BUSINESS REPORT FOR PREDICTIVE MODELING PROJECT

Table of Contents:

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.

Question 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.

Exploratory Data Analysis:

	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

The Dataset consists of 11 variables(First column being unnecessary as its insignificant for the analysis of the problem).

Numerical Columns : carat, depth, table, x, y, z and price.

Non-Numerical Columns : cut, color and clarity.

Descriptive Statistics of the Dataset:

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

Firstly, the column "Unnamed : 0" is removed from the dataset before proceeding further as its insignificant for the analysis. The describe function(describe(include='all')) provides us with the information about how much the data is spread across along with the information on the mean, standard deviation, count along with specific details on non-numerical columns such as unique(number of unique categories), top(most used) and freq(count of number of times of the most used) etc.

We have many categorical variables with most frequency seen in :

cut as '**Ideal**'(10805), color as **'G'** (5653) and clarity as **'SI1'** (6565).

From the looks of the values in numeric columns based on the mean values and the median values we can say that the distribution is relatively normal(not exact but close to normal distribution as we have an instance where even the std of a column is greater than the mean(Price)) .

Skewness Check:

```
Skewness of the Dataset:
 carat    1.114789
depth    -0.026086
table     0.765805
x         0.392290
y         3.867764
z         2.580665
price     1.619116
dtype: float64
```

Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. From the above result we can see that only the variable "depth" is negatively skewed compared everything else(which are right skewed with max seen in "y").

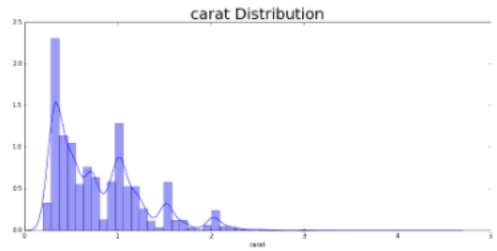
Duplicates Check:

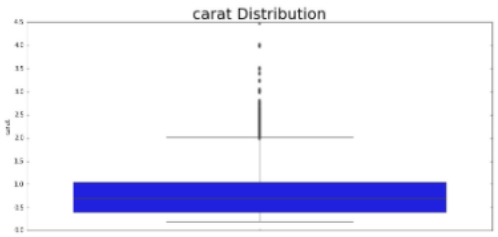
The purpose of this step is the look for any duplicates in the dataset as duplicates are redundant when it comes to analysing the dataset and its generally proposed to be removed. Upon looking into the dataset there was totally **34** duplicated lines present in the dataset which were dropped immediately using the drop_duplicates() function which retains only the unique lines.

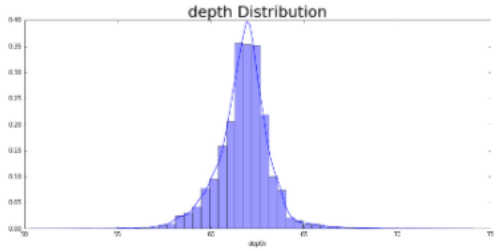
```
Number of duplicate rows = 34
```

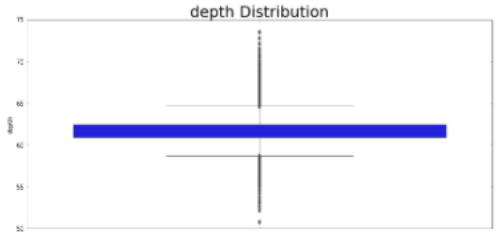
Note: These 34 rows are not exactly pure duplicates(two to three columns are always having different values), but for the purpose of this project, we are going to remove these records and continue further. But this decision should be taken after consulting with the Business side before concluding.

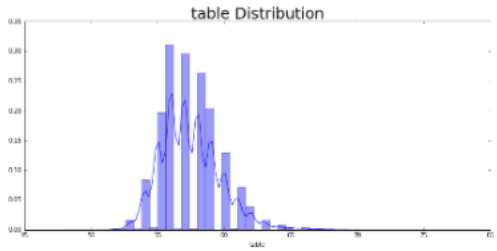
Univariate Data Visualization (Dist-plot and Boxplot analysis):

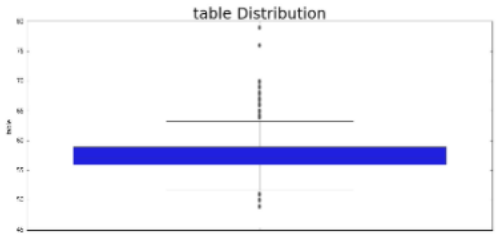


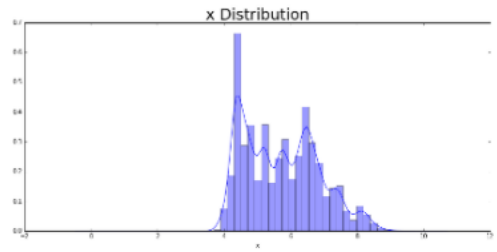


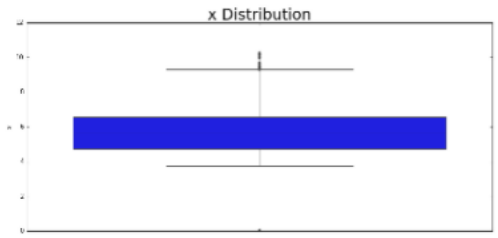


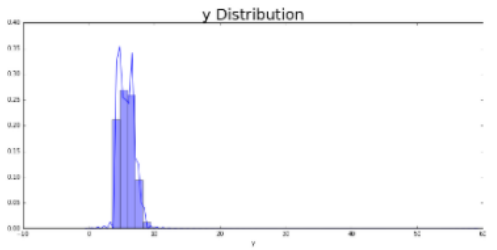


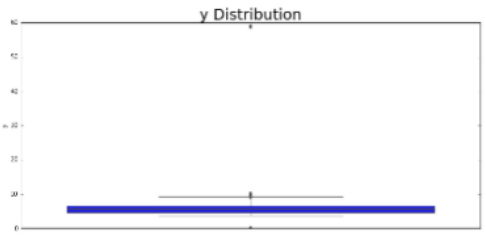


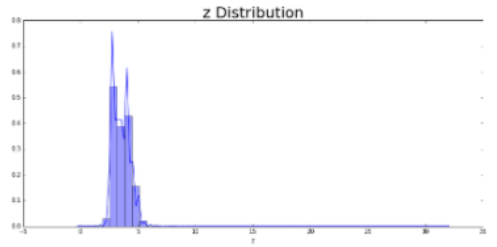


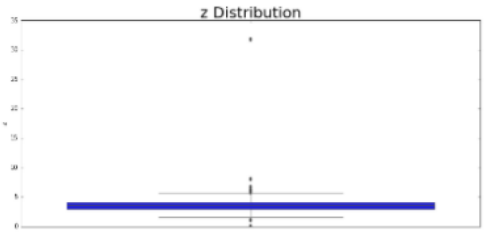








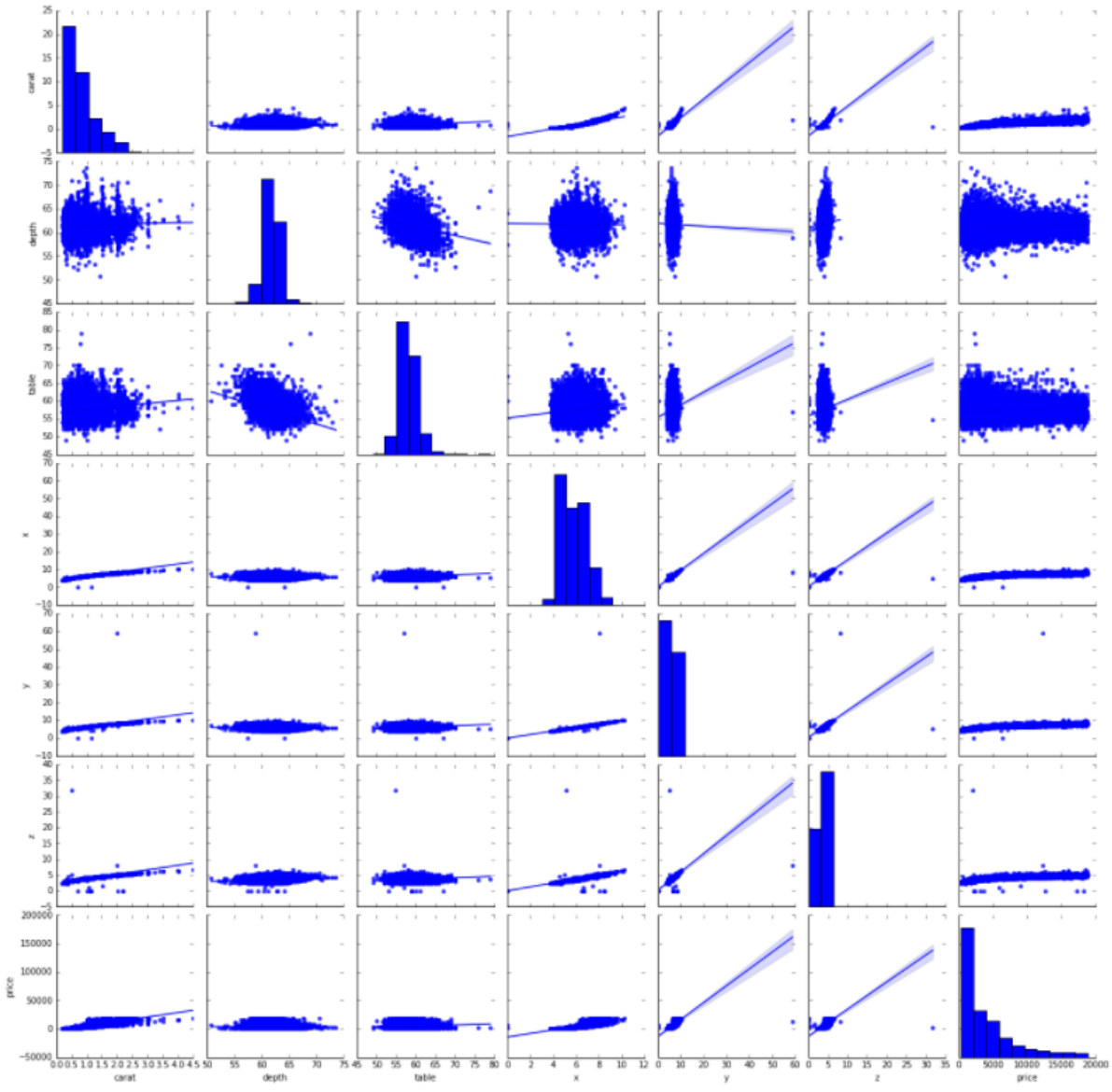




We can see that all the seven numerical Variables are normally distributed(not perfectly normal though, but still). There are a lot of outliers and is found in all the variables that can be seen from the boxplots on the right too.

