PREDICTIVE MODELLING PROJECT

Sushmita Kar

Contents

List of Figures and Table

Problem 2:

Problem 1: Linear Regression

You are hired by a company named Gemstone Co Ltd, which is a cubic zirconia manufacturer. You are provided with the dataset containing the prices and other attributes of approximately 27,000 pieces of cubic zirconia (which is an inexpensive synthesized diamond alternative with similar qualities of a diamond).

Your objective is to accurately predict prices of the zircon pieces. Since the company profits at a different rate at different price levels, for revenue management, it is important that prices are predicted as accurately as possible. At the same time, it is important to understand which of the predictors are more important in determining the price.

OBJECTIVE: Accurately predict price of zircon pieces.

The data dictionary is given below.

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.
Colour	Colour of the cubic zirconia. D being the best and J the worst.
Clarity	Clarity refers to the absence of the Inclusions and Blemishes. (In order from Best to Worst in terms of avg price) IF, VVS1, VVS2, VS1, VS2, Sl1, Sl2, l1
Depth	The Height of cubic zirconia, measured from the Culet to the table, divided by its average Girdle Diameter.
Table	The Width of the cubic zirconia's Table expressed as a Percentage of its Average Diameter.
Price	the Price of the cubic zirconia.
X	Length of the cubic zirconia in mm.
Y	Width of the cubic zirconia in mm.
Z	Height of the cubic zirconia in mm.

Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values such as temperature, age, salary, price, etc.

Linear Regression is a type of Regression Analysis which is used for Predictive analysis.

Linear Regression shows Linear relationship between the independent variable(X-axis) and the dependent variable(Y-axis), hence called linear regression.

If there is only one input variable (x), then such linear regression is called **simple linear regression**. And if there is more than one input variable, then such linear regression is called **multiple linear regression**.

The mathematical equation of Linear Regression is

$$E(Y)=\beta_0+\beta_1X$$

Here, Y = dependent variables (target variables),
X= Independent variables (predictor variables),
$\beta_0$and β_1:areintercept and slope coefficients, respectively, and known as the regression parameters.

1.1. Perform exploratory data analysis (EDA). Identified the response and the Predictors. Find duplicate observation or missing data and variables having symmetric or skewed distribution. Perform both univariate and bivariate analyses. Check for outliers and comment on removing or keeping them while model building. Since this is a regression problem, the dependence of the response on the predictors needs to be thoroughly investigated.

Dependent Variable: The main factor in Regression analysis which we want to predict or understand is called the dependent variable. It is also called **target variable**. Here, in this problem our **target variable** is **'price'**.

Independent Variable: The factors which affect the dependent variables, or which are used to predict the values of the dependent variables are called independent variable, also called as a **predictor**. Here our predictor variables are carat, cut, color, clarity, depth, table, x, y, and z.

Zircon.head()

Unnamed: 0	carat	cut	color	clarity	depth	table	x	y	z	price
0	0.3	Ideal	E	SI1	62.1	58	4.3	4.3	2.7	499
1	0.33	Premium	G	IF	60.8	58	4.4	4.5	2.7	984
2	0.9	Very Good	E	VVS2	62.2	60	6	6.1	3.8	6289
3	0.42	Ideal	F	VS1	61.6	56	4.8	4.8	3	1082
4	0.31	Ideal	F	VVS1	60.4	59	4.4	4.4	2.7	779
5	1.02	Ideal	D	VS2	61.5	56	6.5	6.5	4	9502
6	1.01	Good	H	SI1	63.7	60	6.4	6.3	4	4836
7	0.5	Premium	E	SI1	61.5	62	5.1	5.1	3.1	1415
8	1.21	Good	H	SI1	63.8	64	6.7	6.6	4.3	5407
9	0.35	Ideal	F	VS2	60.5	57	4.5	4.6	2.8	706

Table 1: Head of the Dataset cubic_zirconia

- The dataset contains 26967 observations on 11 variables.
- The first column is “Unnamed: 0”, which is just a label and will not be used in the analysis. Hence, we will drop it.
- The third, fourth and fifth column is “cut”,"color" and "clarity" which are the only categorical variable present in the data.

The five number summary of each of the quantitative variables is presented below.

	count	mean	std	min	25%	50%	75%	max
carat	26967.0	0.798375	0.477745	0.2	0.40	0.70	1.05	4.50
depth	26270.0	61.745147	1.412860	50.8	61.00	61.80	62.50	73.60
table	26967.0	57.456080	2.232068	49.0	56.00	57.00	59.00	79.00
x	26967.0	5.729854	1.128516	0.0	4.71	5.69	6.55	10.23
y	26967.0	5.733569	1.166058	0.0	4.71	5.71	6.54	58.90
z	26967.0	3.538057	0.720624	0.0	2.90	3.52	4.04	31.80
price	26967.0	3939.518115	4024.864666	326.0	945.00	2375.00	5360.00	18818.00

Table2: Summary Statistics of the Dataset.

