Business Report
Predictive Modeling

Arnab Ghosal

27 June 2021

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:

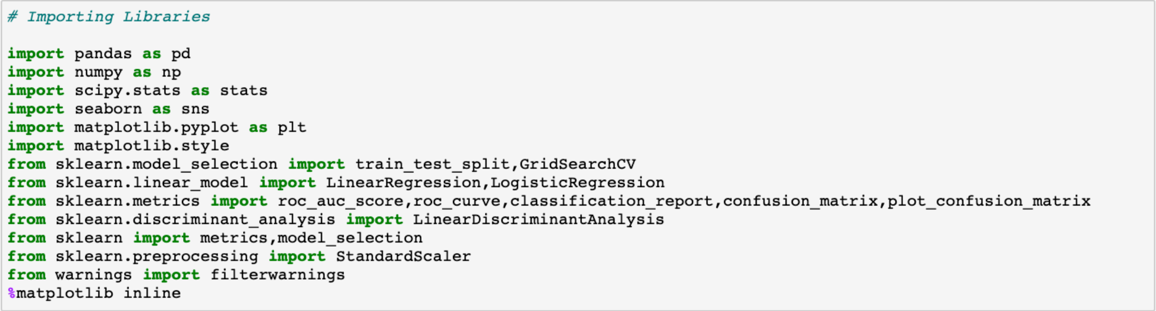
Variable Name	**Description**
Carat	Carat weight of the cubic zirconia.
Cut	Describe the cut quality of the cubic zirconia. Quality is increasing order Fair, Good, Very Good, Premium, Ideal.
Color	Colour of the cubic zirconia.With D being the best and J the worst.
Clarity	cubic zirconia Clarity refers to the absence of the Inclusions and Blemishes. (In order from Best to Worst, FL = flawless, I3= level 3 inclusions) FL, IF, VVS1, VVS2, VS1, VS2, SI1, SI2, I1, I2, I3
Depth	The Height of a 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.

Questions:

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.

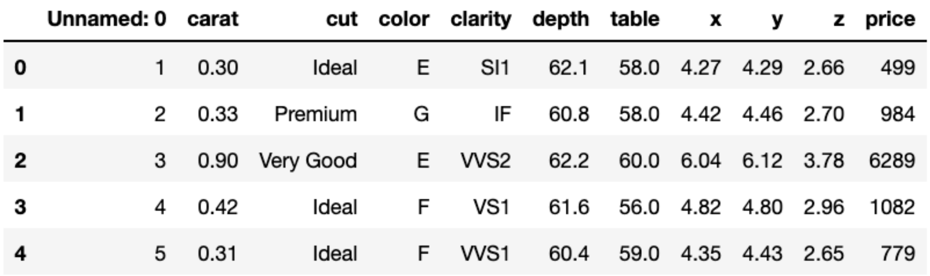
While reading the data and understanding to do exploratory data analysis we followed below steps likewise :

Importing necessary libraries :



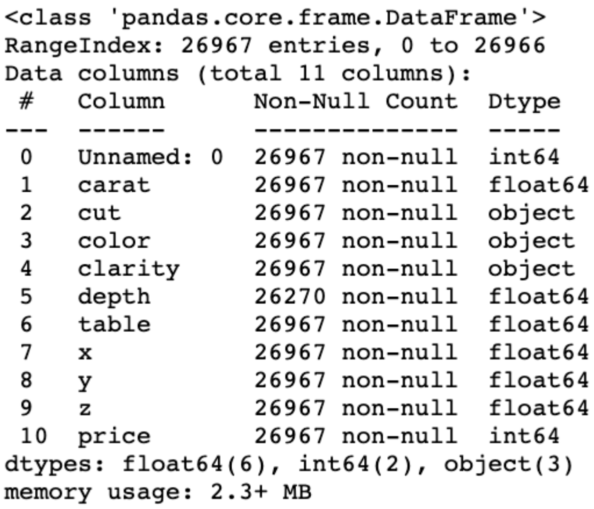
```
# Importing Libraries

import pandas as pd
import numpy as np
import scipy.stats as stats
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.style
from sklearn.model_selection import train_test_split,GridSearchCV
from sklearn.linear_model import LinearRegression,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 import metrics,model_selection
from sklearn.preprocessing import StandardScaler
from warnings import filterwarnings
%matplotlib inline
```

Read the dataset :



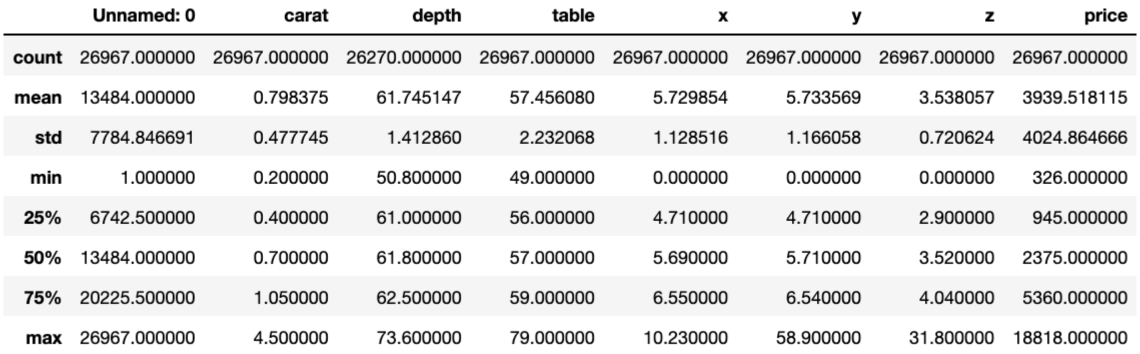
	Unnamed: 0	carat	cut	color	clarity	depth	table	x	y	z	price
0	1	0.30	Ideal	E	SI1	62.1	58.0	4.27	4.29	2.66	499
1	2	0.33	Premium	G	IF	60.8	58.0	4.42	4.46	2.70	984
2	3	0.90	Very Good	E	VVS2	62.2	60.0	6.04	6.12	3.78	6289
3	4	0.42	Ideal	F	VS1	61.6	56.0	4.82	4.80	2.96	1082
4	5	0.31	Ideal	F	VVS1	60.4	59.0	4.35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

Information of the data (data-types & row-column) :



```
<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
```

Descriptive summary of the data :



	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

Shape of the data (no. of row & columns available) :

We got to know there are 26967 rows & 11 columns available

Checking for Null values existence :

We checked the duplicate values by a method called **isnull()** and it returned **True** , that indicates this dataset has null values .

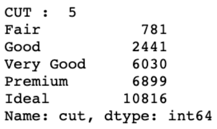
Now we wanted to trace further to know where all null values are existing in the dataset and it returned that column **"depth"** is having it.
(Please refer notebook for better clarity)

Checking for duplicate values existence :

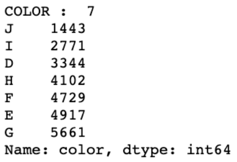
We checked the duplicate values by a method called **duplicated()** and it returned **False** , that indicates this dataset does not have any duplicate values .

Checking unique values in categorical columns:

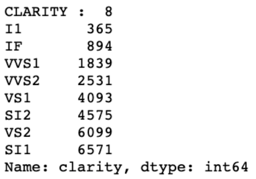
We wanted to check how many & what all unique values are existing in categorical columns:



```
CUT :  5
Fair           781
Good          2441
Very Good     6030
Premium       6899
Ideal        10816
Name: cut, dtype: int64
```

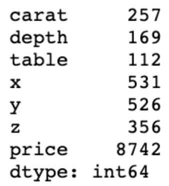


```
COLOR :  7
J    1443
I    2771
D    3344
H    4102
F    4729
E    4917
G    5661
Name: color, dtype: int64
```



```
CLARITY :  8
I1       365
IF       894
VVS1    1839
VVS2    2531
VS1     4093
SI2     4575
VS2     6099
SI1     6571
Name: clarity, dtype: int64
```

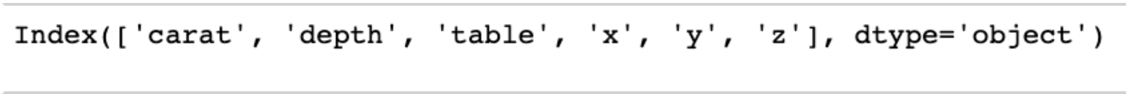
Checking unique values in remaining columns:



```
carat      257
depth      169
table      112
x          531
y          526
z          356
price     8742
dtype: int64
```

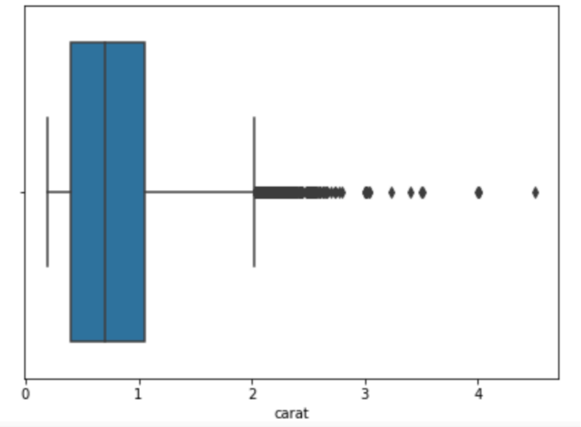
Extracting only continuous variables from the dataset

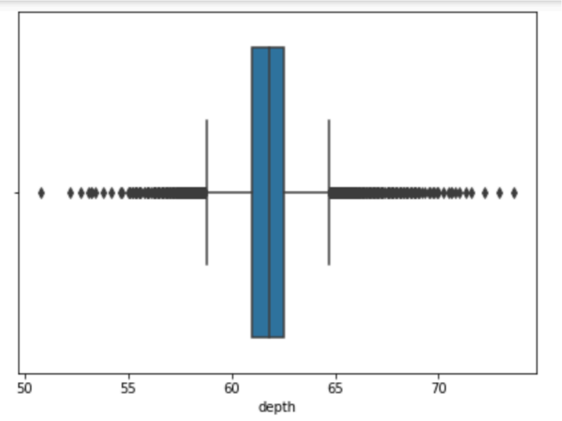
We wanted take out only numeric columns separately and it returned as below :

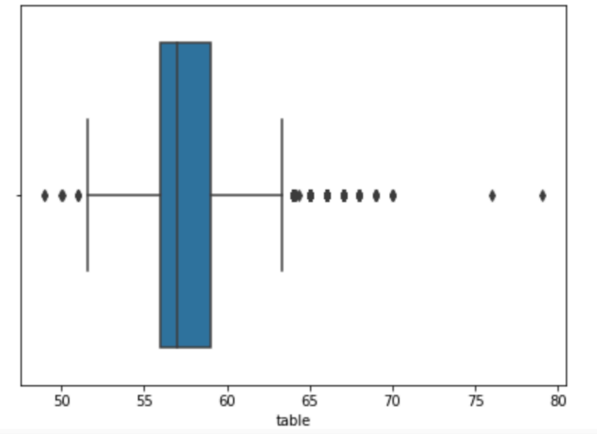


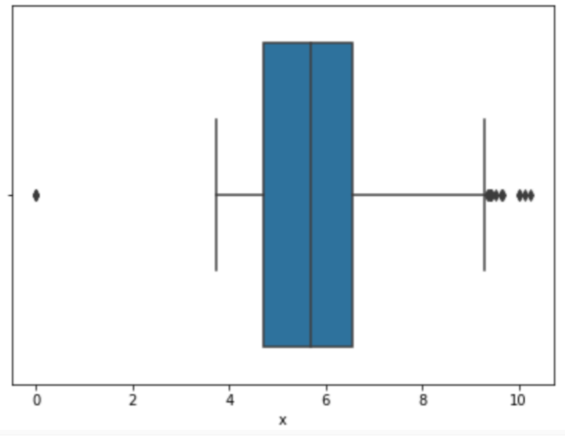
```
Index(['carat', 'depth', 'table', 'x', 'y', 'z'], dtype='object')
```