For the purposes of this project we are going to move forward with outlier treatment for the dataset as it's highly recommended for better performance in linear regression. But its highly recommended to discuss with Business side to know for sure whether to remove certain outliers or not.

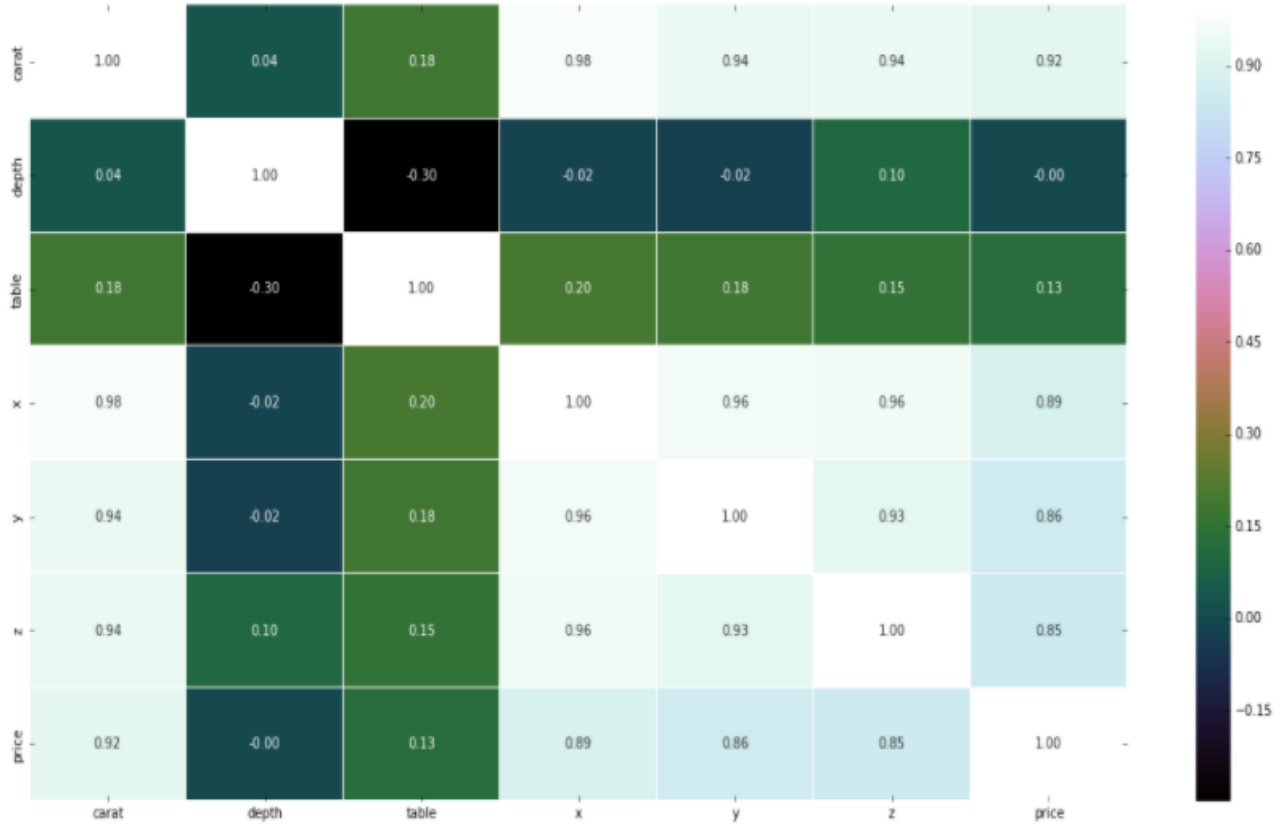
Bivariate Data Visualization (Pair-plot analysis):



A regular Pair-Plot analysis is first performed on the dataset to see their interaction with each of the other variables. From the above result we can say that there are not many strong relationships with the variables. The only notable relationship can be seen between the variable's "x" ,"y" and "z" with "price". Also, there are no significant negative relationship seen here("depth" has some with "table" and "y").

The pair-plot analysis can give us only a rough estimate that there is a linear relationship between the variables and its evident here that we do have some strong relationships.

Bivariate Data Visualization (Heat Map analysis):



Another most important plot that we must look in the case of analysis is the Correlation plot. The Heat-Map analysis is an visualized part of the Correlation matrix that is used to find the relationship between the variables with numerical values to standpoint on how much they are related as compared to other variables.

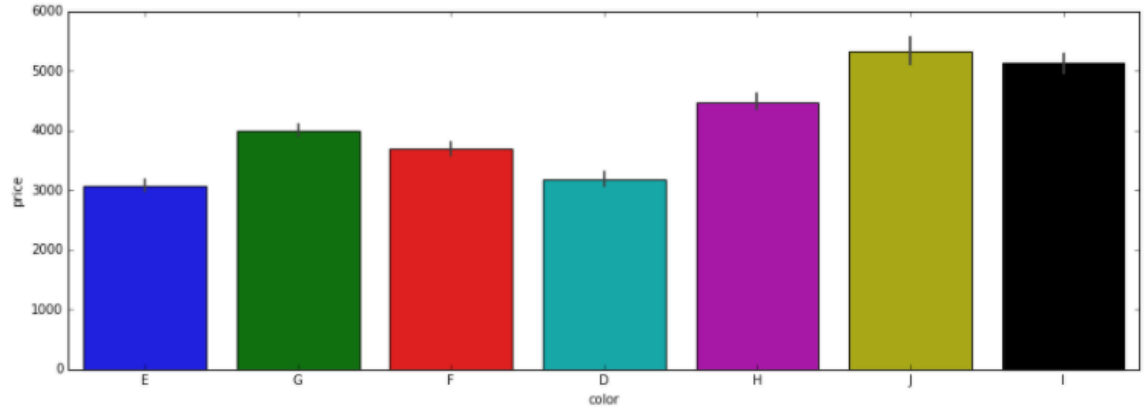
The following conclusions can be sought out of the above results:

- The highest correlation is seen between “carat” and “x” (98%)
- Notable high positive relationships are between “carat” and “y” , “z” and “price”
- Also, the target variable “Price” has some notable high correlations with “carat” , ”x” , ”y” and “z”(more than 85%)
- For negative correlations, the maximum was found between “depth” and “table”(30 %)

Note : Correlation doesn’t mean Causation. We need to analyse further to know more about the above results.

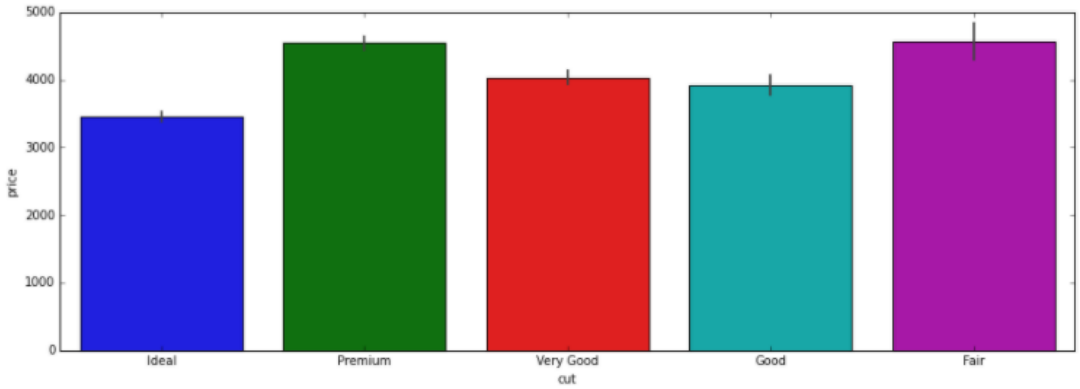
Bivariate Data Visualization (Bar Plot Analysis):

Price VS Color:



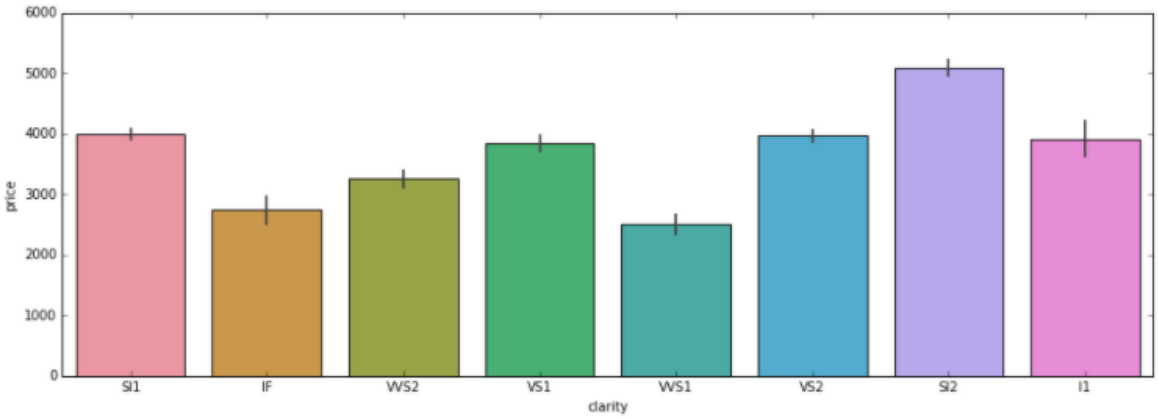
Notably, color "J" is the costliest and "E" is the cheapest.

Price VS Cut:



Notably, cut "Premium" is the costliest(accompanied by "Fair" with KDE variations more than Premium) and "Ideal" is the cheapest.

Price VS Clarity:



Notably, clarity "SI2" is the costliest and "WS1" is the cheapest.

Question 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?

Check for NULL Values:

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

The isnull() and sum() function combined can clearly figure out on whether given dataset has any NULL(N/A) values or not. From the above result it is evident that there are NULL values in column "depth" and therefore we need to remove these null values before we can proceed further in the Exploratory Data Analysis.