- The above output is very convenient for quickly checking for strange values in the numerical variables.
- The are 26967 records in the dataset.
- The maximum value is carat seems to be little high, considering the 75-percentile value i.e. 1.05, 25-percentile i.e. 0.40 and Standard deviation 0.47, the value seems to lie outside the upper bound of the outliers. At this moment it difficult to say anything about this heavy value of carat so I will keep a note of it.

- Depth and table are percentage values and should range from 0 to 100. In this case, the data points looks OK.

- Looking at 'price' which is our target variable, we observe that the cheapest zircon stone is of worth 326. The mean price of the stones are worth 3939.52 and the most expensive zircon stone is of worth 18818. Let's quickly check, in terms of standard deviation, how far is this value from 75-percentile.: (18818-5360)/4024.86=3.34standard deviation.
- So, although the data shows it's quite expensive, given the high variability observed in the price, we would not consider the maximum as an outlier.

- Looking at variable x, y, z the first we notice that its minimum value is showing 'zero'. From what this variable represents, we know that 'zero' values are not possible. This indicates that there is some data entry error.

- Mean value of variable x and y seems same. Std and median are also quite close to each other. In variable z also the mean and median are also very close.
- Seems variable x,y,z are following normal distribution.

Let's check the values of variable x that are equal to 'zero':

```
#Lets check the value in the column x which are equal to zero
zircon.loc[zircon['x']==0]
```

	carat	cut	color	clarity	depth	table	x	y	z	price
5821	0.71	Good	F	SI2	64.1	60.0	0.0	0.0	0.0	2130
6215	0.71	Good	F	SI2	64.1	60.0	0.0	0.0	0.0	2130
17506	1.14	Fair	G	VS1	57.5	67.0	0.0	0.0	0.0	6381

As we see above some of the values of zero in **x** also have zero in other dimensions. We will consider them as missing values since, in this problem zero is not an admissible value. There are many techniques to treat the missing value in a dataset. Here, we will go ahead and drop them from the dataset.

Of course, we are losing data, but our dataset is of 27000 data points so losing 3 records is not a big deal.

```
zircon=zircon.drop(zircon[zircon["x"]==0].index)
```

```
zircon.loc[zircon['x']==0].shape
```

```
(0, 10)
```

Next let's check variable 'y' for the 'zero' values.

```
zircon.loc[zircon['y']==0].shape
```

```
(0, 10)
```

Now, let us check our 'z' variable for 'zero' values

```
zircon.loc[zircon['z']==0]
```

	carat	cut	color	clarity	depth	table	x	y	z	price
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
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

Above output shows, 6 records with 'zero' values. As we did in 'x' variable, we will go ahead and drop these records too.

```
zircon=zircon.drop(zircon[zircon["z"]==0].index)
```

```
zircon.loc[zircon['z']==0].shape
```

```
(0, 10)
```

```
zircon.describe().T
```

	count	mean	std	min	25%	50%	75%	max
carat	26958.0	0.798190	0.477602	0.20	0.4000	0.70	1.05	4.50
depth	26261.0	61.745345	1.412395	50.80	61.0000	61.80	62.50	73.60
table	26958.0	57.455342	2.231227	49.00	56.0000	57.00	59.00	79.00
x	26958.0	5.730105	1.126714	3.73	4.7100	5.69	6.55	10.23
y	26958.0	5.733832	1.164342	3.71	4.7125	5.70	6.54	58.90
z	26958.0	3.539238	0.717838	1.07	2.9000	3.52	4.04	31.80
price	26958.0	3938.311262	4023.359737	326.00	945.0000	2375.00	5358.00	18818.00

Hence, we see that we have lost 9 records from the dataset.

If we see our data description properly, we can see that 'y' and 'z' variables are having extreme max values. As per information provided on zircon diamonds the maximum dimension which can be found is of 3cm i.e. 30mm. So, if we see the data we can say that those are errors in measurements.

```
zircon.loc[(zircon['y']>30)|(zircon['z']>30)]
```

	carat	cut	color	clarity	depth	table	x	y	z	price
344	0.51	Very Good	E	VS1	NaN	54.7	5.12	5.15	31.80	1970
25795	2.00	Premium	H	SI2	58.9	57.0	8.09	58.90	8.06	12210

Now, let's remove these two data points from our dataset by negating the condition we used to find them.

```
zircon=zircon.loc[~((zircon['y']>30)|(zircon['z']>30))]
zircon.shape
```

```
(26956, 10)
```

Next, let's check for Null Values in the dataset.

```
zircon.isnull().sum()
```

```
carat        0
cut          0
color        0
clarity      0
depth      696
table        0
x            0
y            0
z            0
price        0
dtype: int64
```

The output shows 696 null values in 'depth' variable.