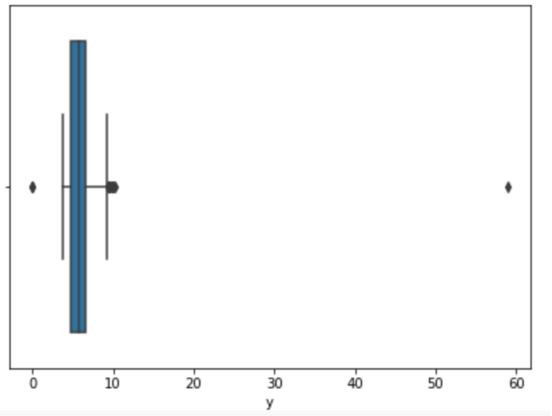
Checking for outliers (Excluded "Unnamed" as its used as index & "price" as its target variable in this dataset)

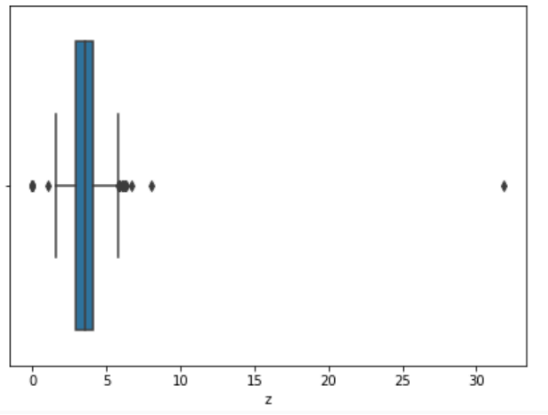












From the above analysis using Box-plot we can get to know that there as existence of outliers in this dataset but we have decided not to treat them or impute them as this data represents a dimensional information of diamonds , Columns are describing about the size of the diamond , weight , purity etc. where outliers existence is a practical and normal scenario.

Univariate Analysis :

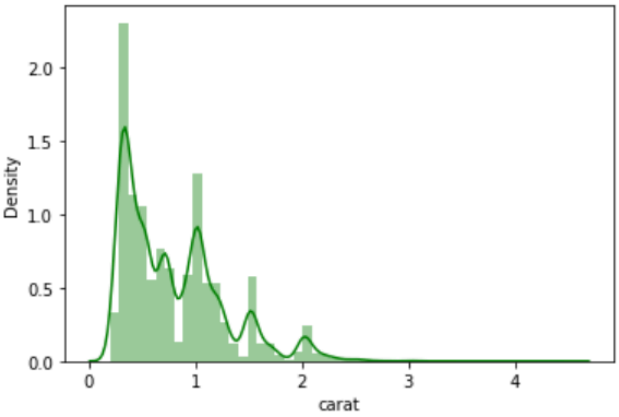
Univariate analysis is the simplest form of analysing data. “Uni” means “one”, so in other words our data takes only one variable at a time. It doesn't deal with causes or relationships (unlike regression) and it's major purpose is to describe; It takes data (individual column) separately , summarises that data and finds patterns in the data.

To evaluate univariate analysis we have segregated numeric & categorical columns.

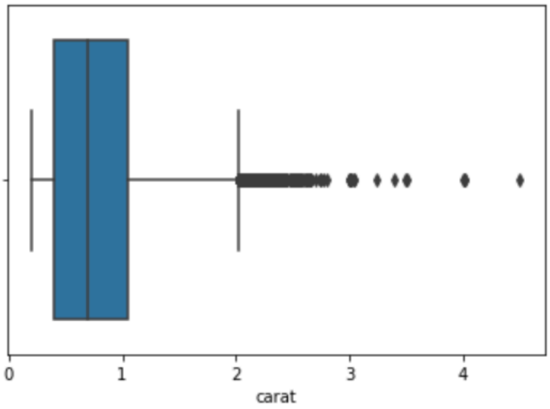
First , **we considered only numeric columns**
(Please refer notebook for better clarity)

```
Description of carat
----------------------------------------------------------------
----------------
count    26967.000000
mean         0.798375
std          0.477745
min          0.200000
25%          0.400000
50%          0.700000
75%          1.050000
max          4.500000
Name: carat, dtype: float64
----------------------------------------------------------------
----------------
```

Distribution of carat



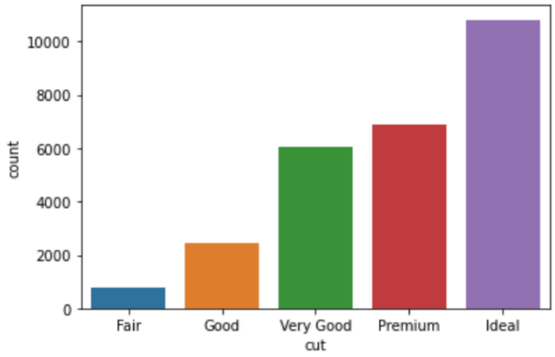
BoxPlot of carat

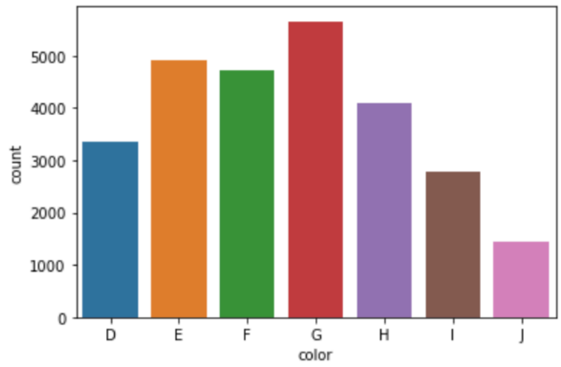


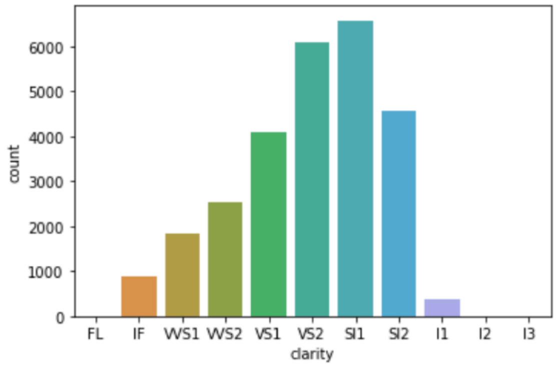
The output displays, total 6*3 = 18 distinct charts/columns & descriptions. Hence I have put the screenshot of only one variable which is **Carat** . **(Please refer the python notebook for better clarity**

While evaluating categorical columns we used **Count Plot.**

(Please refer the python notebook for better clarity)







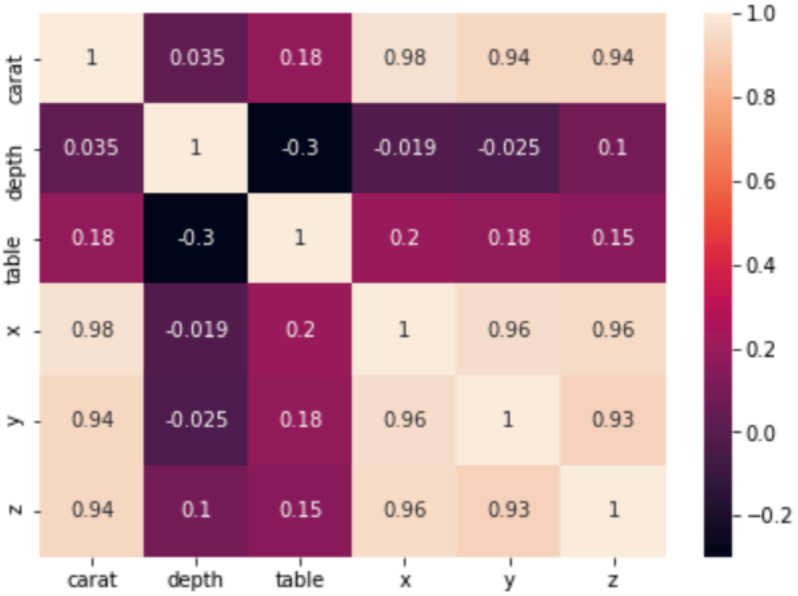
Bi-variate Analysis

As the name suggest it gives analysis of two specific variable at a time and we can derive correlation b/w them. We can use Heat-map for the same and further to highlight each correlated pair we can use scatter-plot / bar plot etc.

Similar to Univariate , we have done the Bi-variate analysis as well separately for categorical as well as numeric columns.

To evaluate numeric columns we have implemented **Heat-map** and tried analysing further through **scatter plot.**

(Please refer the python notebook for better clarity)



From the above Heat-map we can see few correlation b/w few variables , here **carat** is **highly correlated** with **x,y** & **z** . **x, y & z are correlated among themselves** . **Table & Depth are inversely correlated.**

Wanted depict the same through scatter plot.

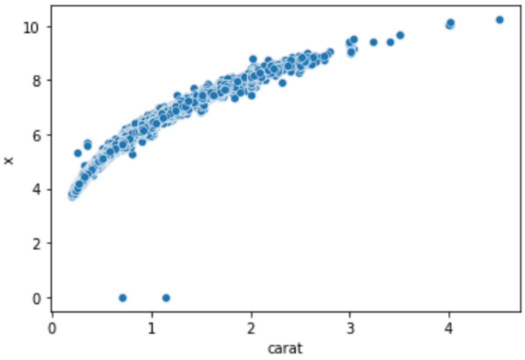
1. Carat & X

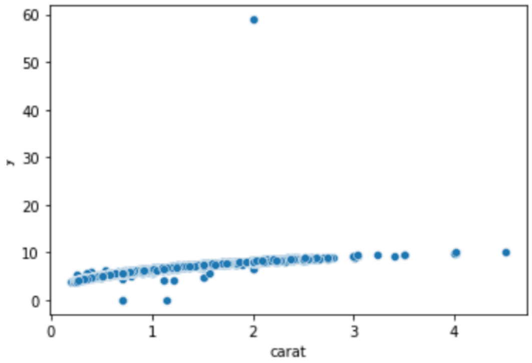
2. Carat & Y

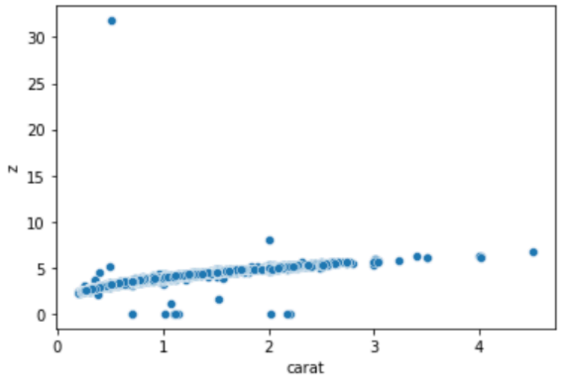
3. Carat & Z

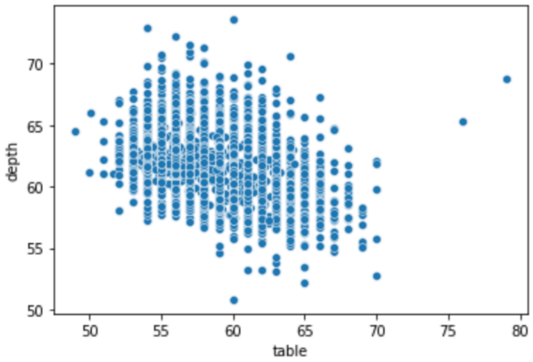
4. Depth & Table

Highlighted pairs are :