Checking for values which have "0" in them and noting their signifance:

The following rows have 0 values in them:

	carat	cut	color	clarity	depth	table	x	y	z	price
5821	0.71	Good	F	SI2	64.1	60.0	0.00	0.00	0.0	2130
6034	2.02	Premium	H	VS2	62.7	53.0	8.02	7.95	0.0	18207
10827	2.20	Premium	H	SI1	61.2	59.0	8.42	8.37	0.0	17265
12498	2.18	Premium	H	SI2	59.4	61.0	8.49	8.45	0.0	12631
12689	1.10	Premium	G	SI2	63.0	59.0	6.50	6.47	0.0	3696
17506	1.14	Fair	G	VS1	57.5	67.0	0.00	0.00	0.0	6381
18194	1.01	Premium	H	I1	58.1	59.0	6.66	6.60	0.0	3167
23758	1.12	Premium	G	I1	60.4	59.0	6.71	6.67	0.0	2383

We can gather from the above result that most of the "0" values originated from the "x", "y" and "z" columns. From the data dictionary provided along with the project the column "x" , "y" and "z" contains values for length, width and height respectively. Since they can't be zero for any diamond, it advisable to drop all these lines. The line "5821" and "17506" has all the three values as 0 and they are to be removed indefinitely, but for the rest only the z values have 0. We can opt to use median for this case, but my recommendation is to remove these lines as well for two reasons. One is that we have only 6 mines like these and its easy to remove them than using median values to provided with wrong accuracy and second reason is we can opt to deal this directly with the business team and get the proper results and proceed further.

Do you think scaling is necessary in this case?

Based on the Data Dictionary provided it is evident that we have values on different scales(Price in thousands and "x", "y" and "z" in single digits) and therefore scaling is definitely necessary in this case to bring out the best performance from the model. The Standard Scaler is performed on the dataset to bring the different scales to scale of mean = 0 and std = 1.

After Scaling:

	carat	cut	color	clarity	depth	table	x	y	z	price
0	-1.067382	0.979367	0.940777	-0.640136	0.286766	0.261968	-1.296530	-1.289659	-1.261558	-0.933395
1	-1.002446	0.080980	-0.231548	2.396449	-0.780365	0.261968	-1.163253	-1.137530	-1.204060	-0.793477
2	0.231349	-0.817407	0.940777	1.181815	0.368853	1.189326	0.276134	0.347964	0.348406	0.736960
3	-0.807636	0.979367	0.354615	0.574498	-0.123669	-0.665390	-0.807849	-0.833272	-0.830318	-0.765205
4	-1.045737	0.979367	0.354615	1.789132	-1.108713	0.725647	-1.225449	-1.164377	-1.275933	-0.852618
5	0.491095	0.979367	1.526940	-0.032819	-0.205756	-0.665390	0.649309	0.679069	0.650274	1.663877
6	0.469450	-1.715794	-0.817710	-0.640136	1.600159	1.189326	0.551573	0.509042	0.707773	0.317784
7	-0.634472	0.080980	0.940777	-0.640136	-0.205756	2.116685	-0.567951	-0.600604	-0.600323	-0.669138
8	0.902360	-1.715794	-0.817710	-0.640136	1.682246	2.812204	0.880322	0.804351	1.038391	0.482512
9	-0.959155	0.979367	0.354615	-0.032819	-1.026626	-0.201711	-1.074402	-1.012248	-1.117812	-0.873677

Question 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 Rsquare, RMSE.

Note :

Before the Data splitting and further moving into the Model Building parts, we must remember that we have many categorical variables in the dataset, and they must be converted to numerical type before proceeding further.

First off, We need to encode the data based on the information provided like:

Cut : Quality is increasing order Fair, Good, Very Good, Premium, Ideal.

Clarity : In order from Best to Worst, FL = flawless, I3= level 3 inclusions) FL, IF, VVS1, VVS2, VS1, VS2, SI1, SI2, I1, I2, I3

Color : With D being the best and J the worst

Based on the provided information, using the "np.where()" function the values are encoded properly and feeded back into the data frame.

Data Split: Splitting the data into test and train:

We are going to split the data with Train : 70% and Test : 30% with random_state = 1 which is going to be common for both the datasets(treated with outliers and without treating outliers).

After the Train-Test split the shape of the dataset is going to be:

```
The training set for the independent variables: (18847, 9)
The training set for the dependent variable: (18847, 1)
The test set for the independent variables: (8078, 9)
The test set for the dependent variable: (8078, 1)
```

Here,

X_train denotes the 70% training dataset that was picked up in random(especially random state = 1) with 9 columns (except the target column called “price”).

X_test denotes 30% test dataset that was picked up in random(especially random state = 1) with 9 columns (except the target column called “price”).

y_train denotes the 70% training dataset that was picked up in random(especially random state = 1) with only the target column called “price”.

y_test denotes 30% test dataset that was picked up in random(especially random state = 1) with only the target column called “price”.

Model Building - Linear Regression:

There are two methods by which the Linear Regression models can be built.

- **Sklearn**
- **Statsmodels**

Sklearn:

First off, we need to import the necessary libraries from the sklearn package necessary to perform the linear regression.

There are no notable Hyper Parameters involved in the model building and therefore we can quick move on the fitting the model built with X_train and y_train.

After the model is fitted with the data, we can investigate the coefficients of each of the features involved to predict the price.

Coefficients:

In linear regression, coefficients are the values that multiply the predictor values. The sign of each coefficient indicates the direction of the relationship between a predictor variable and the response variable. A positive sign indicates that as the predictor variable increases, the response variable also increases and vice versa.

```
The coefficient for carat is 1.1844754737920873
The coefficient for cut is 0.03639268233536705
The coefficient for color is 0.1344726552693927
The coefficient for clarity is 0.2075368704812582
The coefficient for depth is 0.012457420533777791
The coefficient for table is -0.0093756794027128
The coefficient for x is -0.4379147636103137
The coefficient for y is 0.5035061306688992
The coefficient for z is -0.19477298642122876
```

Intercept:

The intercept is the expected mean value of Y when all X(features) = 0.

```
The intercept for our model is -0.0002919730885865852
```

The intercept value that we obtained here maybe a negative value, but it is very minuscule.

We need to start looking into the first performance metric(R Squared) to know how much well the model is built.

Accuracy for the Training Data(R Squared – Train):

```
The Accuracy of the Training data is: 0.9312287832968958
```

Accuracy for the Testing Data(R Squared – Test):

```
The Accuracy of the Testing data is: 0.9316264205922415
```

We can see that the Testing data has slightly more accuracy than the Training data which is good, and it shows that there is no overfitting of the data. Also, we have got more than 93% as accuracy for both training and testing which is another good news.

Next logical step is move on to the next performance metrics on how good the model that is built actually is.

Root Mean Squared Error:

Generally we can use two methods to calculate RMSE, one is the use of metrics library to calculate the MSE value followed by performing a Square root of it and the next one is the more general one i.e, taking standard deviation of residuals. Residuals are a measure of how far from the regression line data points are. RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of

best fit. The lesser the value of RMSE the better the model is and more accurate the predictions are found out to be.

RMSE for the Training Data:

```
The RMSE of the Training data is: 0.26163092383676895
```

RMSE for the Testing Data:

```
The RMSE of the Testing data is: 0.26289869575092717
```

RMSE value ≥0.5 reflects the poor ability of the model to accurately predict the data. Since both the Training and Testing data both have RMSE value lesser than 0.5 we can clearly see that the model built is good to go.

Statsmodels:

Like mentioned before there is one more way by which the linear regression model can be built which is by using the Statsmodels library.

Similar to that of sklearn, we can use the train and test sets to build the model.

To build the model we can use expression or formula with the target variable as “price” and using the rest as the independent variables in the following way.

Expr = ‘(Target Variable) ~ ID1 + ID2 + ID3 + …… + IDN’

Where IDx = Independent Variables.

The default OLS model is applied here(OLS – Ordinal Least Squares) and the expression is fitted into this model to get the results.

We can us the “params” variable to access the coefficients similar to that of the sklearn method.

```
Intercept   -0.000292
carat        1.184475
cut          0.036393
color        0.134473
clarity      0.207537
depth        0.012457
table       -0.009376
x           -0.437915
y            0.503506
z           -0.194773
dtype: float64
```

We can see from the above results that we have the same as that of sklearn method which is not that surprising. We can use the Summary() to determine all the results obtained in the model as a whole.

Summary Details:

OLS Regression Results

Dep. Variable:	price	R-squared:	0.931
Model:	OLS	Adj. R-squared:	0.931
Method:	Least Squares	F-statistic:	2.834e+04
Date:	Fri, 30 Oct 2020	Prob (F-statistic):	0.00
Time:	15:16:14	Log-Likelihood:	-1472.3
No. Observations:	18847	AIC:	2965.
Df Residuals:	18837	BIC:	3043.
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-0.0003	0.002	-0.153	0.878	-0.004	0.003
carat	1.1845	0.011	107.642	0.000	1.163	1.206
cut	0.0364	0.002	15.498	0.000	0.032	0.041
color	0.1345	0.002	66.558	0.000	0.131	0.138
clarity	0.2075	0.002	97.724	0.000	0.203	0.212
depth	0.0125	0.004	3.196	0.001	0.005	0.020
table	-0.0094	0.002	-3.851	0.000	-0.014	-0.005
x	-0.4379	0.044	-9.963	0.000	-0.524	-0.352
y	0.5035	0.043	11.738	0.000	0.419	0.588
z	-0.1948	0.028	-6.979	0.000	-0.249	-0.140

Omnibus:	2652.224	Durbin-Watson:	2.004
Prob(Omnibus):	0.000	Jarque-Bera (JB):	9564.804
Skew:	0.690	Prob(JB):	0.00
Kurtosis:	6.206	Cond. No.	62.9

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Each of the values mentioned above forms some sort of significant meaning to the overall performance of the model.

Just like the coefficients mentioned for the stats model's method using the params variable we can see the coefficients(along with the standard error) from the summary from above as well.

Just like the R-Squared value from the sklearn method we have similar results found in the summary of the statsmodels. The adjusted R squared value is found to be the same of the R Squared value which says that the model build is a good one and needs no more changes to the data to increase the performance. The P-Value of the total model is very much less than that of 0.05 as well as each of the individual independent variables which shows that is the model built is a good one.

We have many more variables found on the bottom of the summary which further adds to the findings from the R-Squared values such as skewness which is found to be "0.690" (positively skewed), Durbin-Watson which is found to be "2.004" (1-2 is normal).

Residuals check:

```
5030    -0.417840
12108   -0.012716
20181   -0.027577
4712    -0.009196
2548     0.282797
           ...
10965    0.031544
17309    0.580020
5193    -0.108470
12182   -0.014034
235      0.041586
Length: 18847, dtype: float64
```

A residual is the vertical distance between a data point and the regression line. Each data point has one residual. They are positive if they are above the regression line and negative if they are below the regression line. If the regression line passes through the point, the residual at that point is zero.