We know that the logistic regression model does not work well with Null values as we have NaN values in the dataset.

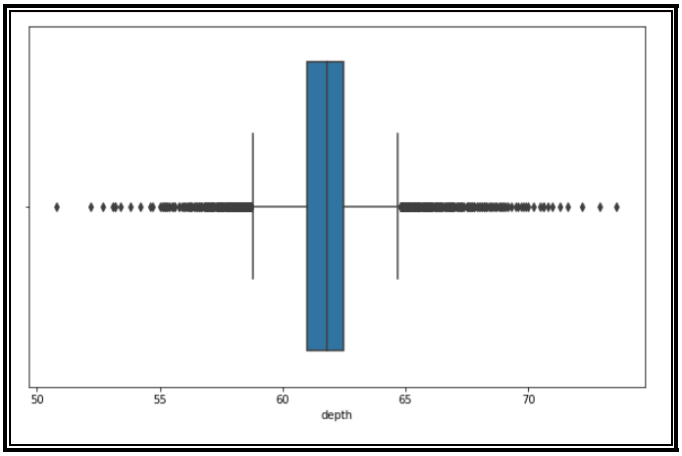
Only some of the machine learning algorithms can work with missing data like KNN, which will ignore the values with Nan values.

There are different ways of treating the Null Values.

In this problem, we will fit the missing values with certain numbers.

The possible way to do so, is by filling the missing data with mean or median values, if its a numerical variable.

Let us check the boxplot of the variable 'depth'

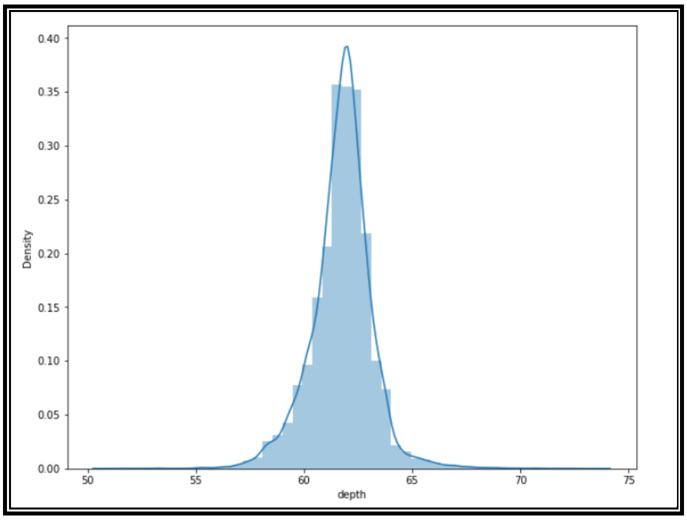


We see that there are several or large numbers of data points that are acting as outliers.

Outlier's data points will have a significant impact on the mean and hence, in such cases, it is not recommended to use the mean for replacing the missing values.

Using mean values for replacing missing values may not create a great model and hence gets ruled out.

Thus, we will use median value to replace the missing values.



We can also observe a similar pattern from the plotting distribution plot. One can observe that there are several high-value of depth in the data points. The data looks to be left-skewed (long tail in the left).

```
zircon.depth.skew()
-0.027495258224894285
```

Skewness essentially measures the symmetry of the distribution.

From the above distplot and skew(), we see that out data is Negative skewed or left-skewed.
In negatively skewed, the mean of the data is less than the median(can be verified from describe() output).
Negatively Skewed Distribution is a type of distribution where the mean, median, and mode of the distribution are negative rather than positive or zero.
Median is the middle value, and mode is the highest value, and due to unbalanced distribution median will be higher than the mean.

```
zircon.depth.kurt()
3.6809900859402256
```

Kurtosis refers to the degree of presence of outliers in the distribution.
Kurtosis is a statistical measure, whether the data is heavy-tailed or light-tailed in a normal distribution.
Excess kurtosis can be positive and is also called Leptokurtic distribution.
Our output shows 'Leptokurtic' (kurtosis > 3)
Leptokurtic is having very long and skinny tails, which means there are more chances of outliers(Boxplot shows the presence of outliers).
Positive values of kurtosis indicate that distribution is peaked and possesses thick tails.
An extreme positive kurtosis indicates a distribution where more of the numbers are located in the tails of the distribution instead of around the mean.