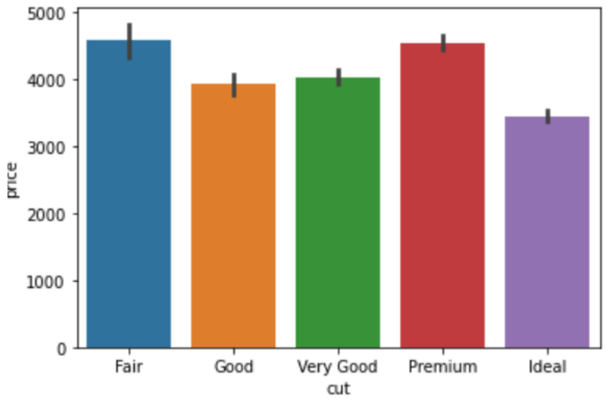


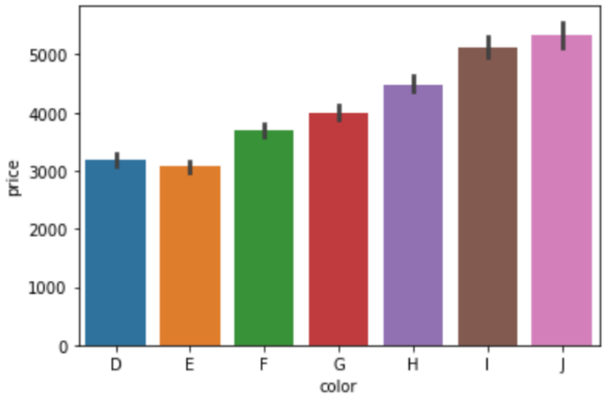


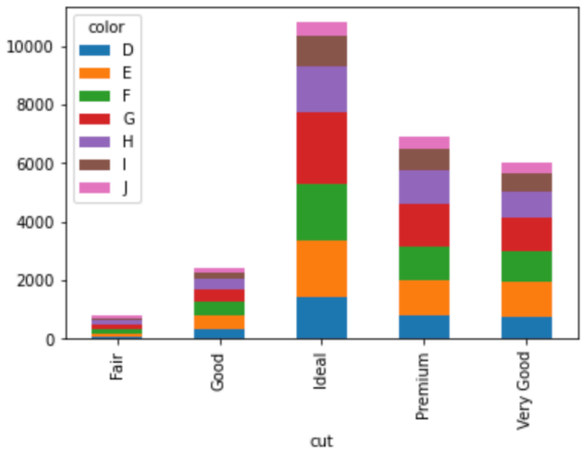


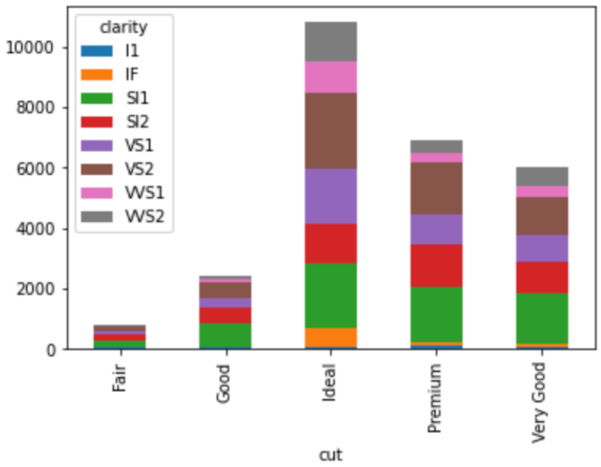


While evaluating categorical columns we used **Bar Plot.**





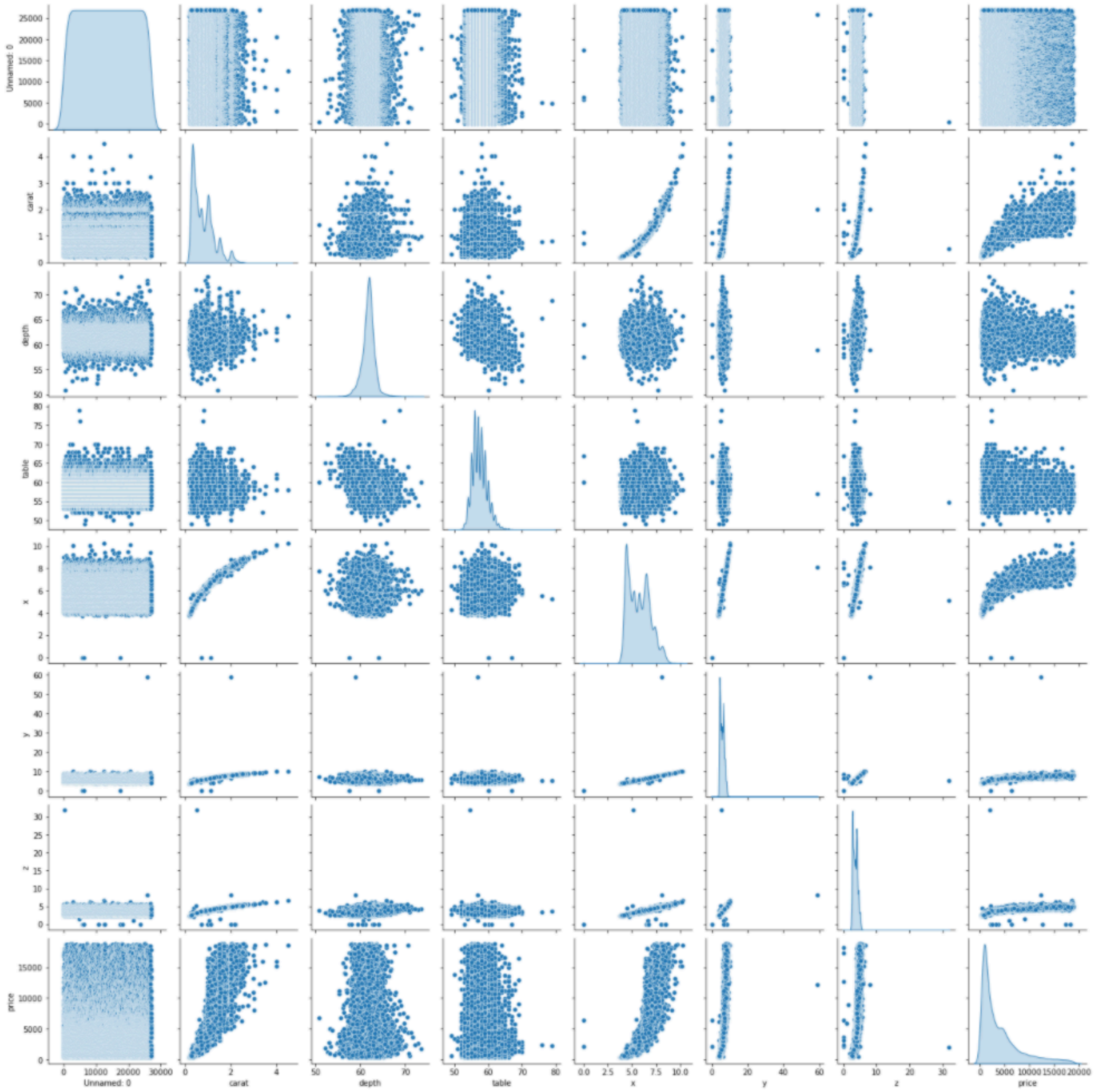




Please refer the python notebook for better clarity)

While evaluating bi-variate analysis we included price column to analyse Categoric columns like **" Cut" ,"Clarity "**& **"Color"**

Along with Heat-map to extend our analysis we wanted to show the entire distribution of the dataset. Hence we tried implementing **pair plot**



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?

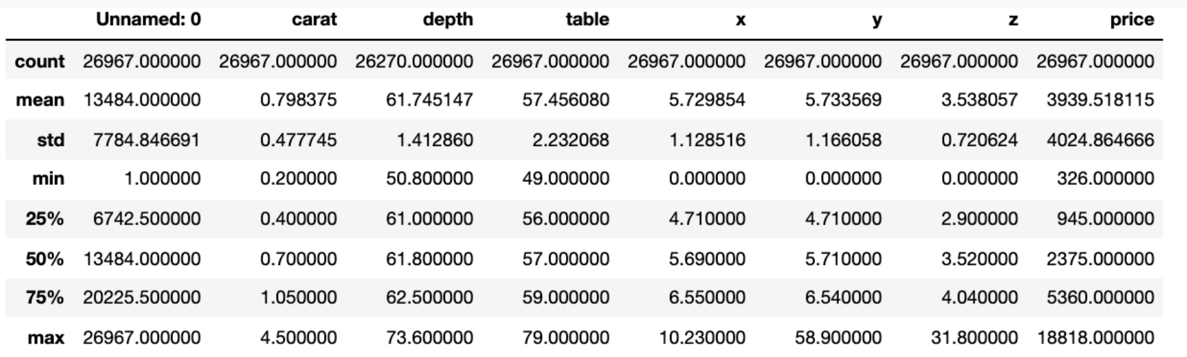
Checking for Null values existence :

We checked the duplicate values by a method called **isnull()** and it returned **True** , that indicates this dataset has null values .

Now we wanted to trace further to know where all null values are existing in the dataset and it returned that column **"depth"** is having it.
(Please refer notebook for better clarity)

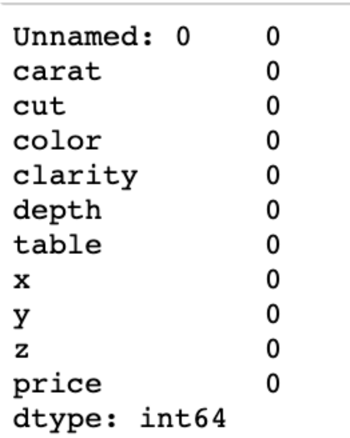
We can impute them by replacing with median as there is outliers existence in this dataset.

we have also seen that X , Y & Z columns minimum value is projected as 0.0 which is wrong as its related to diamond shape & size, it can't be practically 0. Hence we wanted to treat the same by dropping them.



	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

So together we had to remove "0" from x,y,z & depth column. We clean the same and recheck our dataset whether its imputed correctly or not & dataset returned as below :
(Please refer notebook for better clarity)

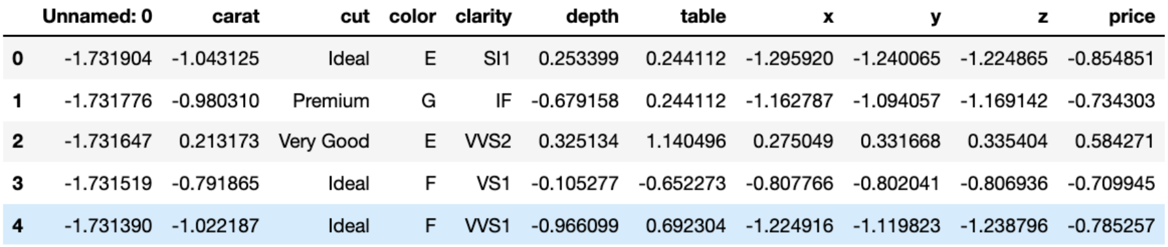


```
Unnamed: 0    0
carat         0
cut           0
color         0
clarity       0
depth         0
table         0
x             0
y             0
z             0
price         0
dtype: int64
```

Above screenshot depicts that there is no missing value after treating the impurities .