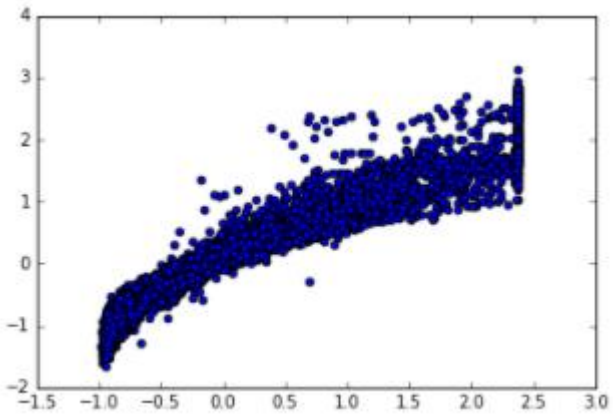
Just like the RMSE calculated from the sklearn model, we can also calculate the RMSE values from the statsmodels as well.

From the Square Root of MSE we got : 0.26163092383676884

From the Square Root of Resid we got : 0.2617003606393495

The results obtained from both the methods are more or less the same which is not surprising.

Scatter Plot(Actual and Predicted Values):



The above scatter plot is based on the scaled data and that's why range for both the Predicted and the actual Prices is between -3 to +3. The X- axis here is the Actual Prices of the test data and the Y axis is the Predicted values of the test data. Because of the huge volumes of test data the scatter plot seem to be to crowded which we can correct by decreasing the test data amount significantly but nonetheless the scatter plot is an effective one to see the spread of data.

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

Based on the model built above, we can go ahead to predicting the prices of the test values to check how far along are we to the original values present, so that we can make some recommendations and form business insights.

Prediction of the Prices:

Finally, we can move on to the prediction of the prices from the Test Values.

	Predicted	Actual
0	8287.158769	8758.0
1	4857.596683	4718.0
2	13477.002871	11965.0
3	11643.281714	11965.0
4	8618.019952	8165.0
...	...	...
8073	4199.950913	4642.0
8074	4321.270299	4038.0
8075	-28.192887	613.0
8076	408.627314	844.0
8077	5238.444168	5198.0

8078 rows × 2 columns

	Predicted	Actual
0	1.313408	1.449240
1	0.324015	0.283743
2	2.810623	2.374426
3	2.281614	2.374426
4	1.408858	1.278166
...	...	...
8073	0.134291	0.261818
8074	0.169290	0.087570
8075	-1.085484	-0.900507
8076	-0.959466	-0.833866
8077	0.433885	0.422218

8078 rows × 2 columns

Please note that the predictions that are found here belongs to the non-scaled data so that we can clearly see the Actual values present in the test data. The specific need of the project presented scaling as necessary to proceed forward and the we have thus built the models above to show accuracy and how it performs in performance metrics. The above Predicted results shows somewhat similar results to that of the actual prices which says that the model built is actually a good one.

Collinearity check:

One of the major issues that are usually found in the linear regression(or more of an assumption) is the multicollinearity issue. Multicollinearity is nothing but having more correlation between the Independent variables which makes them redundant for the model.

Here we use a method called Variance Inflation Factor(VIF) which can let us know whether the features have multicollinearity issue or not.

Mathematically, the VIF for a regression model variable is equal to the ratio of the overall model variance to the variance of a model that includes only that single independent variable.

	Variable_Inf_factor	features
0	32.891284	carat
1	1.509590	cut
2	1.119859	color
3	1.241452	clarity
4	4.453927	depth
5	1.618348	table
6	417.370935	x
7	398.581660	y
8	234.837061	z

The lesser the VIF values the lesser the feature suffers from Multicollinearity.

Usually that values between 1 – 5 usually mean that the feature is better off from multicollinearity and the values above 5 suffers from it.

From the above results we can see that “cut”, “color”, “clarity”, “depth” and “table” is somewhat better when compared to rest of the features as they have huge values which shows that there is a heavy influence on multicollinearity which needs to be treated.

The best 5 attributes that are most important:

To find out the best attributes that are the most important let us first take a look on the linear equation that is produced by the model for the target variable “Price”.

Price = (-0.0) * Intercept + (1.18) * carat + (0.04) * cut + (0.13) * color + (0.21) * clarity + (0.01) * depth + (-0.01) * table + (-0.44) * x + (0.5) * y + (-0.19) * z

From the above results we can say that the five most important attributes based on order are:

1. Carat
2. Y
3. X(when taken a look on negative weightage)
4. Clarity
5. Z

Based on these attributes the business side can take informed decision for price calculations, For Ex: Increasing the “Carat” can increase the price with significant difference or even play around with Y(width or even X(length- inversely proportional)) or Clarity to look for market trends. The most profitable or high profitable stones can be distinguished with these 5 attributes and the profit margins can be increased in these fashions.

Still there can be some adjustments made to the models to show better results nonetheless like having more samples for predictions as increasing the sample size has always proven to show better results for the model.

Problem 1 Summary:

- **1.1)** Data ingestion was performed. All the basic EDA along with the univariate and the Bivariate analysis were performed and analysed including the descriptive statistics.
- **1.2)** Null value condition check was performed and the values which are equal to zero were checked along with identifying their meaning and dropped. Scaling was also performed in this case.
- **1.3)** Data was encoded (having string values) for Modelling. Data Splitting: Splitting the data into test and train (70:30) was performed. Linear regression as applied to the data. Performance Metrics were calculated like: Checking the performance of Predictions on Train and Test sets using Rsquare, RMSE.
- **1.4)** Inference on Basis on these predictions along with the business insights and recommendations were provided.

Conclusion:

Looking into the data of "cubic_zirconia.csv", we saw some interesting insights from EDA methods. We were able to split the data properly and build the linear regression model. There were a lot of thoughts poured into the optimization and the analysis of the results via various performance metrics, but we were able to properly utilize the given data to provide model on zirconia data that provides proper predictions on the price details for every cubic zirconia. Also, as an added advantage we were also able to provide some insights on improving this data to further improve the performance as well.

Problem 2: Logistic Regression and Linear Discriminant Analysis

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.

Question 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.

Exploratory Data Analysis:

	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

The Dataset consists of 8 variables(First column being unnecessary as its insignificant for the analysis of the problem).

Numerical Columns : Salary, age, educ, no_young_children and no_older_children.

Non-Numerical Columns : Holliday_Package and foreign.

Descriptive Statistics of the Dataset:

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

Firstly, the column “Unnamed : 0” is removed from the dataset before proceeding further as its insignificant for the analysis. The describe function(describe(include='all')) provides us with the information about how much the data is spread across along with the information on the mean, standard deviation, count along with specific details on non-numerical columns such as unique(number of unique categories), top(most used) and freq(count of number of times of the most used) etc.

We have two categorical variables with most frequency seen in :

Holliday_Package as '**no**'(471) and foreign as **'no'** (656).

From the looks of the values in numeric columns based on the mean values and the median values we can say that the distribution is relatively normal(not exact but close to normal distribution as we have an instances where even the std of a column is greater than the mean(no_young_children and no_older_children)) .

Check for NULL Values:

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

The isnull() and sum() function combined can clearly figure out on whether given dataset has any NULL(N/A) values or not. From the above result it is evident that there are no NULL values and therefore we can proceed further in the Exploratory Data Analysis.

Skewness Check:

```
Skewness of the Dataset:
 Salary               3.103216
age                   0.146412
educ                 -0.045501
no_young_children     1.946515
no_older_children     0.953951
dtype: float64
```

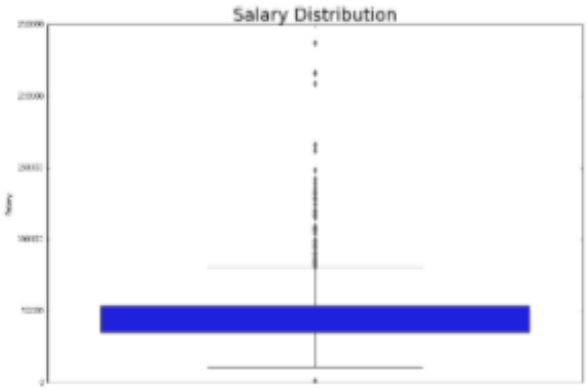
Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. From the above result we can see that only the variable “educ” is negatively skewed compared everything else(which are right skewed with max seen in “Salary”).

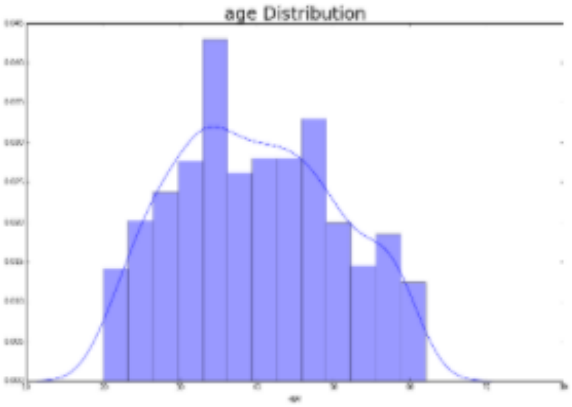
Duplicates Check:

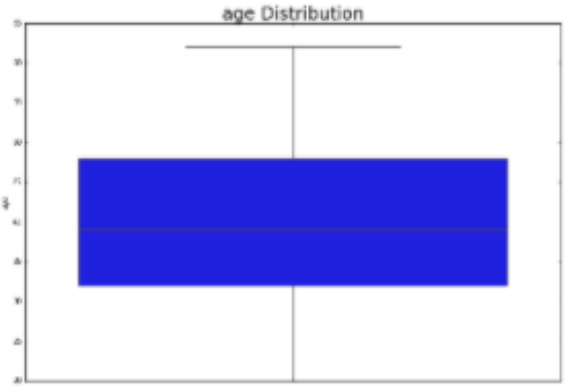
The purpose of this step is the look for any duplicates in the dataset as duplicates are redundant when it comes to analysing the dataset and its generally proposed to be removed. Upon looking into the dataset there was totally **0** duplicated lines present in the dataset which makes it easier to proceed further as there is no need to handle the data for duplicates.