Hence, we will go ahead and treat the missing value with median.

```
zircon['depth']=zircon['depth'].fillna(zircon['depth'].median())
zircon.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 26956 entries, 0 to 26966
Data columns (total 10 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   carat    26956 non-null  float64
 1   cut      26956 non-null  object 
 2   color    26956 non-null  object 
 3   clarity  26956 non-null  object 
 4   depth    26956 non-null  float64
 5   table    26956 non-null  float64
 6   x        26956 non-null  float64
 7   y        26956 non-null  float64
 8   z        26956 non-null  float64
 9   price    26956 non-null  int64  
dtypes: float64(6), int64(1), object(3)
memory usage: 2.3+ MB
```

Above output shows no missing value. Next step is to check the Duplicate records in the dataset.

In [60]:

```
# checking for duplicate
dups = zircon.duplicated()
print('Number of duplicate rows = %d'% (dups.sum()))
zircon[dups]
```

Number of duplicate rows = 33

Out[60]:

	carat	cut	color	clarity	depth	table	x	y	z	price
4756	0.35	Premium	J	VS1	62.4	58.0	5.67	5.64	3.53	949
8144	0.33	Ideal	G	VS1	62.1	55.0	4.46	4.43	2.76	854
8919	1.52	Good	E	I1	57.3	58.0	7.53	7.42	4.28	3105
9818	0.35	Ideal	F	VS2	61.4	54.0	4.58	4.54	2.80	906
10473	0.79	Ideal	G	SI1	62.3	57.0	5.90	5.85	3.66	2898
10500	1.00	Premium	F	VVS2	60.6	54.0	6.56	6.52	3.96	8924
12894	1.21	Premium	D	SI2	62.5	57.0	6.79	6.71	4.22	6505
13547	0.43	Ideal	G	VS1	61.9	55.0	4.84	4.86	3.00	943
13783	0.79	Ideal	G	SI1	62.3	57.0	5.90	5.85	3.66	2898
14389	0.60	Premium	D	SI2	62.0	57.0	5.43	5.35	3.34	1196
14410	1.00	Very Good	D	SI1	63.1	56.0	6.34	6.30	3.99	5645
15798	0.90	Very Good	I	VS2	58.4	62.0	6.29	6.35	3.69	3334
16852	0.79	Ideal	G	SI1	62.3	57.0	5.90	5.85	3.66	2898
17263	1.04	Premium	I	SI2	62.0	57.0	6.53	6.47	4.03	3774
18025	1.51	Good	I	SI1	63.8	57.0	7.21	7.18	4.59	6046
18777	0.32	Premium	H	VS2	60.6	58.0	4.47	4.44	2.70	648
18837	1.01	Premium	H	VS1	61.2	61.0	6.44	6.41	3.93	5294
19731	0.30	Good	J	VS1	63.4	57.0	4.23	4.26	2.69	394
19877	2.01	Premium	I	VS2	60.3	62.0	8.13	8.08	4.89	15939
20301	0.30	Ideal	H	SI1	62.2	57.0	4.26	4.29	2.66	450
20760	1.80	Ideal	H	VS1	62.3	56.0	7.79	7.76	4.84	15105
22322	2.05	Premium	I	SI2	62.0	58.0	8.13	8.08	5.02	9850
22488	2.42	Premium	J	VS2	61.3	59.0	8.61	8.58	5.27	17168
22583	0.33	Ideal	F	IF	61.2	56.0	4.47	4.49	2.74	1240
23458	2.66	Good	H	SI2	63.8	57.0	8.71	8.65	5.54	16239
23564	1.50	Premium	F	SI2	58.5	60.0	7.52	7.48	4.39	7644
24351	2.50	Fair	H	SI2	64.9	58.0	8.46	8.43	5.48	13278
24816	1.50	Good	G	SI2	57.5	63.0	7.53	7.49	4.32	6006
25268	1.20	Premium	I	VS2	62.6	58.0	6.77	6.72	4.22	5699
25759	0.30	Ideal	G	IF	62.1	55.0	4.32	4.35	2.69	863
25941	0.51	Premium	F	SI2	58.1	59.0	5.26	5.24	3.05	1052
26191	2.54	Very Good	H	SI2	63.5	56.0	8.68	8.65	5.50	16353
26530	0.41	Ideal	G	IF	61.7	56.0	4.77	4.80	2.95	1367

Observation:

The dataset shows 33 duplicate rows.

Duplicate cases increase the sample used in statistical inference, reduce the variance, and thus they may artificially increase statistical power of estimation methods. This may lead to more significant coefficients, thus affecting the conclusions.