In terms of Scaling , we found that **Yes** , **Scaling is necessary for this dataset as the magnitude of the columns are not in same range** . By scaling we can bring them in the same magnitude and it will give us a better result . **(Please refer notebook for better clarity)**

We have used standard scaler to scale the entire dataset as below :



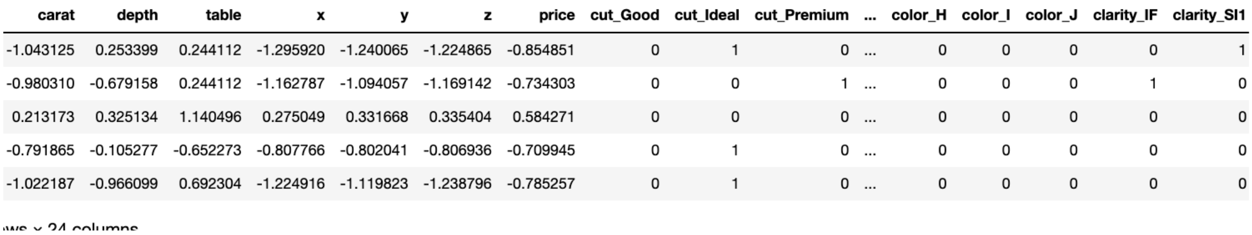
	Unnamed: 0	carat	cut	color	clarity	depth	table	x	y	z	price
0	-1.731904	-1.043125	Ideal	E	SI1	0.253399	0.244112	-1.295920	-1.240065	-1.224865	-0.854851
1	-1.731776	-0.980310	Premium	G	IF	-0.679158	0.244112	-1.162787	-1.094057	-1.169142	-0.734303
2	-1.731647	0.213173	Very Good	E	VVS2	0.325134	1.140496	0.275049	0.331668	0.335404	0.584271
3	-1.731519	-0.791865	Ideal	F	VS1	-0.105277	-0.652273	-0.807766	-0.802041	-0.806936	-0.709945
4	-1.731390	-1.022187	Ideal	F	VVS1	-0.966099	0.692304	-1.224916	-1.119823	-1.238796	-0.785257

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

(Please refer notebook for better clarity)

Converting categorical to dummy variables (One Hot Encoding)

Defining **"X"** & **"Y"** by dropping & popping target variable . In this case it is **"price".**



carat	depth	table	x	y	z	price	cut_Good	cut_Ideal	cut_Premium	...	color_H	color_I	color_J	clarity_IF	clarity_SI1
-1.043125	0.253399	0.244112	-1.295920	-1.240065	-1.224865	-0.854851	0	1	0	...	0	0	0	0	1
-0.980310	-0.679158	0.244112	-1.162787	-1.094057	-1.169142	-0.734303	0	0	1	...	0	0	0	1	0
0.213173	0.325134	1.140496	0.275049	0.331668	0.335404	0.584271	0	0	0	...	0	0	0	0	0
-0.791865	-0.105277	-0.652273	-0.807766	-0.802041	-0.806936	-0.709945	0	1	0	...	0	0	0	0	0
-1.022187	-0.966099	0.692304	-1.224916	-1.119823	-1.238796	-0.785257	0	1	0	...	0	0	0	0	0

ws × 24 columns

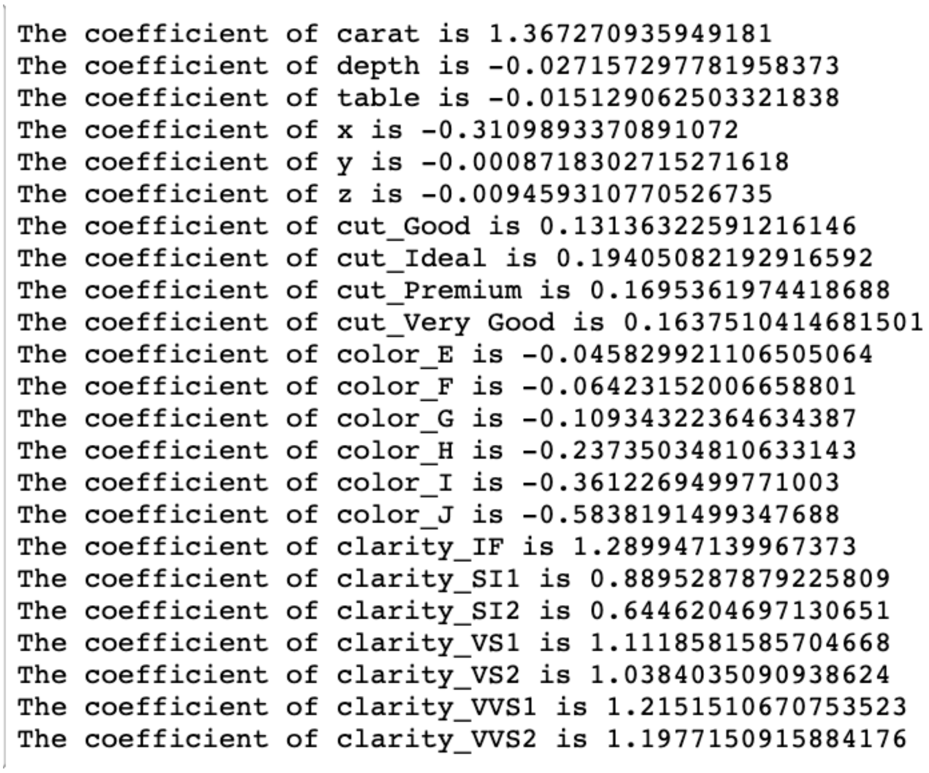
(Please refer notebook for better clarity)

1. Split the Train & Test data into 70:30 ratio

2. Applied Linear Regression Model

(Please refer notebook for better clarity)

After fitting the model we started deriving the coefficients of each predictor variables .



```
The coefficient of carat is 1.367270935949181
The coefficient of depth is -0.027157297781958373
The coefficient of table is -0.015129062503321838
The coefficient of x is -0.3109893370891072
The coefficient of y is -0.0008718302715271618
The coefficient of z is -0.009459310770526735
The coefficient of cut_Good is 0.13136322591216146
The coefficient of cut_Ideal is 0.19405082192916592
The coefficient of cut_Premium is 0.1695361974418688
The coefficient of cut_Very Good is 0.1637510414681501
The coefficient of color_E is -0.045829921106505064
The coefficient of color_F is -0.06423152006658801
The coefficient of color_G is -0.10934322364634387
The coefficient of color_H is -0.23735034810633143
The coefficient of color_I is -0.3612269499771003
The coefficient of color_J is -0.5838191499347688
The coefficient of clarity_IF is 1.289947139967373
The coefficient of clarity_SI1 is 0.8895287879225809
The coefficient of clarity_SI2 is 0.6446204697130651
The coefficient of clarity_VS1 is 1.1118581585704668
The coefficient of clarity_VS2 is 1.0384035090938624
The coefficient of clarity_VVS1 is 1.2151510670753523
The coefficient of clarity_VVS2 is 1.1977150915884176
```

Intercept for the model returned **-0.9907889549988897**

Deriving R-square value of Train & Test data

R square on training data : **0.9232445774547247**

R square on testing data : **0.9171155258688372**

Deriving RMSE value of Train & Test data

RMSE on Training data : **0.274480473337745**

RMSE on Testing data : **0.29399280117472737**

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

The given dataset is to predict the price of the stone and to provide insights for the company on the profits on different prize slots.

Initial analysis displayed that the **ideal** cut was most preferred and costly and in turn out to be profitable for the company. The **ideal, premium** and **very good** types of cut are generating profits whereas fair and good are not generating profits.

Inferences drawn from the above analysis:

1. Few significant cuts which are in general profitable to the company those should be sold & marketed better.

2. The clarity of the diamond is an important attribute. Hence the company should focus on maintaining the same quality for continuous profits.

3. Few other attributes to be looked after are: **carat, clarity_VS1, clarity_VS2, clarity_VVS1,clarity_VVS2.**

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.¶

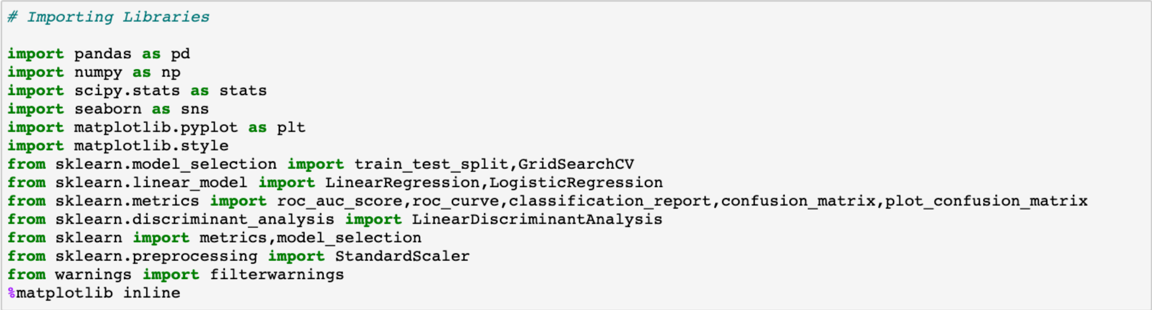
Variable Name	**Description**
Holiday_Package	Opted for Holiday Package yes/no?
Salary	Employee salary
age	Age in years
edu	Years of formal education
no_young_children	The number of young children (younger than 7 years)
no_older_children	Number of older children
foreign	foreigner Yes/No

Questions:

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.

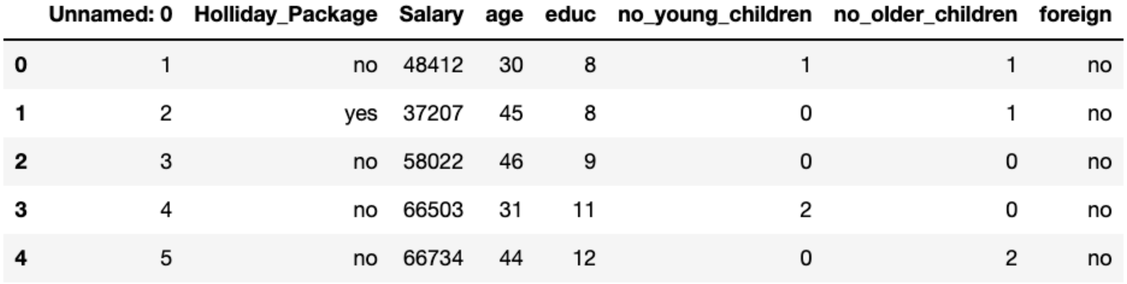
While reading the data and understanding to do exploratory data analysis we followed below steps likewise :

Importing necessary libraries :



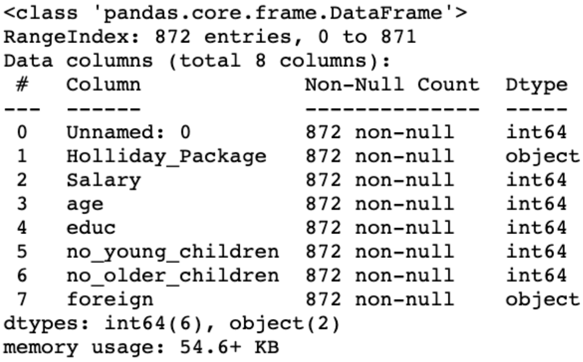
```
# Importing Libraries

import pandas as pd
import numpy as np
import scipy.stats as stats
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.style
from sklearn.model_selection import train_test_split,GridSearchCV
from sklearn.linear_model import LinearRegression,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 import metrics,model_selection
from sklearn.preprocessing import StandardScaler
from warnings import filterwarnings
%matplotlib inline
```

Read the data :



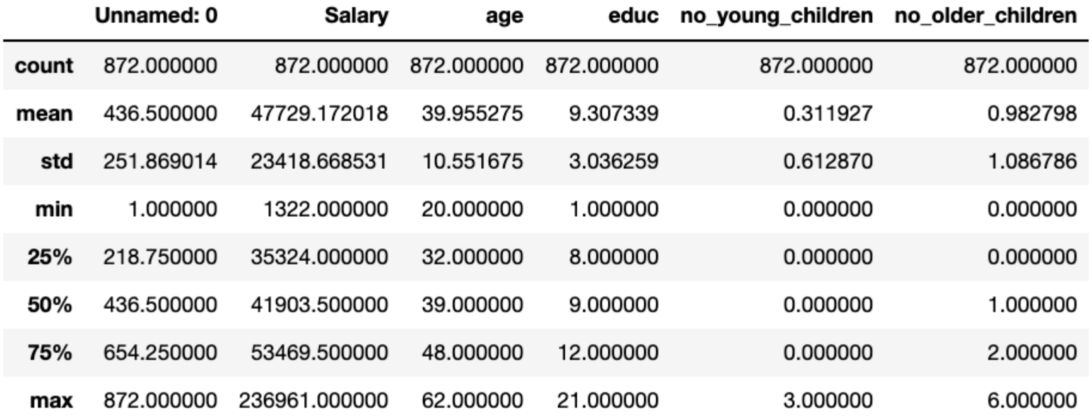
	Unnamed: 0	Holliday_Package	Salary	age	educ	no_young_children	no_older_children	foreign
0	1	no	48412	30	8	1	1	no
1	2	yes	37207	45	8	0	1	no
2	3	no	58022	46	9	0	0	no
3	4	no	66503	31	11	2	0	no
4	5	no	66734	44	12	0	2	no

Information of the data (data-types & row-column) :



```
<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
```

Descriptive statistics to summarize the data :



	Unnamed: 0	Salary	age	educ	no_young_children	no_older_children
count	872.000000	872.000000	872.000000	872.000000	872.000000	872.000000
mean	436.500000	47729.172018	39.955275	9.307339	0.311927	0.982798
std	251.869014	23418.668531	10.551675	3.036259	0.612870	1.086786
min	1.000000	1322.000000	20.000000	1.000000	0.000000	0.000000
25%	218.750000	35324.000000	32.000000	8.000000	0.000000	0.000000
50%	436.500000	41903.500000	39.000000	9.000000	0.000000	1.000000
75%	654.250000	53469.500000	48.000000	12.000000	0.000000	2.000000
max	872.000000	236961.000000	62.000000	21.000000	3.000000	6.000000

Shape of the data (No. of Row's & Column's)

We got to know there are 872 rows & 8 columns available

Checking for null values : (if it is there)

We checked the null values by a method called **isnull()** and it returned **false** , that indicates this dataset does not have any null values .