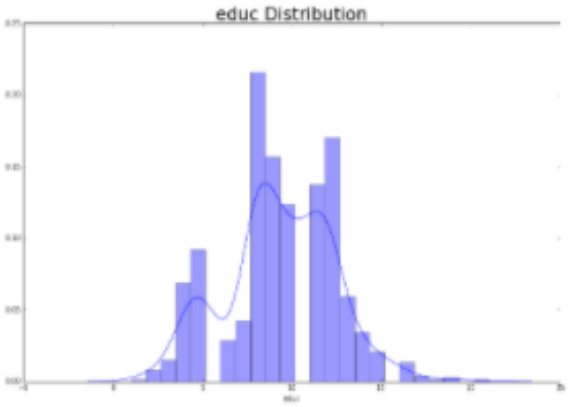
```
Number of Duplicate rows in the Dataset: 0
```

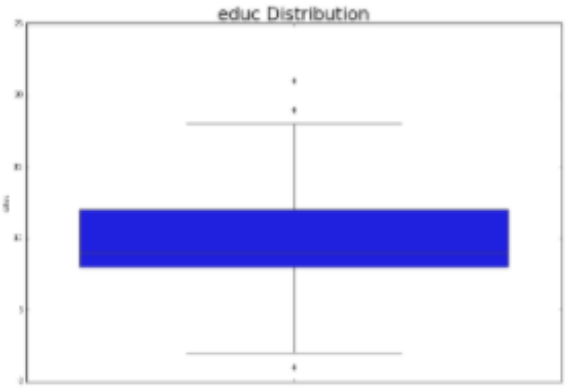
Univariate Data Visualization (Dist-plot and Boxplot analysis):







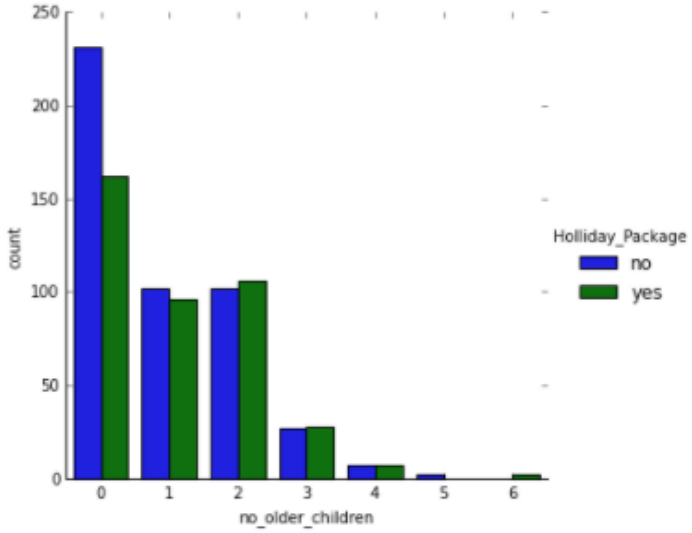




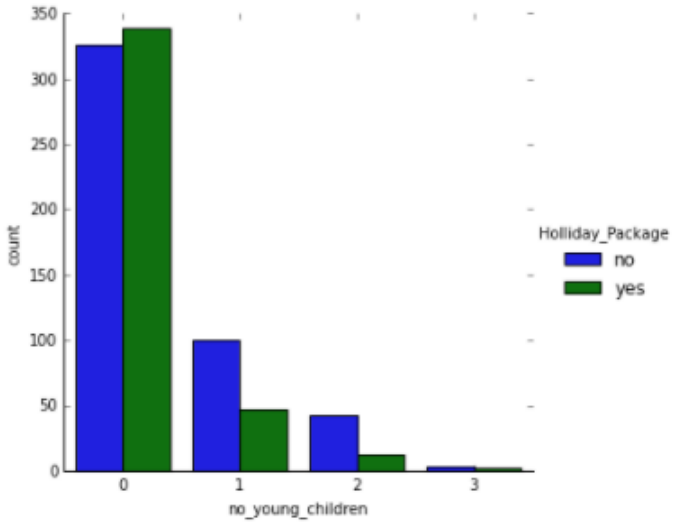
We can see that the three numerical Variables are normally distributed(not perfectly normal though, but still). There are a lot of outliers and is found in all the "Salary" that can be seen from the boxplots on the right too which is normal as in real life we have variations of salary options with employees on either end of income range.

For the purposes of this project we are going to move forward without the outlier treatment for the dataset as the outliers present in these features seems genuine. But its highly recommended to discuss with Business side to know for sure whether to remove certain outliers or not.

Univariate Data Visualization (Catplot Analysis):

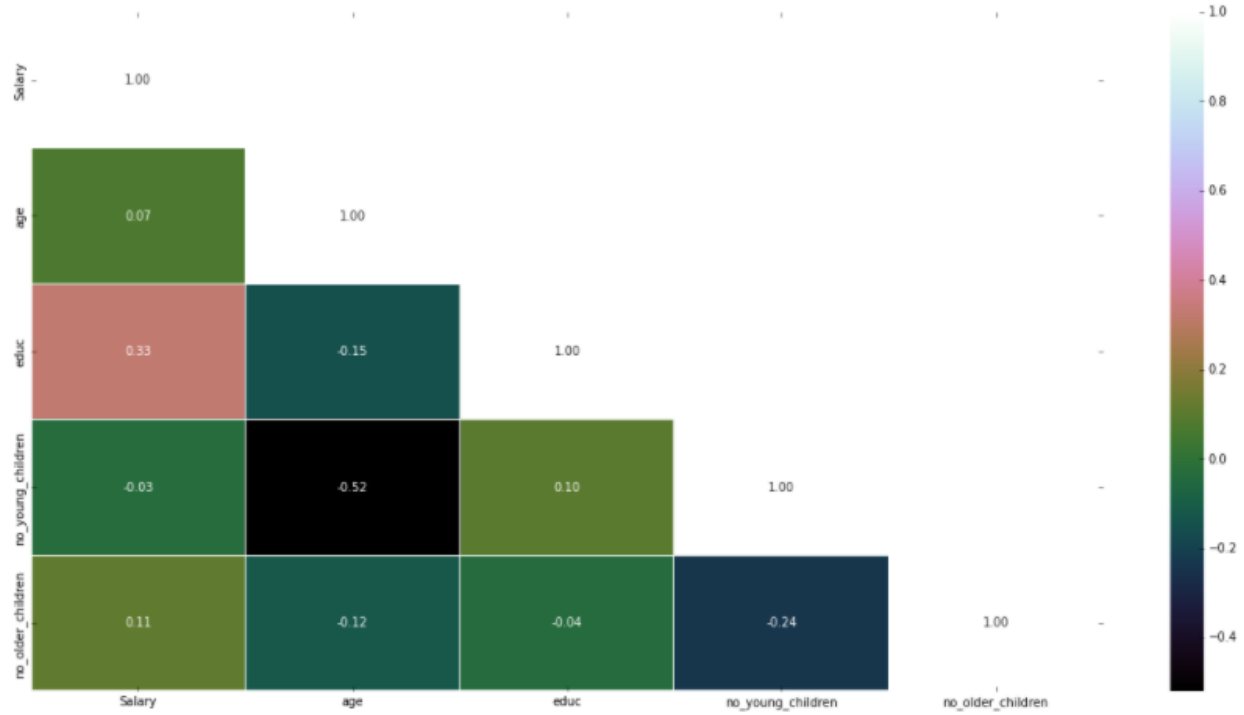


The above plot deals with the no_older_children associated with the dataset and we have segregated that with the "Holliday_Package" preference and we can see that employees with no children seems to top the list with "no" being the maximum preference. But the interesting part is that this trend seems to change when it comes to more than 2 Older children(the count maybe significantly less, but an interesting fact nonetheless).



The above plot is an interesting find as we have an entirely opposite effect on “no” and “yes” when it comes to the younger children category. The trend in the count seems to follow a similar trend, but we have the preference of employees with younger children “0” seem to prefer on the “Holliday_Package” than that of employees not accepting the package.

Bivariate Data Visualization (Heat Map analysis):

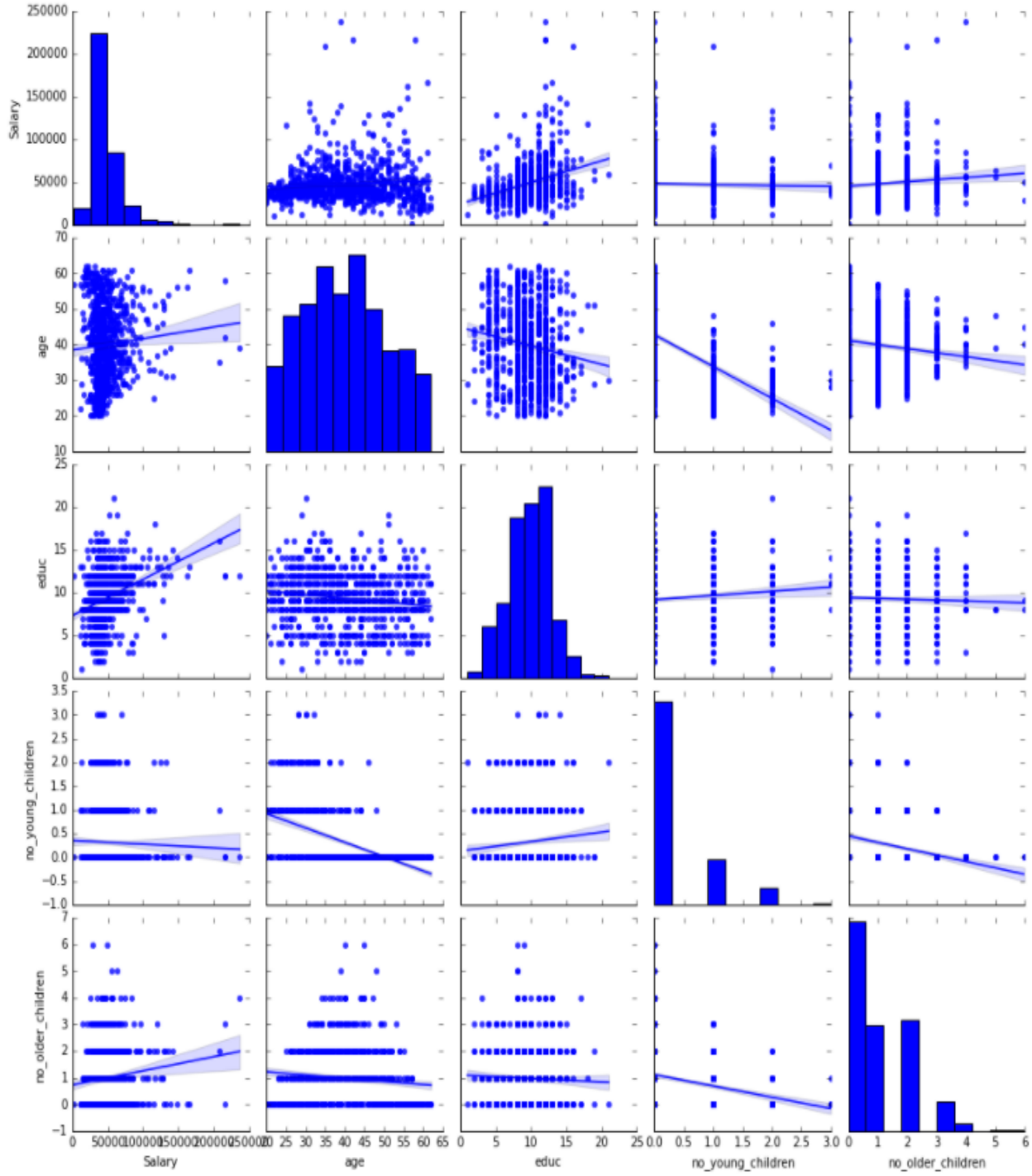


Another most important plot that we must look in the case of analysis is the Correlation plot. The Heat-Map analysis is an visualized part of the Correlation matrix that is used to find the relationship between the variables with numerical values to standpoint on how much they are related as compared to other variables.