In this problem there is no unique identifier to confirm if these duplicate values are of no use, but after checking the data I see the value of variable price, carat, cut, color, clarity, depth, table, x,y,z are same. Hence, I will go ahead and drop them.

```
zircon.drop_duplicates(subset=None, keep="first", inplace=True)
```

```
zircon.shape
```

```
(26923, 10)
```

Therefore,

```
zircon.describe(include='all').T
```

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
carat	26923.0	NaN	NaN	NaN	0.797787	0.477043	0.2	0.4	0.7	1.05	4.5
cut	26923	5	Ideal	10805	NaN	NaN	NaN	NaN	NaN	NaN	NaN
color	26923	7	G	5650	NaN	NaN	NaN	NaN	NaN	NaN	NaN
clarity	26923	8	SI1	6564	NaN	NaN	NaN	NaN	NaN	NaN	NaN
depth	26923.0	NaN	NaN	NaN	61.747086	1.3934	50.8	61.1	61.8	62.5	73.6
table	26923.0	NaN	NaN	NaN	57.455425	2.231345	49.0	56.0	57.0	59.0	79.0
x	26923.0	NaN	NaN	NaN	5.72932	1.126025	3.73	4.71	5.69	6.55	10.23
y	26923.0	NaN	NaN	NaN	5.731199	1.11784	3.71	4.71	5.7	6.54	10.16
z	26923.0	NaN	NaN	NaN	3.537603	0.695983	1.07	2.9	3.52	4.04	6.72
price	26923.0	NaN	NaN	NaN	3936.015711	4020.798496	326.0	945.0	2373.0	5352.5	18818.0

Table 3: Summary statistics of Dataset after treating Null Duplicate and other anomalies

Data Visualization

Univariate Analysis- "Uni" +"Variate" **Univariate,** means one variable/feature analysis. The **univariate** analysis basically tells us how data in each feature is distributed

Bivariate Analysis: "Bi" +"Variate" **Bi-variate,** means two variables or features are analysed together, that how they are related to each other. Generally, we use to perform to find the relationship between the dependent and independent variable.

Multi-Variate Analysis: means more than two variables or features are analysed together. that how they are related to each other.

For Categorical Variable/Features:

CUT:

```
Cut: 5
Fair           779
Good          2434
Very Good     6026
Premium       6879
Ideal        10805
Name: cut, dtype: int64
```

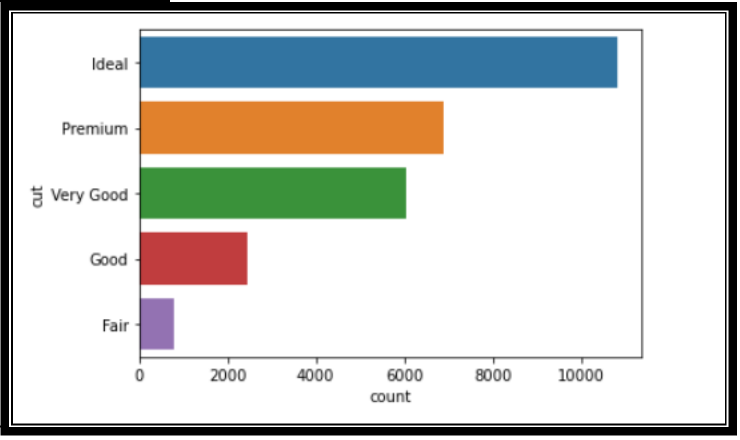


Figure 1.1: Countplot of Categorical Variable 'cut'

The above countplot is showing the display of occurrences or frequency of 'cut' categorical variable data using bar. As per the plot, no. of Ideal cut zircon diamonds are more in number than premium, then very good, then good and then fair.

Color:

```
Color: 7
J    1440
I    2765
D    3341
H    4090
F    4722
E    4915
G    5650
Name: color, dtype: int64
```

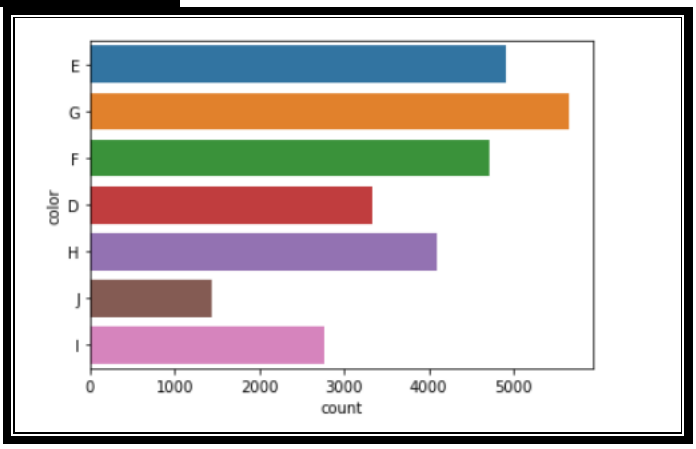


Figure 1.2: Countplot of Categorical Variable 'color'

The above countplot shows the frequency of categorical variable 'color' using bars. The plot shows color G of zircon diamond is most in number followed by E & F. With D being the best and J being the worst.