Checking for duplicate values existence :

We checked the duplicate values by a method called **duplicated()** and it returned **False** , that indicates this dataset has no duplicate values .

Checking Unique values present in categorical Columns :

```
HOLLIDAY_PACKAGE :  2
yes    401
no     471
Name: Holliday_Package, dtype: int64


FOREIGN :  2
yes    216
no     656
Name: foreign, dtype: int64
```

Above analysis depicts that **HOLIDAY_PACKAGE** & **FOREIGN** columns have two category of each **"yes"** & **"NO"** with above mentioned proportion.

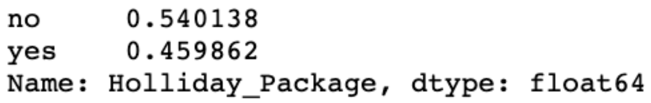
Columns of the dataset :

Listed columns are :

```
'Unnamed: 0', 'Holliday_Package', 'Salary', 'age',
'educ','no_young_children', 'no_older_children', 'foreign'
```

Checking the proportion of "yes" & "no" in the target column :

We wanted to check whether this dataset is a balanced or not & we were returned with



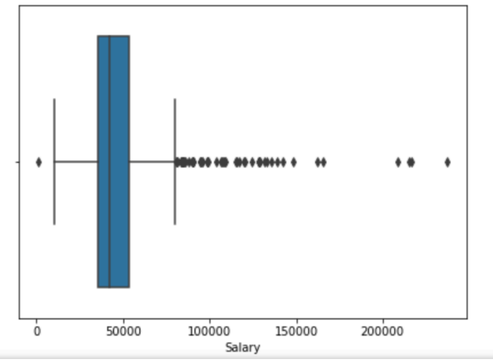
```
no     0.540138
yes    0.459862
Name: Holliday_Package, dtype: float64
```

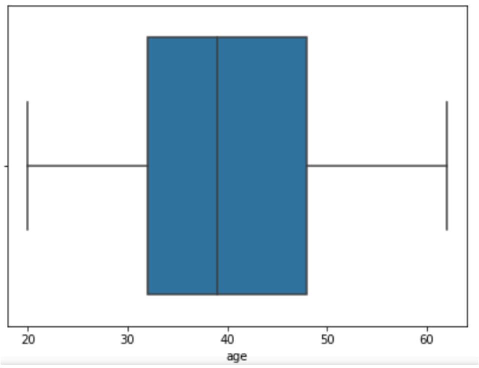
Extracting only continuous variables from the dataset :

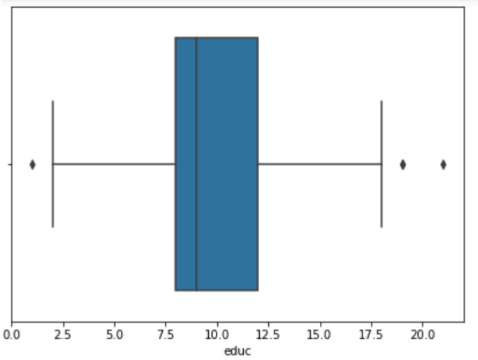
There were total **6** columns got extracted.

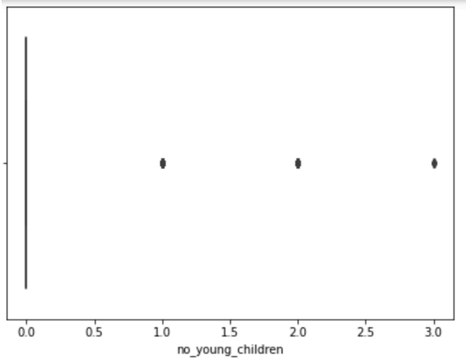
Checking for outliers :

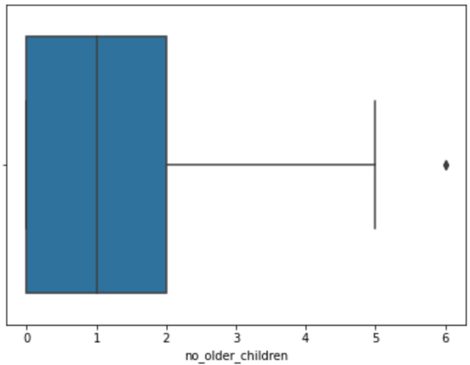
To know whether this dataset consists outliers or not we verified it through Box-plot.











Here from the above visualisation we get to know there are outlier's existence in this dataset but looking at the practicality we decided not to treat them as of now .

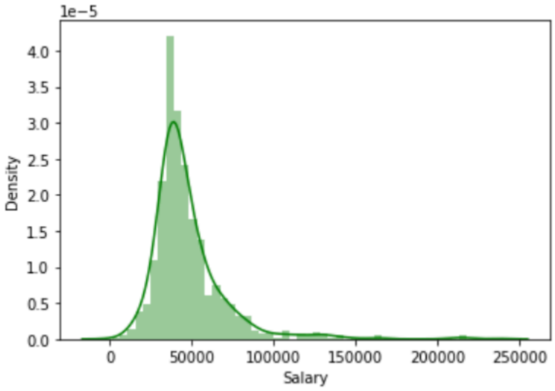
Univariate Analysis :

Univariate analysis is the simplest form of analysing data. “Uni” means “one”, so in other words our data takes only one variable at a time. It doesn't deal with causes or relationships (unlike regression) and it's major purpose is to describe; It takes data (individual column) separately , summarises that data and finds patterns in the data.

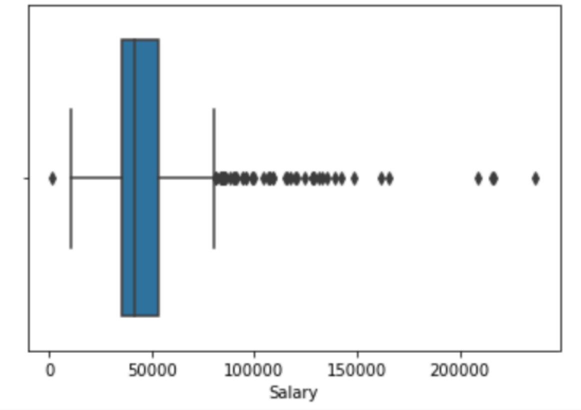
Here we have segregated categorical & numeric columns and analysed them separately.

(Please refer notebook for better clarity)

```
Description of Salary
------------------------------------------------------------
----------------
count         872.000000
mean        47729.172018
std         23418.668531
min          1322.000000
25%         35324.000000
50%         41903.500000
75%         53469.500000
max        236961.000000
Name: Salary, dtype: float64
------------------------------------------------------------
----------------
Distribution of Salary
------------------------------------------------------------
----------------
```



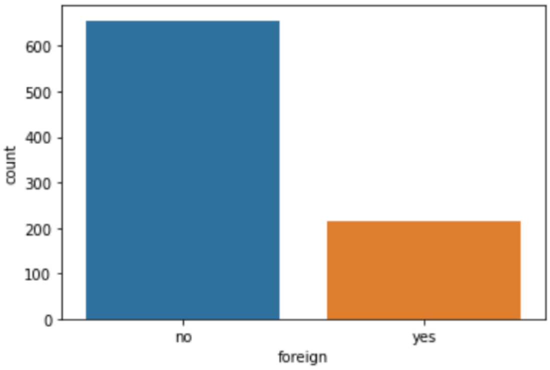
```
BoxPlot of Salary
-------------------------------------------------------------
----------------
```

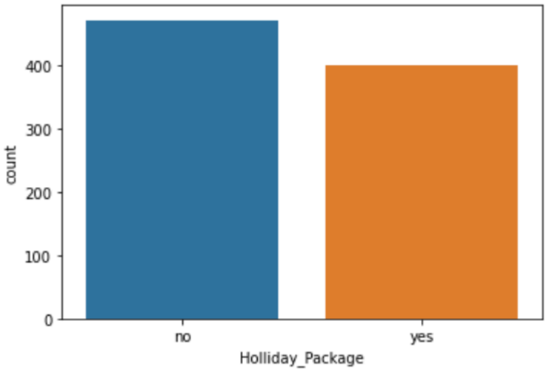


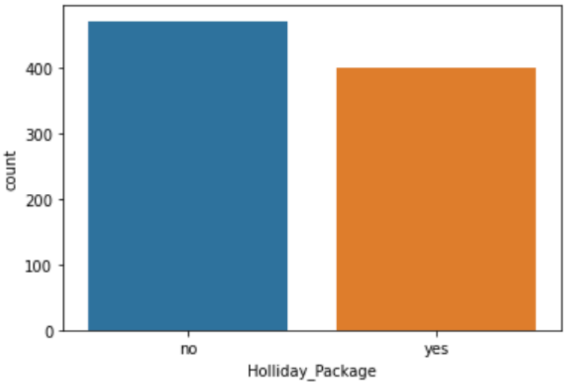
The output displays, total 6*3 = 18 distinct charts/columns & descriptions including column “Unnamed:0”. Hence I have put the screenshot of only one variable which is **Salary** .

(Please refer the python notebook for better clarity)

We wanted the same analysis **(Univariate analysis)** for **categorical** variables . We tried depicting them through **count plot .**





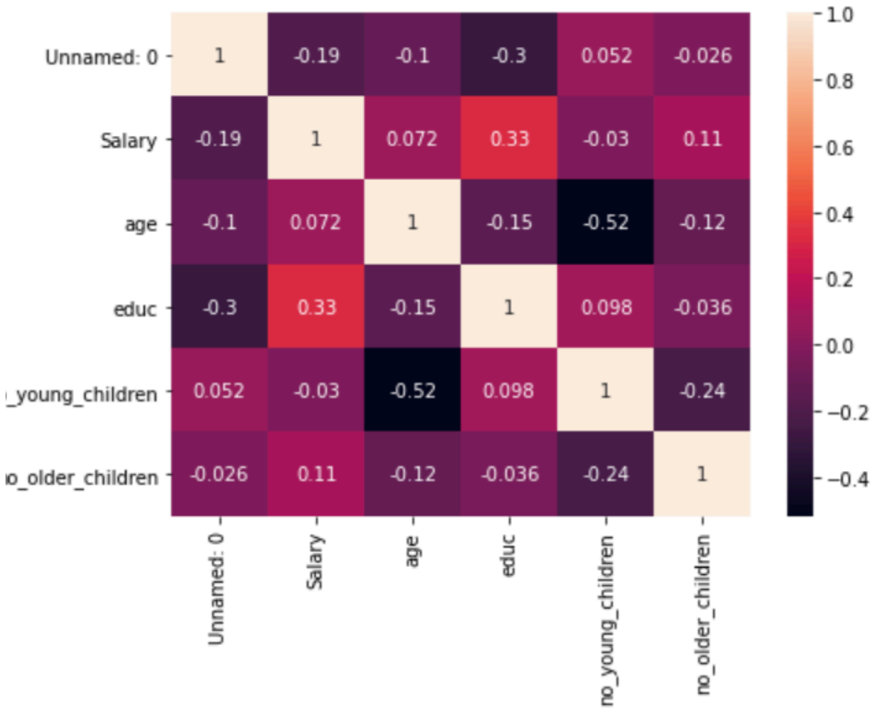


Bi-variate Analysis

As the name suggest it gives analysis of two specific variable at a time and we can derive correlation b/w them. We can use Heat-map for the same and further to highlight each correlated pair we can use scatter-plot .