The following conclusions can be sought out of the above results:

- The highest correlation is seen between “educ” and “salary” (33%)
- There are no other highly notable positive relationships are between the independent.
- For negative correlations, the maximum was found between “age” and “no_young_children” (52 %) which is actually more than the maximum percentage on positive correlation.
- Other than these there are no high or very significant positive or negative correlation between the variables.
- Note : Correlation doesn’t mean Causation. We need to analyse further to know more about the above results.

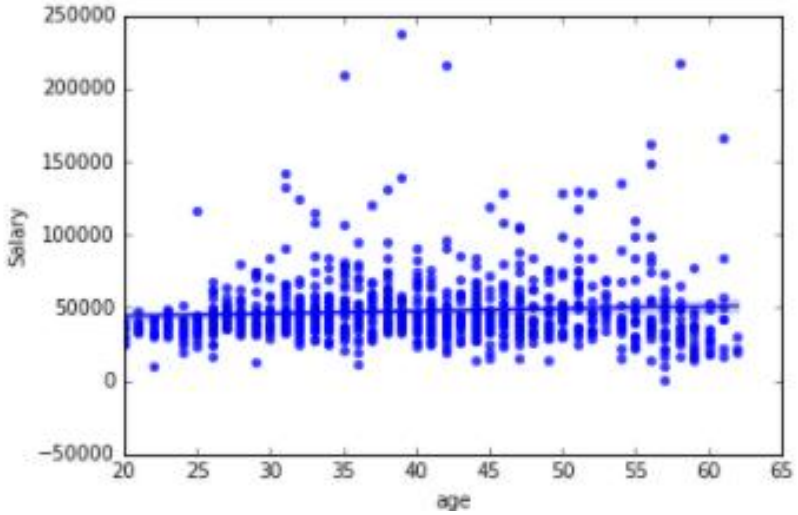
Bivariate Data Visualization (Pair-Plot analysis):



A regular Pair-Plot analysis is first performed on the dataset to see their interaction with each of the other variables. From the above result we can say that there are not many but indeed some strong relationships with the variables. The notable relationships can be seen between the variable's "educ" and "Salary"(which is normal and practical) , "no_older_children" and "Salary" and "age" and "Salary". Also, there are some significant negative relationship seen here as well, such as between, "age" and "no_young_children" , "no_older_children" and "no_young_children" and "age" and "no_older_children".

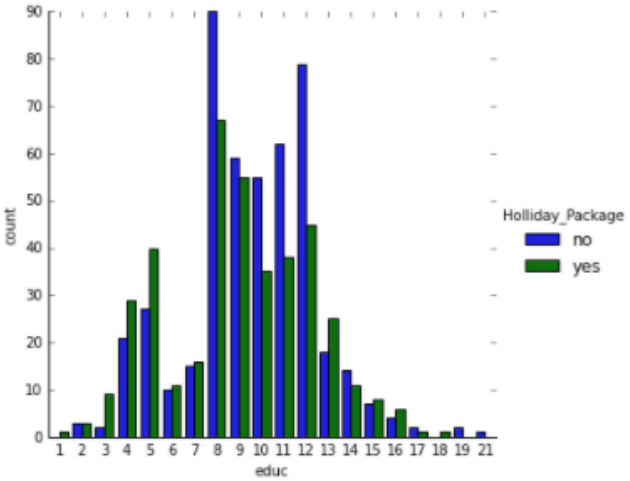
Bivariate Data Visualization (Regression-Plot analysis):

Salary VS Age:



The above plot is a regression plot is that built between “Age” and “Salary” as we can see the trend is increasing(and only slightly) when there is an increase in age. Most of the datapoints lie between 0 and 100000, with only a few very points reaching the 250000 mark which shows the salary trend for the company.

Bivariate Data Visualization (Cat-Plot analysis converted to Bivariate with HUE as “Holliday_Package”):



From the above plot is an interesting plot because we have two groups seen here when it comes to the number of years of formal education. One is the group between the educ years between “8 – 12” where we can see there is a clear preference given to not opting the “Holliday_Package” and we can see the rest of the groups having the opposite effect where the preference is a “yes” on the package. The count for the latter group is less though compared to the first.

Question 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).

First off, the given requirement is to not scale the data(using Z-score) which is going to be the case here when it comes to modelling.

Note : Before the Data splitting and further moving into the Model Building parts, we must remember that we have two categorical variables in the dataset, and they must be converted to numerical type before proceeding further.

Encoding the Data:

When it comes to encoding the data, there are two major features that have to be treated here.

- Holliday_Package
- foreign

When checked in the descriptive statistics, we saw that both of these variables had only two categories and it's the same ("Yes" and "No"). Since there are only two variables, we can proceed these with many methods. The method which is applied here is called the "**Label Encoding**" method.

The Label encoding method converts the "Yes" and "No" string values in the data into "1" and "0" respectively (that comes into the Label encoding package itself as there it forms a ranking system and assigns value starting from 0 and since there are only two values here) and 1 are applied). But the limitation of Label encoding is that it can't be applied in all situations and especially in situations where ranking doesn't apply at all. For EX: for countries we can't rank them from 0 to any number of countries unless there is a specific ask from the business team and therefore the Label encoder is not applied formally here. My recommendation is to assign each category into the business required format using the np.where() function in those cases.

After Encoding the data, the next step is to go for data splitting and start preparing the models. We also must be clear on the column that we use as a target variable when creating the models. The column that is focused on is "Holliday_Package", i.e, whether the employee has claimed the Holliday_Package or not.

Also, its important to know about the percentage of distribution of the Target(Holliday_Package) variable as to identify whether there are any class imbalance there or not. Class Imbalance can hugely affect the performance of a model as any huge shortage of a particular value can become hard in training and identifying the variable in real case scenario.

Based on the above statement,

```
0    0.540138
1    0.459862
Name: Holliday_Package, dtype: float64
```

We can see that the values are of 54-46 ratio which is of class imbalance(but not affecting too much as its accepted if the ration is anywhere above 70-30 ratio) but can be ignored as the difference is not that much to be actually significant in affecting the model.

Data Split: Splitting the data into test and train:

We are going to split the data with Train : 70% and Test : 30% with random_state = 1 which is going to be common for both the datasets(treated with outliers and without treating outliers).

After the Train-Test split the shape of the dataset is going to be:

```
The training set for the independent variables: (610, 6)
The training set for the dependent variable: (610,)
The test set for the independent variables: (262, 6)
The test set for the dependent variable: (262,)
```

Here,

X_train denotes the 70% training dataset that was picked up in random(especially random state = 1) with 6 columns (except the target column called “Holliday_Package”).

X_test denotes 30% test dataset that was picked up in random(especially random state = 1) with 6 columns (except the target column called “Holliday_Package”).

y_train denotes the 70% training dataset that was picked up in random(especially random state = 1) with only the target column called “Holliday_Package”.

y_test denotes 30% test dataset that was picked up in random(especially random state = 1) with only the target column called “Holliday_Package”.

Model Building - Logistic Regression:

After the Train-Test split is performed, the next step is to import the libraries required for Logistic regression and fitting the train data to the model. But before fitting the model it is important to know about the hyperparameters that is involved in model building.

Parameters:

- penalty
- solver
- max_iter
- tol

These are the parameters that are usually tweaked to bring the best possible model for any dataset(there are more parameters to consider, but these usually form most of the criterion for better built model).

For tweaking these parameters, a method called “GridSearchCV” is performed that can-do multiple combinations of these parameters simultaneously and can provide us with the best optimum results.Based on the tweaking of parameters the best possible combination came out to be

from

- 'penalty': ['l2','none','l1','elasticnet'],
- 'solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'],
- 'max_iter': [100,1000,10000,100000],
- 'tol': [0.0001,0.00001,0.001]

To

- 'penalty' = 'l2',
- 'solver' = 'newton-cg',
- 'max_iter' = 100,
- 'tol' = 0.0001

There can be many such addition to the GridSearchCV to provide the optimum result, but the system cannot handle such aa huge combination especially when mentioned CV = 3(CV is nothing but Cross Validation the randomly shifts the training and test location in the dataset). The reason being the above example handles 4*5*4*3*3 = 720 combinations of Logistic Regression models to decide on the optimum one.

After figuring out the parameters that we are going to use to the model, we rebuild the model with the aforementioned parameters and refit the model to check the changes that we have brought.

For the next step we can create a DataFrame that can clearly define for each record on how much likelihood that they belong to category “0” or “1”(0 being not preferring Holiday Package and 1 being preferring Holiday Package).

	0	1
0	0.753599	0.246401
1	0.287308	0.712692
2	0.888743	0.111257
3	0.974783	0.025217
4	0.499096	0.500904
5	0.738768	0.261232
6	0.904156	0.095844
7	0.665797	0.334203
8	0.462652	0.537348
9	0.635633	0.364367

We can clearly see from the above result that the newly built model is tested with the "Test" dataset(30% - 262 records) and the results on probability on which side each test record belongs to is shown.

Statsmodels can also be used here in building the Logistic regression model to more about the statistics of the model in the background.