Clarity:

```
Clarity: 8
I1        362
IF        891
VVS1     1839
VVS2     2530
VS1      4085
SI2      4560
VS2      6092
SI1      6564
Name: clarity, dtype: int64
```

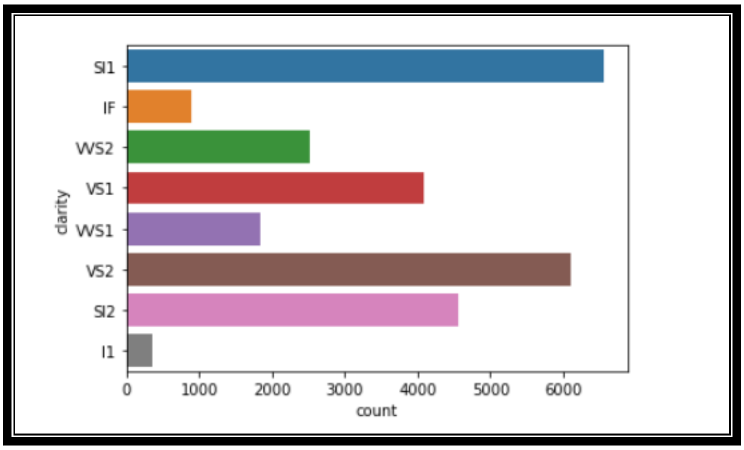


Figure 1.3: Countplot of Categorical Variable 'clarity'

The above countplot shows the frequency distribution of categorical variable 'clarity' in bars. The plot shows that Clarity of zircon diamond corresponding to SI1 is most in numbers, followed by VS2 and SI2.

Plotting the distribution of Price across the classes of categorical features.

<u>Price distribution of 'cut' variable</u>

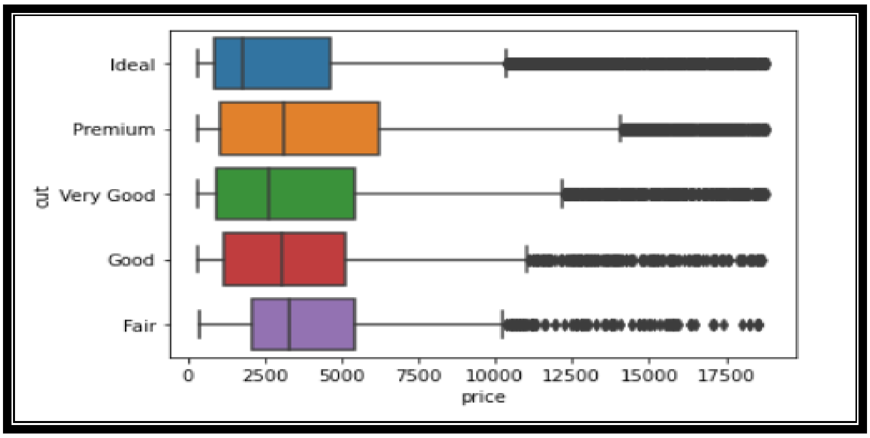


Figure 1.4: Boxplot of 'price' with 'cut'

- The boxplot shows price distribution across five type of zircon diamond cut.
- It shows that most sold is ideal type of cut and least sold is fair type of cut.
- All the five types of 'cut' is showing outliers.

Price distribution of 'color' variable

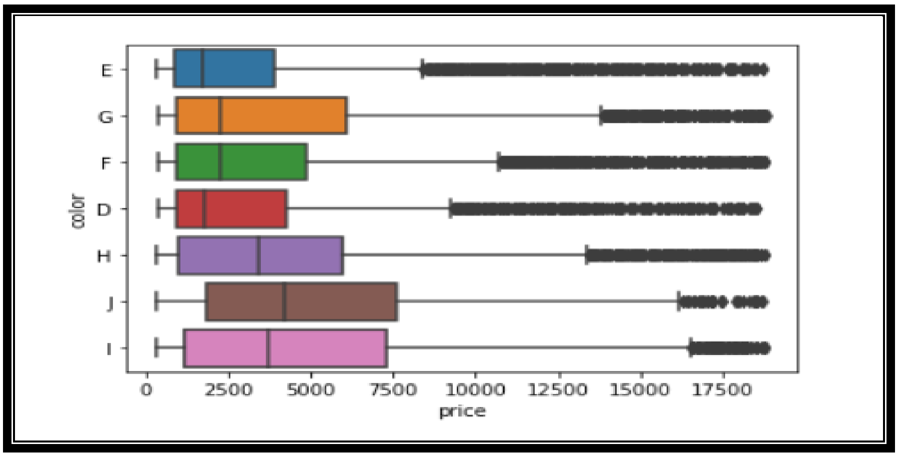


Figure 1.5: Boxplot of 'price' with 'color'

- For color the most sold is G color zircon diamond and least sold is J color zircon diamond.
- All the color type of zircon are having ouliers.

