0.40       262
```

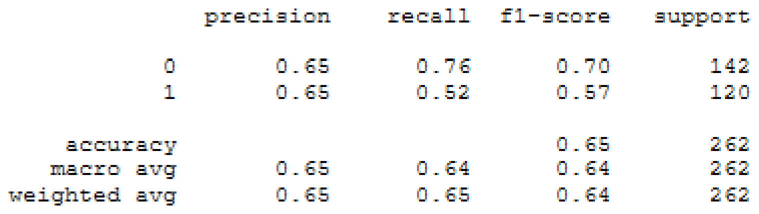
Comparison between Logistic regresstion and Linear Discriminant Ananlysis.

Classification Report of the X_Test and y_test



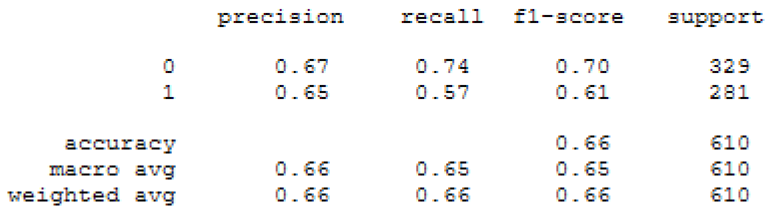
```
              precision    recall  f1-score   support

           0       0.67      0.74      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      0.66       610
```

Classification Repory of Y_test and ytest_predict

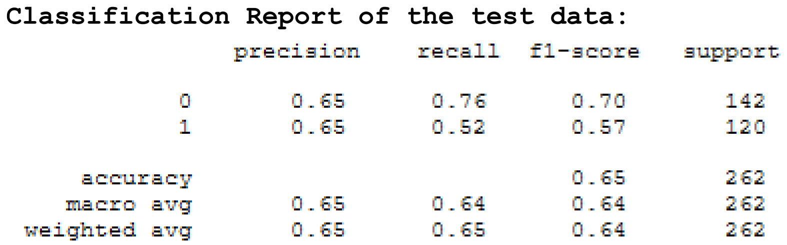


```
              precision    recall  f1-score   support

           0       0.65      0.76      0.70       142
           1       0.65      0.52      0.57       120

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

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      0.65      0.65       610
weighted avg       0.66      0.66      0.66       610
```



```
Classification Report of the test data:
              precision    recall  f1-score   support

           0       0.65      0.76      0.70       142
           1       0.65      0.52      0.57       120

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

Comparison between Logistic regresstion and Linear Discriminant Ananlysis.

- As per the above classification report all the values like accuracy, precision, recall and f1 score are almost equal.
- Hence considering the F1 score which is higher for the LDA model we can consider it a better model when compared to Logistic regression model.
- The F1 score for the LDA model is 0.71 and 0.70 for the logistic regresstion model.
- Hence the LDA is preferred for modeling for logistic regression model.

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

Answer:

Based on predictions we would like to use the LDA model as it is slightly better than the logistic regression model.

The above model values are almost same for most of the parameters.

Hence, I will be considering the F1 score for the linear discriminant model being better.

Here the data analysis for the business will be the employees with mid age having less no of young children prefer foreign trips. When compared to the employees with older children. Hence there is corelation and positive nature for the above variables.

The company should focus on employees who are young and have one young child or no child, as the probability of them opting for the package is more than the employees who have older kids.

The Holiday package options should be increased so that the employees can choose for the options they are looking for within their budget.

The insights are business should look at the employee’s pattern who opt for the holiday package and offer them few goodies which will also make them opt for the holiday packages.

Business should have different foreign travel options which employees will opt for depending on the past history. More number of options gives the customer more options to opt for the package.