Logit Regression Results

Dep. Variable:	Holliday_Package	**No. Observations:**	872
Model:	Logit	**Df Residuals:**	865
Method:	MLE	**Df Model:**	6
Date:	Sun, 01 Nov 2020	**Pseudo R-squ.:**	0.1281
Time:	17:09:08	**Log-Likelihood:**	-524.53
converged:	True	**LL-Null:**	-601.61
Covariance Type:	nonrobust	**LLR p-value:**	1.023e-30

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

From the above result we got to know more on the mathematical sense on how good the model has performed.

Pseudo R^2 = 0.1281 shows that the model didn't perform that well, but we can look into the sklearn model that we have built to rectify that.

The logistic regression equation is :

Holliday_Package = (2.32589) * Intercept + (-2e-05) * Salary + (-0.04816) * age + (0.03917) * educ + (-1.31734) * no_young_children + (-0.02038) * no_older_children + (1.32157) * foreign

Model Building – Linear Discriminant Analysis:

For this model a we use the Train-Test again for model building. We go for the packages from sklearn that we can use to build the LDA model. This process is continued by fitting the model with the data. When we are defining the function, there won't be many parameters like Logistic Regression here. We are going to stick with the default parameters when defining the function. The default cut-off we are using here is 0.5 i.e., if we get the result anywhere below 0.5 its going to go under 0 and vice versa.

The default might be 0.5, but we can indeed play around with the cut off values and try to find the cut-off that can produce us with best accuracy or F1 scores(which will be explained down the line in the project report).

For the next step we can create a DataFrame that can clearly define for each record on how much likelihood that they belong to category "0" or "1"(0 being not preferring Holiday Package and 1 being preferring Holiday Package).

	0	1
0	0.736312	0.263688
1	0.277893	0.722107
2	0.887243	0.112757
3	0.967803	0.032197
4	0.523170	0.476830
5	0.739020	0.260980
6	0.889165	0.110835
7	0.674832	0.325168
8	0.397757	0.602243
9	0.641310	0.358690

We can clearly see from the above result that the newly built model is tested with the "Test" dataset(30% - 262 records) and the results on probability on which side each test record belongs to is shown.

The coefficients found out for the LDA model are:

```
The coefficient for Salary is -1.475495480988139e-05
The coefficient for age is -0.054303783061133676
The coefficient for educ is 0.07596537387390201
The coefficient for no_young_children is -1.4285464350098693
The coefficient for no_older_children is -0.04635929801474013
The coefficient for foreign is 1.6239034671206718
```

Intercept found out to be:

```
The intercept of the model is  [2.08934324]
```

Question 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.

Performance Metrics:

Usually there are many performance metrics that are generally used in assessing the strength of the model to understand how the model has performed as well as to take an informed decision on whether to go forward with the model in the real time scenario or not.

The industrial standards are generally the following methods:

- Classification Accuracy.
- Confusion Matrix.
- Classification Report.
- Area Under ROC Curve(visualization) and AUC Score

1. Classification Accuracy:

Classification accuracy is the number of correct predictions made as a ratio of all predictions made.

2. Confusion Matrix:

The Confusion Matrix is a table that presents predictions on the x-axis and accuracy outcomes on the y-axis. The cells of the table are the number of predictions made by a machine learning algorithm.

3. Classification Report:

Its a convenience report that is created when working on classification problems to give us a quick idea of the accuracy of a model using a number of measures.

4. Area Under ROC Curve(visualization) and AUC Score:

The ROC Curve measures how accurately the model can distinguish between two things(generally classification information). Larger the curve, the better the model. AUC measures the entire two-dimensional area underneath the ROC curve. This score gives us a good idea of how well the classifier will perform.

Logistic Regression:

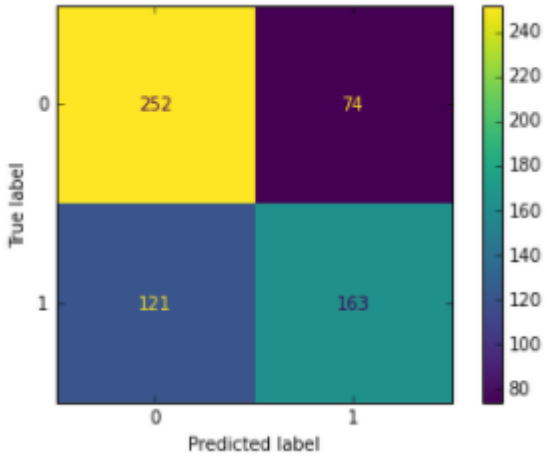
Accuracy:

Training Accuracy : 0.680327868852459

Testing Accuracy : 0.6450381679389313

Confusion Matrix:

For Training :



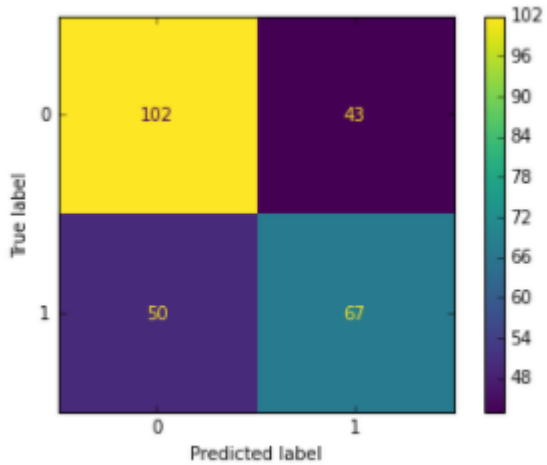
True Negative : 252

False Positive : 74

False Negative : 121

True Positive : 163

For Testing:



True Negative : 252

False Positive : 74

False Negative : 121

True Positive : 163

Classification Report:

For Training :

	precision	recall	f1-score	support
0	0.68	0.77	0.72	326
1	0.69	0.57	0.63	284
accuracy			0.68	610
macro avg	0.68	0.67	0.67	610
weighted avg	0.68	0.68	0.68	610

```
LR_train_precision  0.69
LR_train_recall  0.57
LR_train_f1  0.63
```

For Testing :

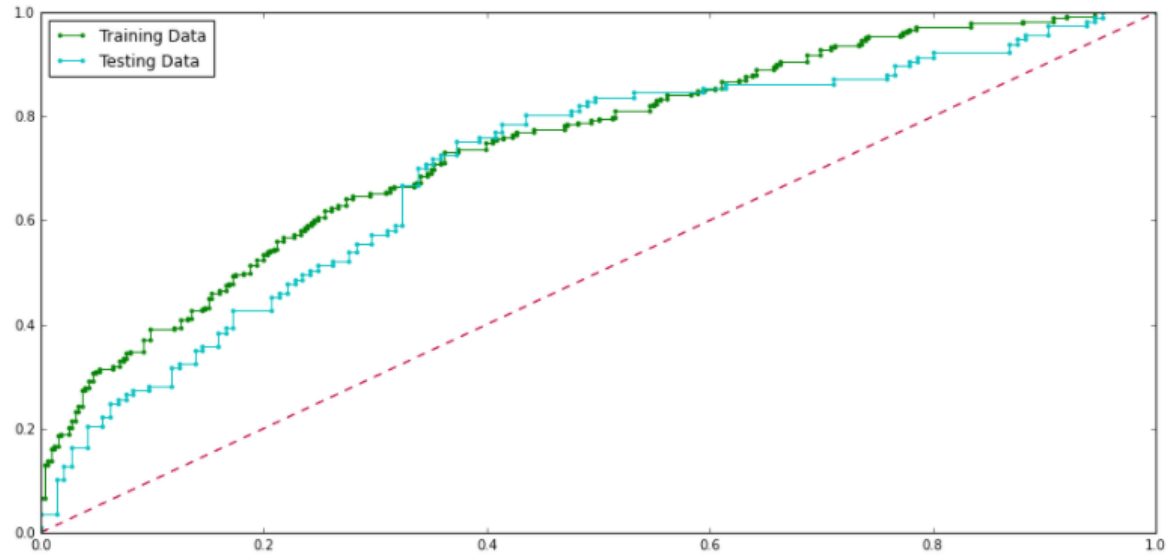
	precision	recall	f1-score	support
0	0.67	0.70	0.69	145
1	0.61	0.57	0.59	117
accuracy			0.65	262
macro avg	0.64	0.64	0.64	262
weighted avg	0.64	0.65	0.64	262

```
LR_test_precision  0.61
LR_test_recall  0.57
LR_test_f1  0.59
```

Area Under ROC Curve(Visualization) and AUC Score:

For both Training and Testing:

```
AUC for Training Data: 0.743
AUC for testing: 0.705
```



Linear Discriminant Analysis:

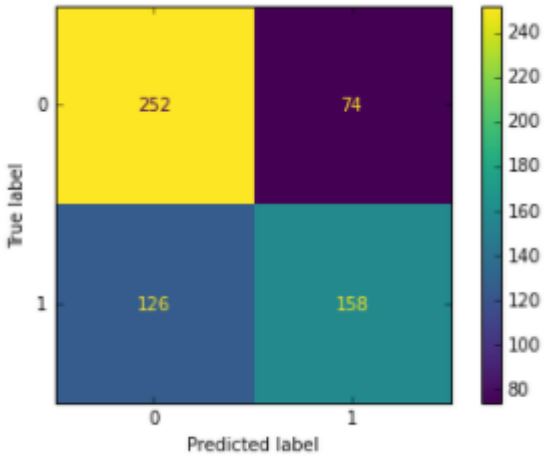
Accuracy:

Training Accuracy : 0.6721311475409836

Testing Accuracy : 0.6412213740458015

Confusion Matrix:

For Training :



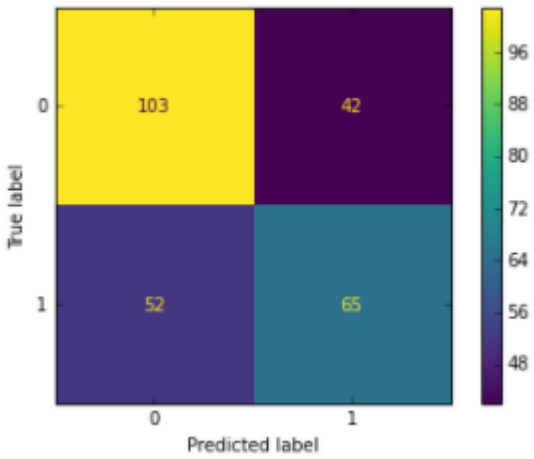
True Negative : 252

False Positive : 74

False Negative : 126

True Positive : 158

For Testing:



True Negative : 103

False Positive : 42

False Negative : 52

True Positive : 65

Classification Report:

For Training :

	precision	recall	f1-score	support
0	0.67	0.77	0.72	326
1	0.68	0.56	0.61	284
accuracy			0.67	610
macro avg	0.67	0.66	0.66	610
weighted avg	0.67	0.67	0.67	610

```
LDA_train_precision  0.68
LDA_train_recall  0.56
LDA_train_f1  0.61
```

For Testing :

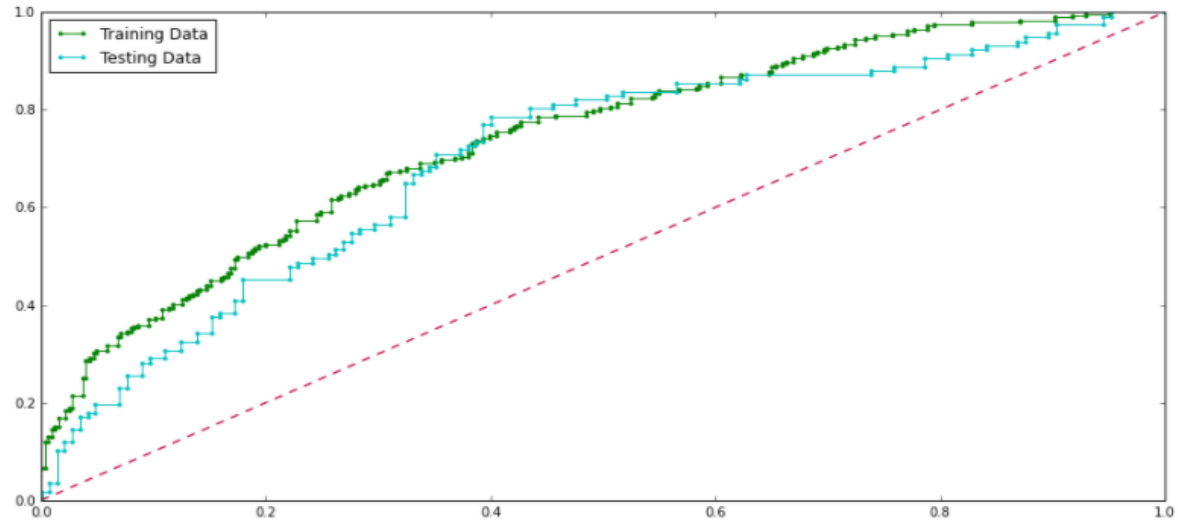
	precision	recall	f1-score	support
0	0.66	0.71	0.69	145
1	0.61	0.56	0.58	117
accuracy			0.64	262
macro avg	0.64	0.63	0.63	262
weighted avg	0.64	0.64	0.64	262

```
LDA_train_precision  0.61
LDA_train_recall  0.56
LDA_train_f1  0.58
```

Area Under ROC Curve(Visualization) and AUC Score:

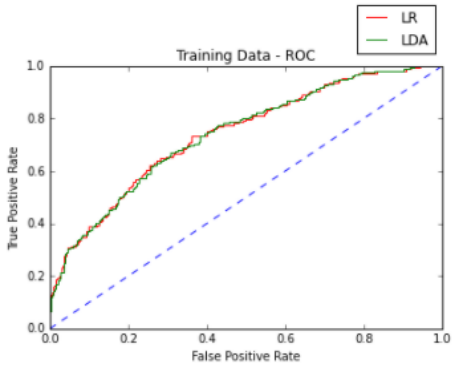
For both Training and Testing:

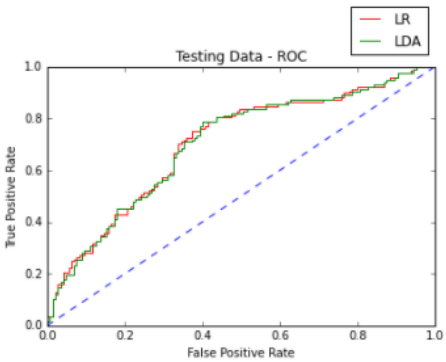
```
AUC for Training Data: 0.742
AUC for testing Data: 0.703
```



Model Comparison - Logistic Regression + LDA:

	LR Train	LR Test	LDA Train	LDA Test
Accuracy	0.680	0.645	0.672	0.641
AUC	0.743	0.705	0.742	0.703
Recall	0.570	0.570	0.560	0.560
Precision	0.690	0.610	0.680	0.610
F1 Score	0.630	0.590	0.610	0.580





The above result is a combined form of some of the important performance metrics that is used to determine whether a model works better or not. We have used it in such as way as to combine the above said results for both Logistic Regression and Linear Discriminant Analysis so that we can compare the performance of the model's side by side to reach an informed decision. From the above results we can come to a following conclusions:

- Accuracy: When it comes to accuracy of the models, the **logistic regression outperforms LDA** by a slight difference(mainly in train than in test, but significant nonetheless) making it a model to go for when it comes to this data.
- AUC score produces no difference results compared to Accuracy and therefore the Logistic regression should be preferred when it comes to AUC scores as well.
- Other than accuracy the main detail that is looked upon when it comes to the performance metrics are **Precision and Recall.** By looking at the results we can see that Recall is taken over by logistic regression by a considerable margin, but when it comes to precision the logistic regression works better here too, but the test data shows more or less the same results to that of LDA.
- Finally, the F1 scored is looked upon as well as its hard to pinpoint the Precision and Recall values and a combined effect is seen in the F1 scores. By this metric, the Logistic Regression takes a comfortable lead in both the training and testing sets.

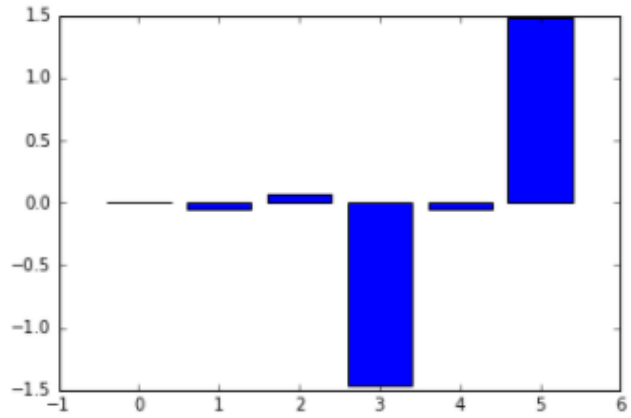
In conclusion, we can observe that the Logistic Regression should be the preferred model as it has shown the ability to predict better than that of LDA. That doesn't mean that LDA hasn't performed bad as well. The pure numbers show that was seen above states that the Logistic regression model is the better optimized model when compared to Linear Discriminant Analysis (By a slight margin).

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

To provide inferences lets look at some of the results as shown below.

The best 5 attributes that are most important:

```
Feature: 0, Score: -0.00002
Feature: 1, Score: -0.05296
Feature: 2, Score: 0.07150
Feature: 3, Score: -1.45908
Feature: 4, Score: -0.04638
Feature: 5, Score: 1.47623
```



The above scores and plot are from the coefficients extracted to form the feature importance of all the variables present in the data.

Here,

- Feature 0 = Salary
- Feature 1 = age
- Feature 2 = educ
- Feature 3 = no_young_children
- Feature 4 = no_older_children
- Feature 5 = foreign

From the above plot we can see importance of all the features that is being brought to the predictions(both negative and positive). Here the calculation that can get us the odds for the feature is performing exponential to the feature scores.

The top important features in order of odds (and feature importance from coefficients) are(in order): (the same order seen from the plot values as well)

1. foreign (+ve) – 337 % increase per unit
2. no_young_children (-ve) – 77 % decrease per unit
3. educ (+ve) – 7 % increase per unit
4. age (-ve) – 5 % decrease per unit
5. no_older_children (-ve) – 4.6 % decrease per unit

Insights and recommendations:

- Firstly when it comes to foreign variable we can see a huge importance given to that feature which says that being a foreigner is very importance when it comes to predicting whether an employee would prefer to take the Holiday Package(or Tour Package) or not. This is a subjective one, as just foreigner(not mentioning which country he/she hails from) doesn't say much on the ability to prefer holiday packages, but nonetheless given much importance. The relation was a positive one in this case.

- Secondly when it comes to no_young_children variable we can see a huge importance given to that feature but in the opposite trend to that of the foreign variable. It shows that being employees with younger children(preferably less than 7) forms a negative trend when it comes to predicting whether an employee would prefer to take the Holiday Package(or Tour Package) or not. This is not surprising as the employees might prefer to save money during those times with their children to better opt for holiday packages in the future. The relation was a negative one in this case.

- Thirdly when it comes to educ variable we can see an average importance given to that feature but in the positive trend. It shows that being employees with increase in number of years of formal education forms a positive trend when it comes to preference to take the Holiday Package(or Tour Package). This is not surprising as the employees who has many years of formal education will definitely prefer or have the means to do so. The relation was a positive one in this case.

- Fourthly when it comes to age variable, we can see a small importance given to that feature but in the negative trend. It shows that being employees with increase in number of age forms a negative trend when it comes to preference to take the Holiday Package(or Tour Package). This shows that with a small percentage the employees who are young prefer to opt for the package. The relation was a negative one in this case.

- Finally, when it comes to no_older_children variable, we can see a small importance given to that feature but in the negative trend just like age. It shows that being employees with number of older children forms a negative trend when it comes to preference to take the Holiday Package(or Tour Package). This shows that with a small percentage the employees who have older children forms a negative trent when it comes to opting for the package, but better when it compared to the employees with younger children which shows change in trend. The relation was a negative one in this case though.

Problem 2 Summary:

- **2.1)** Data ingestion was performed. All the basic EDA along with the univariate and the Bivariate analysis were performed and analysed including the descriptive statistics and null value check. The inferences were written as well.
- **2.2)** Scaling was not performed as per the requirement. The data (having string values)was encoded for Modelling. The data was splitted into train and test (70:30). Logistic Regression and LDA (linear discriminant analysis) were applied.
- **2.3)** Performance Metrics: The performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model were checked properly. Both the models were compared, and inferences were written on which model is the best one.
- **2.4)** Inference on Basis on these predictions along with the business insights and recommendations were provided.

Conclusion:

Looking into the data of "Holiday_Package.csv", we saw some interesting insights from EDA methods. We were able to split the data properly and build both the logistic regression and LDA models. There were a lot of thoughts poured into the optimization and the analysis of the results via various performance metrics, but we were able to properly utilize the given data to provide model on holiday package data that provides proper predictions on the whether an employee would opt for the package or not. Also, as an added advantage we were also able to provide some insights on this data based on the importance of features to take better informed decisions with the business side.