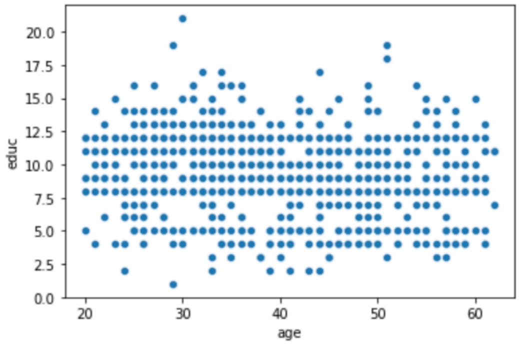
Like Univariate analysis here also we have segregated numeric & categorical fields and analysed them separately.

Here we have done analysis through Heat-map only on numeric fields .



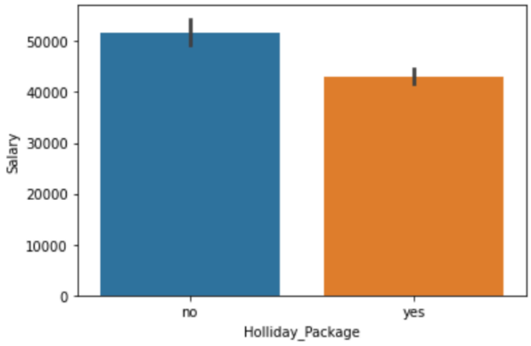
From the above heat-map we can derive that there is hardly any correlation observed among the continuous predictors here. Out of these we can see that **"Salary"** & **"educ"** have comparatively better correlation , which is **33 %.**

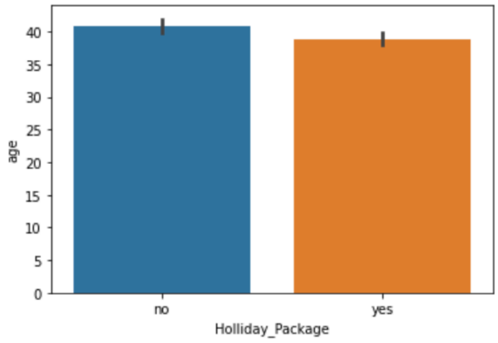
By using scatter plot we can depicts the correlation in a better way .

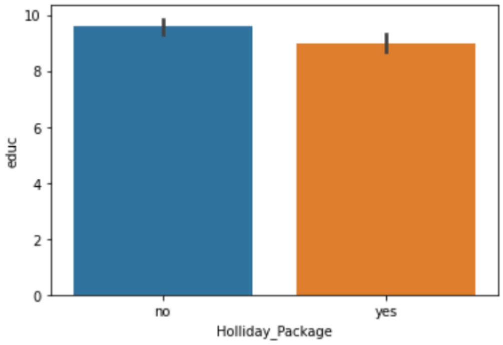


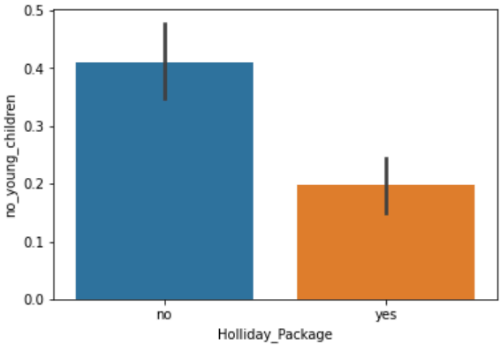
Along with the numerical fields we wanted to do the same **(Bi-variate Analysis)** for all categorical variable through **Bar Plot & Scatter plot.**

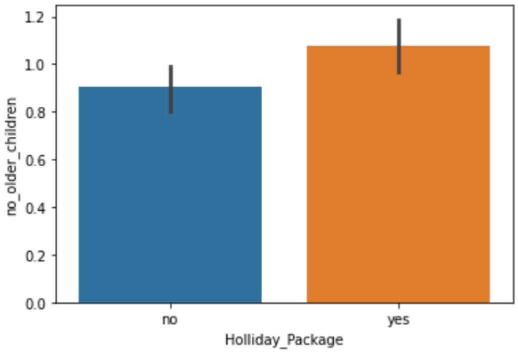
Here we have included target variable to understand existing different hidden pattern in the dataset and tried plotting them in multiple combination , as below :

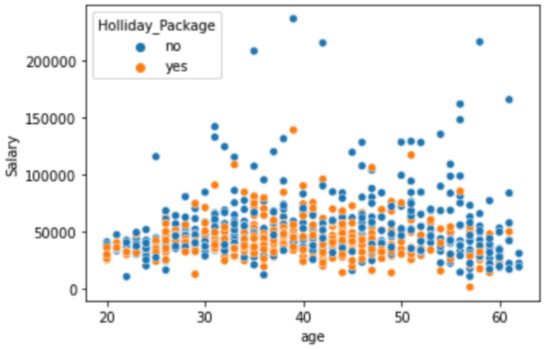


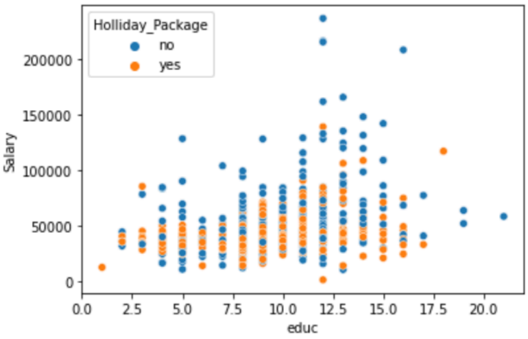


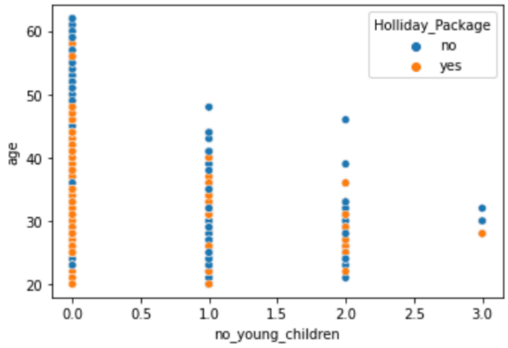


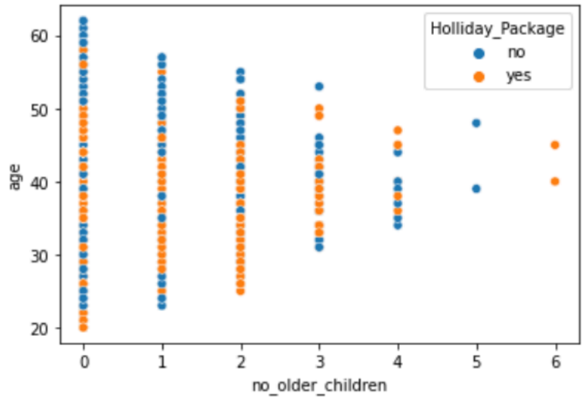




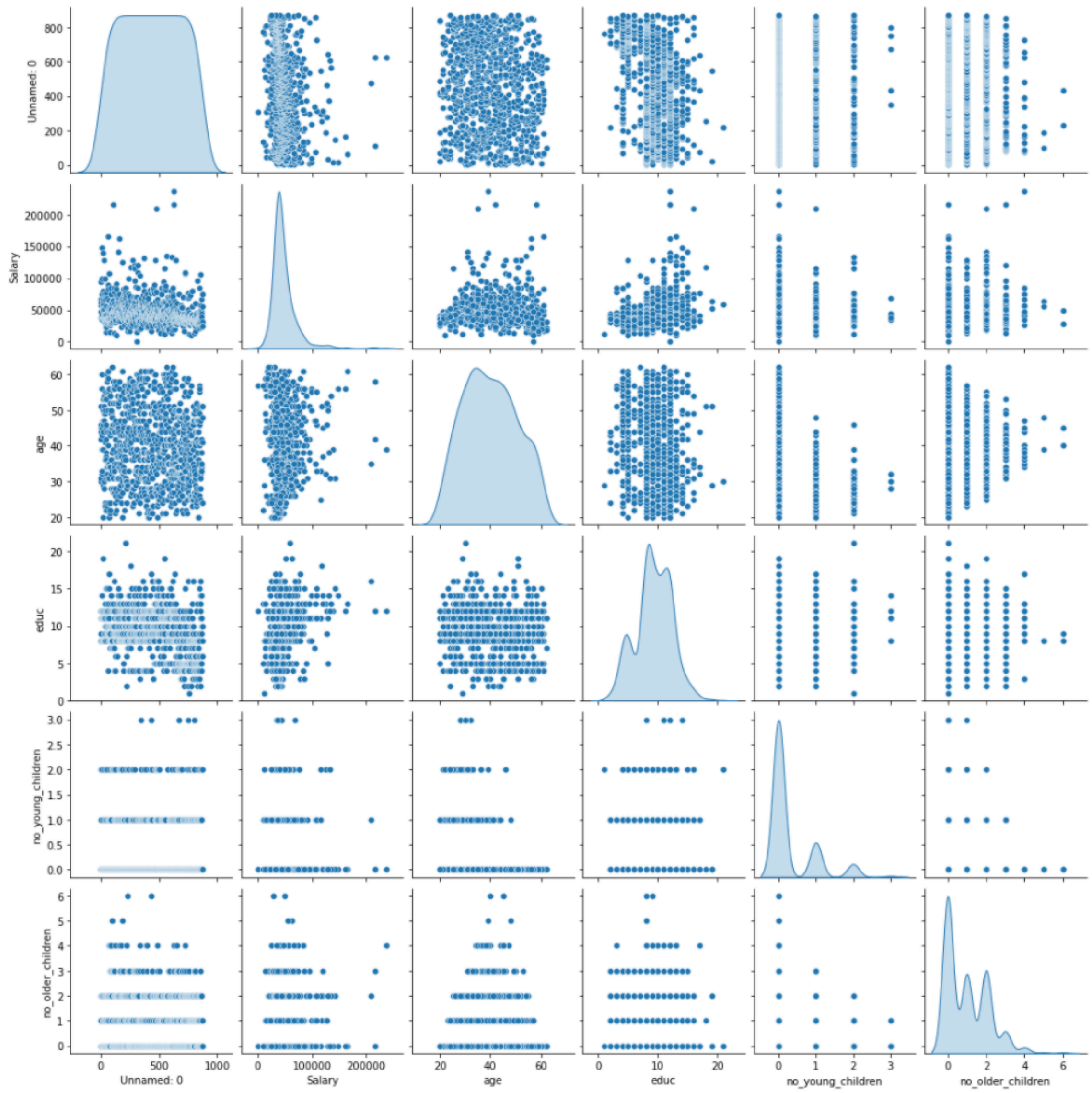








Wanted to see entire Data distribution through **Pair plot**



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)**

(Please refer notebook for better clarity)

Before doing **Train & Test split** , wanted to check the proportion of Claimed & Not Claimed

```
no      0.540138
yes     0.459862
```

From this we can get to know almost ~**46%** of people has taken Holiday Package & remaining **54%** people have not taken the same.

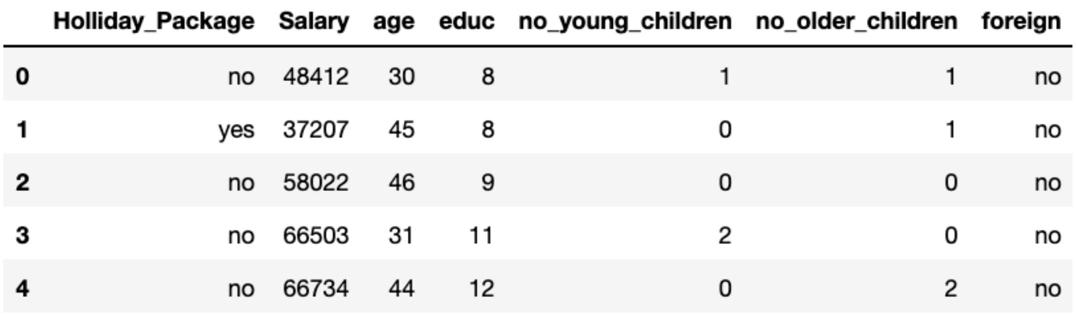
(Please refer notebook for better clarity)

We tried keeping here 30% data in the test set and splitting data into training and test set .

Logistic Regression :

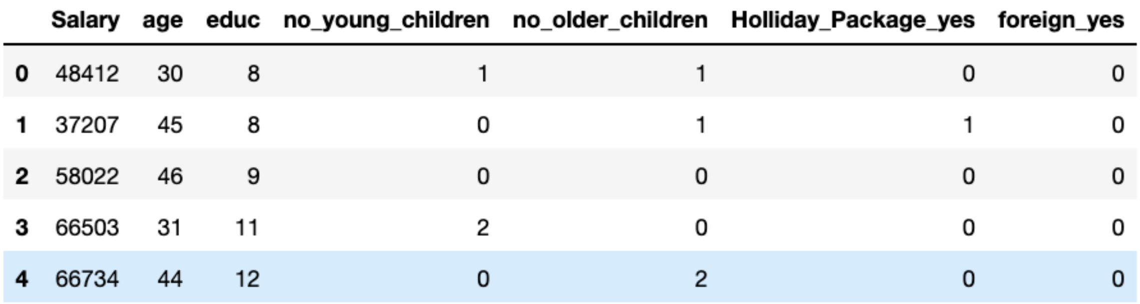
Dropping Unnamed:0

Before going ahead we are dropping the Unnamed:0 column as it is useless for the model



	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

Converting categorical to dummy variables in the dataset



	Salary	age	educ	no_young_children	no_older_children	Holliday_Package_yes	foreign_yes
0	48412	30	8	1	1	0	0
1	37207	45	8	0	1	1	0
2	58022	46	9	0	0	0	0
3	66503	31	11	2	0	0	0
4	66734	44	12	0	2	0	0

Finally , checking the predictor columns

```
'Salary', 'age', 'educ', 'no_young_children',
'no_older_children','Holliday_Package_yes', 'foreign_yes'
```

(Please refer notebook for better clarity)

Here after by capturing the target column **("Holliday_Package")** into separate vectors for training set and test set by **drop()** & **pop()** mechanism. This is how we derive X & Y by copying all independent variables into X & target into the Y

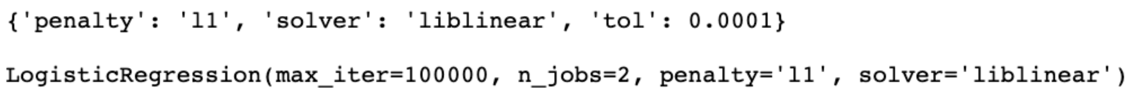
Implementing GridSearchCV upon Logistic Regression for best results

(Please refer notebook for better clarity)

Implemented Logistic Regression by **.fit()**method . Used parameter as below:

```
params={'penalty':['l1','l2','none'],

       'solver':['lbfgs', 'liblinear'],

       ‘tol':[0.0001,0.000001]}
```

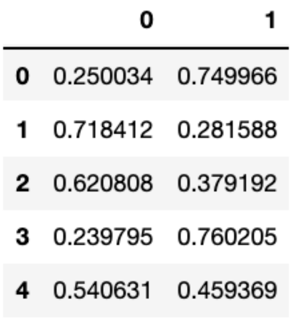
After fitting the model we wanted to derive **best params** & **best estimator**



```
{'penalty': 'l1', 'solver': 'liblinear', 'tol': 0.0001}

LogisticRegression(max_iter=100000, n_jobs=2, penalty='l1', solver='liblinear')
```

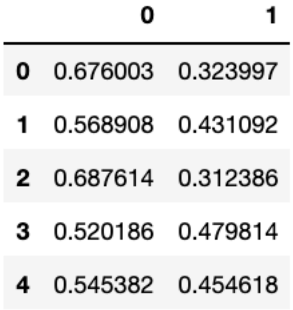
Assigning a model named as “best_model” using **best_estimator_** and derive the probabilities on train & test set.
(Please refer notebook for better clarity)

Probabilities on train set :



	0	1
0	0.250034	0.749966
1	0.718412	0.281588
2	0.620808	0.379192
3	0.239795	0.760205
4	0.540631	0.459369

Probabilities on test set :



	0	1
0	0.676003	0.323997
1	0.568908	0.431092
2	0.687614	0.312386
3	0.520186	0.479814
4	0.545382	0.454618

Post this we generated the classification report of train & test data along with confusion matrix from Logistic Regression.

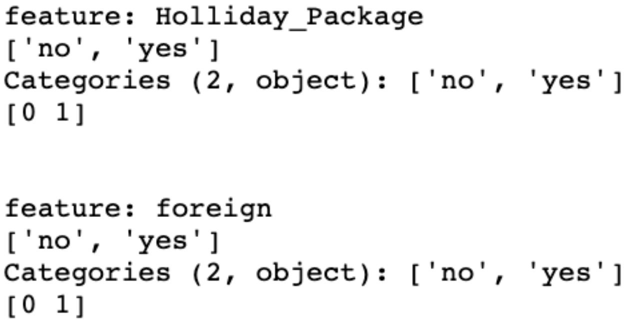
(Please refer notebook for better clarity)

<u>Linear Discriminant analysis</u>

While implementing LDA we took the original dataset **df_holiday** along with following steps

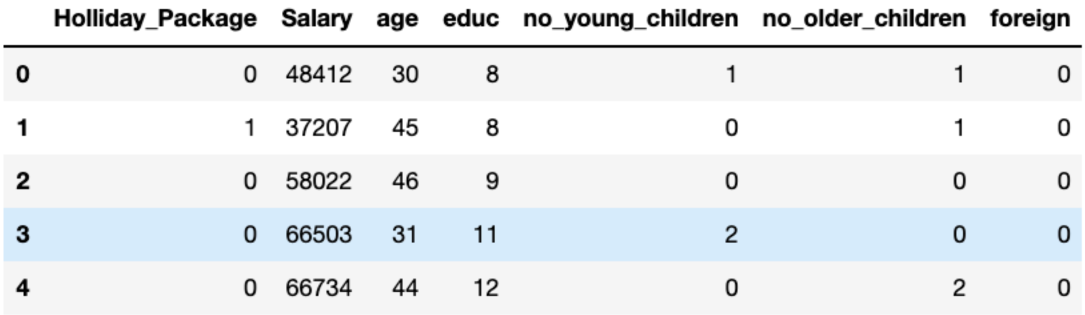
(Please refer notebook for better clarity)

Label encoding for categorical variables



```
feature: Holliday_Package
['no', 'yes']
Categories (2, object): ['no', 'yes']
[0 1]


feature: foreign
['no', 'yes']
Categories (2, object): ['no', 'yes']
[0 1]
```

Checking head of the data after conversion



	Holliday_Package	Salary	age	educ	no_young_children	no_older_children	foreign
0	0	48412	30	8	1	1	0
1	1	37207	45	8	0	1	0
2	0	58022	46	9	0	0	0
3	0	66503	31	11	2	0	0
4	0	66734	44	12	0	2	0

Here after by capturing the target column **("Holliday_Package")** into separate vectors for training set and test set by **drop()** & **pop()** mechanism. This is how we derive X & Y by copying all independent variables into X & target into the Y

Here we tried using default values with **LDA** instead **GridSearchCV**

(Please refer notebook for better clarity)

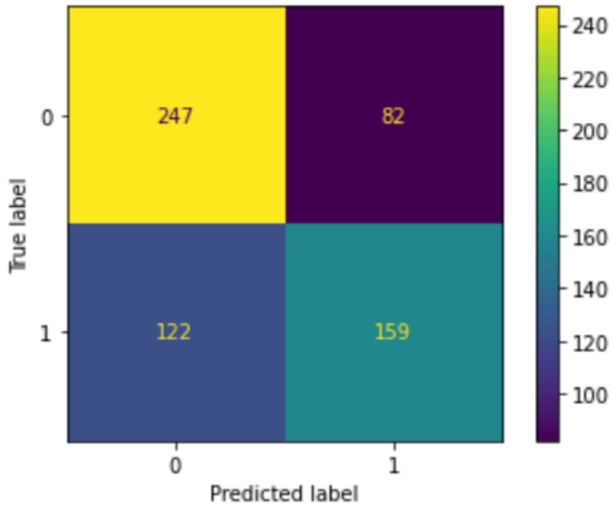
Post this we generated the classification report of train & test data along with confusion matrix from **LDA (Linear Discriminant Analysis)**

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.

(Please refer notebook for better clarity)

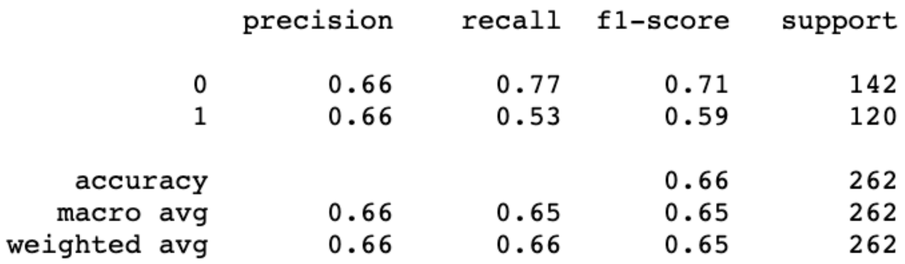
Printing Confusion matrix & Classification Report on train data (LR) :

```
              precision    recall  f1-score   support

           0       0.67      0.75      0.71       329
           1       0.66      0.57      0.61       281

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

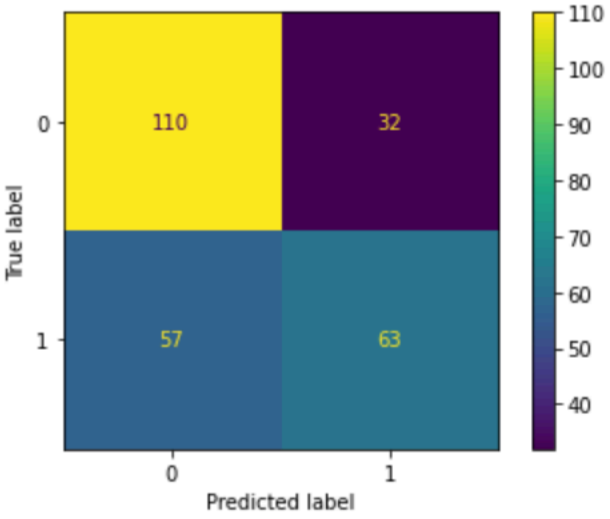


(Please refer notebook for better clarity)

Printing Confusion matrix & Classification Report on test data (LR) :



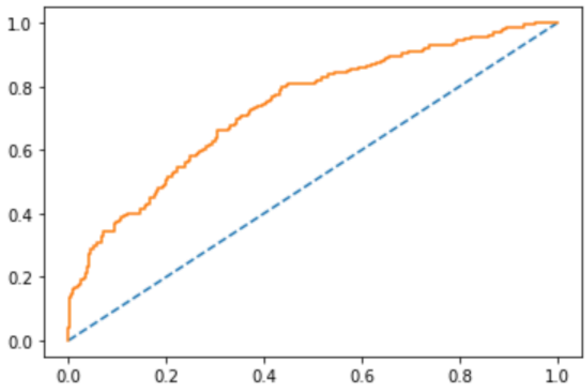
```
              precision    recall  f1-score   support

           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
```



(Please refer notebook for better clarity)

AUC and ROC of the train dataset (LR) :

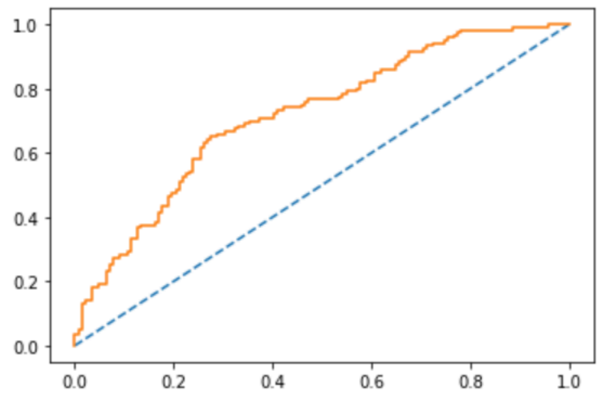
```
AUC: 0.735
```



(Please refer notebook for better clarity)

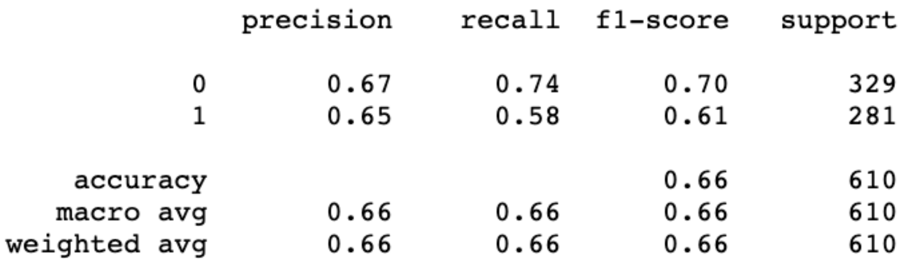
AUC and ROC of the test dataset (LR) :

```
AUC: 0.718
```

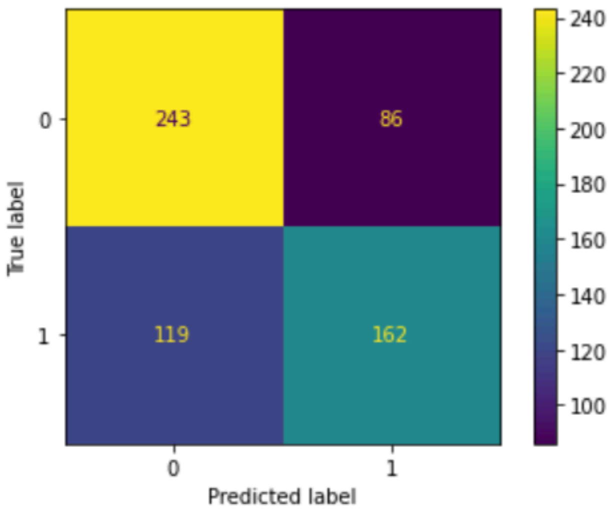


Printing Confusion matrix & Classification Report on train data(LDA) :

(Please refer notebook for better clarity)

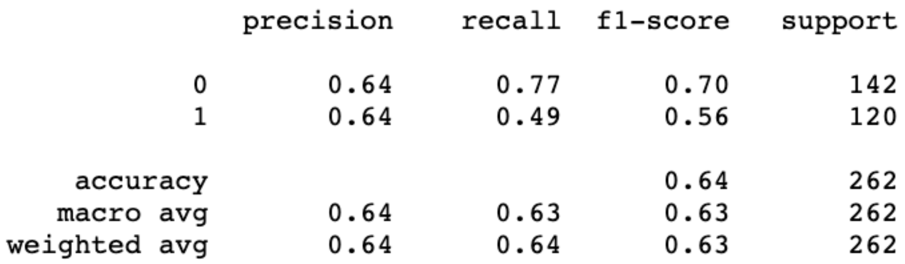


```
              precision    recall  f1-score   support

           0       0.67      0.74      0.70       329
           1       0.65      0.58      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
```

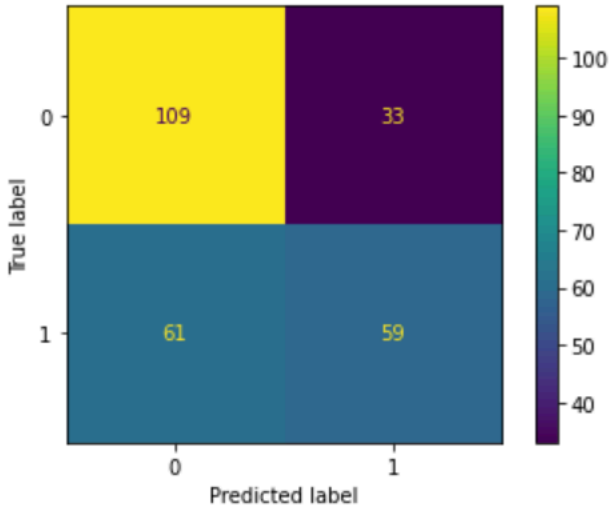


Printing Confusion matrix & Classification Report on test data (LDA) :

(Please refer notebook for better clarity)



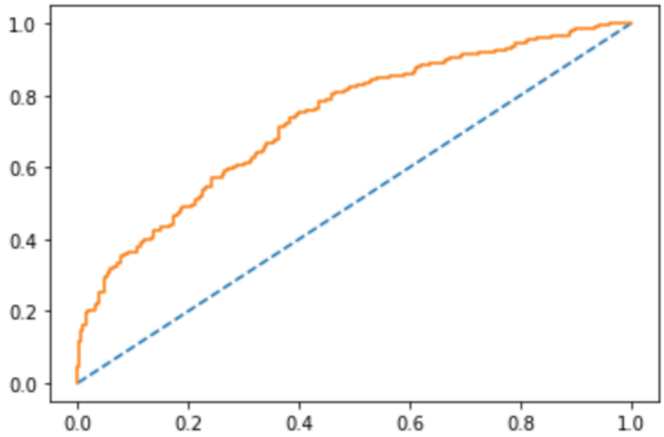
```
              precision    recall  f1-score   support

           0       0.64      0.77      0.70       142
           1       0.64      0.49      0.56       120

    accuracy                           0.64       262
   macro avg       0.64      0.63      0.63       262
weighted avg       0.64      0.64      0.63       262
```



AUC and ROC of the train dataset (LDA) :

(Please refer notebook for better clarity)

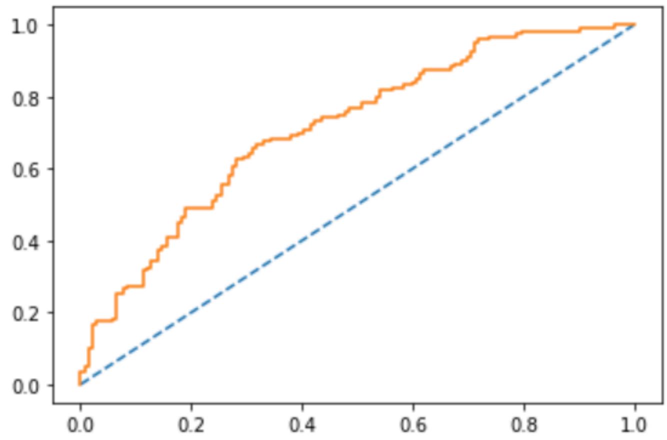
```
AUC: 0.733
```



AUC and ROC of the test dataset (LDA) :

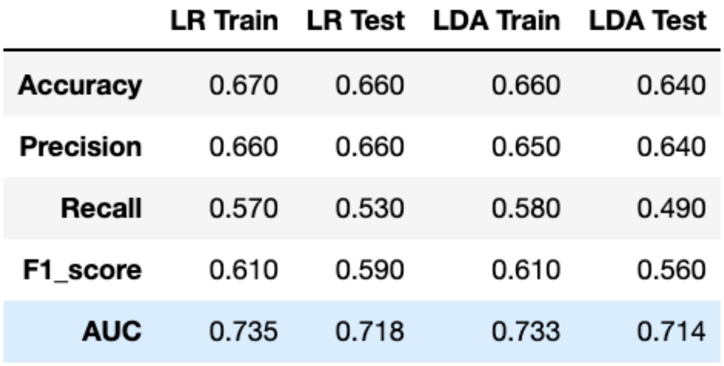
(Please refer notebook for better clarity)

```
AUC: 0.714
```



Comparison table based on above 2 (LR , LDA) reports

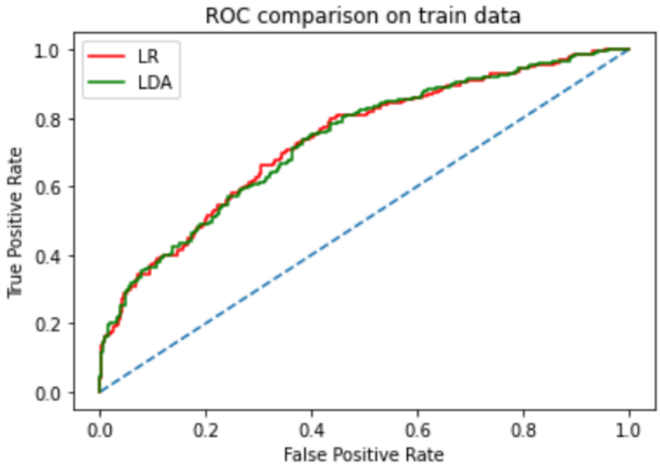
(Please refer notebook for better clarity)



	LR Train	LR Test	LDA Train	LDA Test
Accuracy	0.670	0.660	0.660	0.640
Precision	0.660	0.660	0.650	0.640
Recall	0.570	0.530	0.580	0.490
F1_score	0.610	0.590	0.610	0.560
AUC	0.735	0.718	0.733	0.714

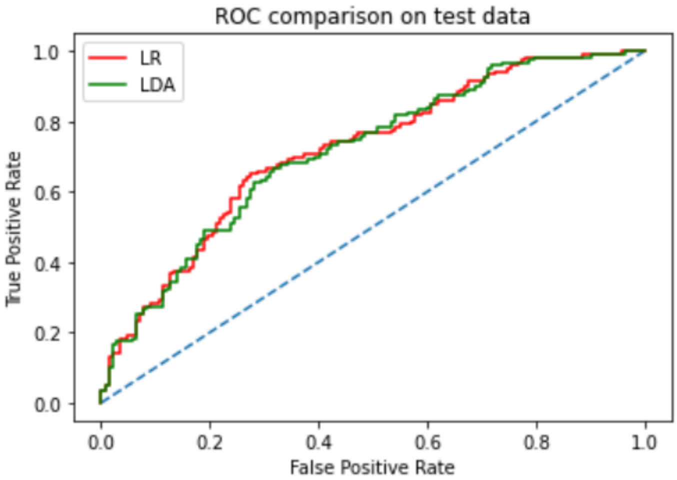
Comparison graph based on above 2 ROC Curve (LR , LDA) based on train data

(Please refer notebook for better clarity)



Comparison graph based on above 2 ROC Curve (LR , LDA) based on test data

(Please refer notebook for better clarity)



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

From the initial analysis we can see below patterns :

1. Employees who were aged beyond 50 or more did not opt much for holiday packages as that may not be feasible.

2. Employees with no children opted more for holiday packages as there is no dependency . Salary range also played an important role for opting for these holiday packages.

Business Inferences driven :

1. Packages has to be More affordable (budget-friendly) and the same sort of packages need to be introduced as more people can avail it.

2. The packages should have a flexibility option like where employees can customize based on their interests , age groups & salaries

3. As we have seen there is significant amount of reduction based on no. of older children so we suggest for few employees with more than 2/3 children there should be some amount of discount offers introduced so that they can be interested in the packages and think of availing them.

4. For senior employees, a group package can be designed by which all the needs should be fulfilled, so they might get more interest to avail them.

THE END