Price distribution of 'clarity' variable

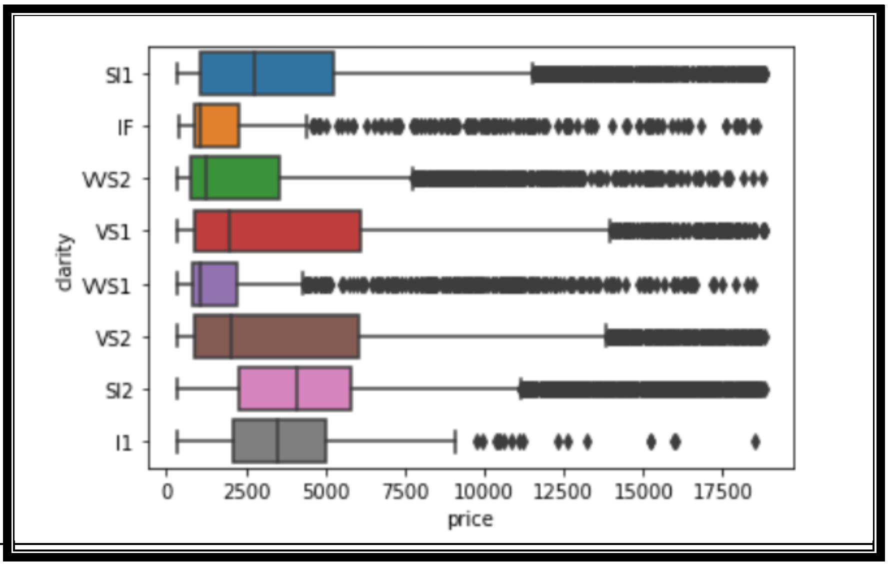


Figure 1.6: Boxplot of 'price' with 'clarity'

- The plot shows that SI1 clarity type zircon are most sold than other.
- The least sold type of clarity zircon is I1.
- This plot also shows outliers.

Let's see how all the other numeric features, not just Price, change with each categorical feature by summarizing the numeric features across the classes. We use the Dataframe's groupby function to group the data by a category and calculate a metric (such as mean, median, min, std, etc) across the various numeric features.

cut	carat	depth	table	x	y	z	price
Fair	1.061900	63.886521	59.290629	6.292311	6.224159	3.998139	4565.768935
Good	0.849010	62.359326	58.703328	5.843726	5.858439	3.646175	3927.074774
Ideal	0.701430	61.707728	55.956205	5.500229	5.511296	3.396558	3454.820639
Premium	0.887735	61.282863	58.714738	5.964956	5.939231	3.645260	4540.186192
Very Good	0.813182	61.823328	57.963929	5.752359	5.781583	3.569637	4032.267961

color	carat	depth	table	x	y	z	price
D	0.658515	61.705747	57.374828	5.414385	5.419129	3.341152	3184.827597
E	0.656019	61.661168	57.516843	5.403961	5.409329	3.338973	3073.940399
F	0.731144	61.678950	57.438776	5.599748	5.603681	3.453973	3700.277001
G	0.770335	61.746673	57.302301	5.678966	5.680949	3.506989	4004.967434
H	0.909543	61.828624	57.483916	5.977773	5.985243	3.694962	4469.778049
I	1.033515	61.866727	57.565533	6.236796	6.236604	3.855732	5124.816637
J	1.161653	61.898056	57.793542	6.514146	6.513729	4.030708	5329.706250

clarity	carat	depth	table	x	y	z	price
I1	1.279309	62.626519	58.373481	6.758536	6.708785	4.217486	3915.013812
IF	0.495443	61.506397	56.449270	4.943962	4.965230	3.045567	2739.534231
SI1	0.849395	61.853093	57.636898	5.884581	5.884721	3.638021	3996.614564
SI2	1.082195	61.775729	57.910634	6.412804	6.414096	3.958305	5088.169919
VS1	0.726542	61.665516	57.319652	5.568490	5.573554	3.441605	3838.130201
VS2	0.767733	61.722029	57.430532	5.664782	5.665993	3.495957	3963.159225
VVS1	0.499929	61.628929	56.910984	4.946900	4.962501	3.053861	2502.874388
VVS2	0.593047	61.656206	57.060632	5.208213	5.222810	3.214542	3263.042688

Interpretation of above output:

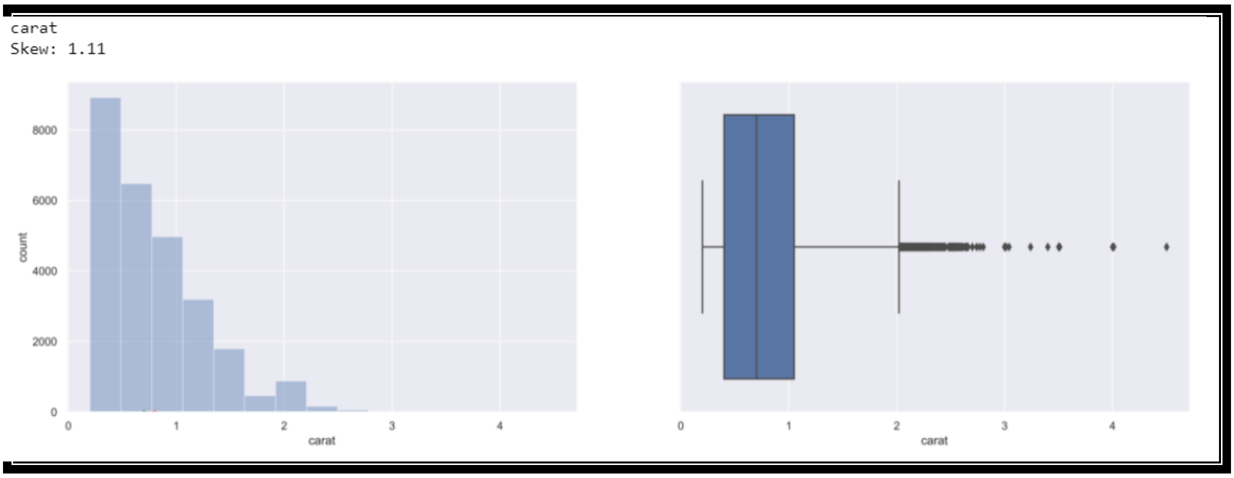
As we check the first object column ‘cut’ with relation to price, we see that mean price of cut type is increasing from ideal then good then very good then premium and then fair. This pretty much shows an order ranking of the ‘cut’ type (does not match the order provided in data dictionary). Since its specification mention in Project FAQ to follow the order ranking provided in data dictionary, we will encode them with ordinal encoder as this seems to be ordinal variable. (Refer to question no. 1.2 for the encoding of the ‘cut’ variable).

The ‘color’ variable seems to have a good impact of the price variable. We are sure it will be used further in linear regression model.

From the category ‘clarity’ it is difficult to see any kind of order and the variable seems to be having a direct impact on the price variable.

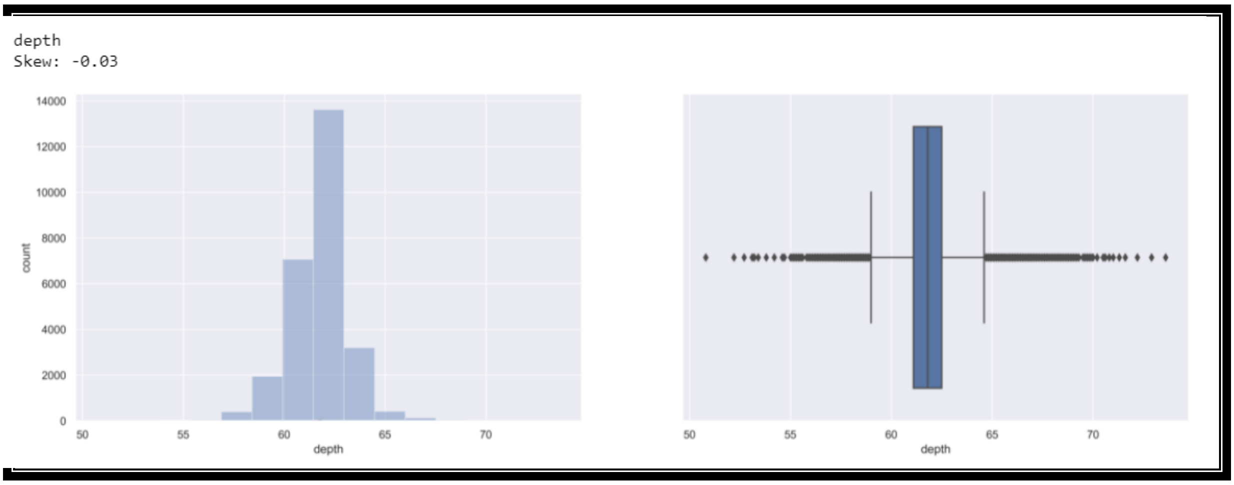
Further, handling of Categorical variable will be done in Problem no. 1.2 for model building.

Relationships between numeric features and other numeric features



carat
Skew: 1.11

Figure 1.7: Distplot and Boxplot of variable 'carat'



depth
Skew: -0.03

Figure 1.8: Distplot and Boxplot of variable 'depth'

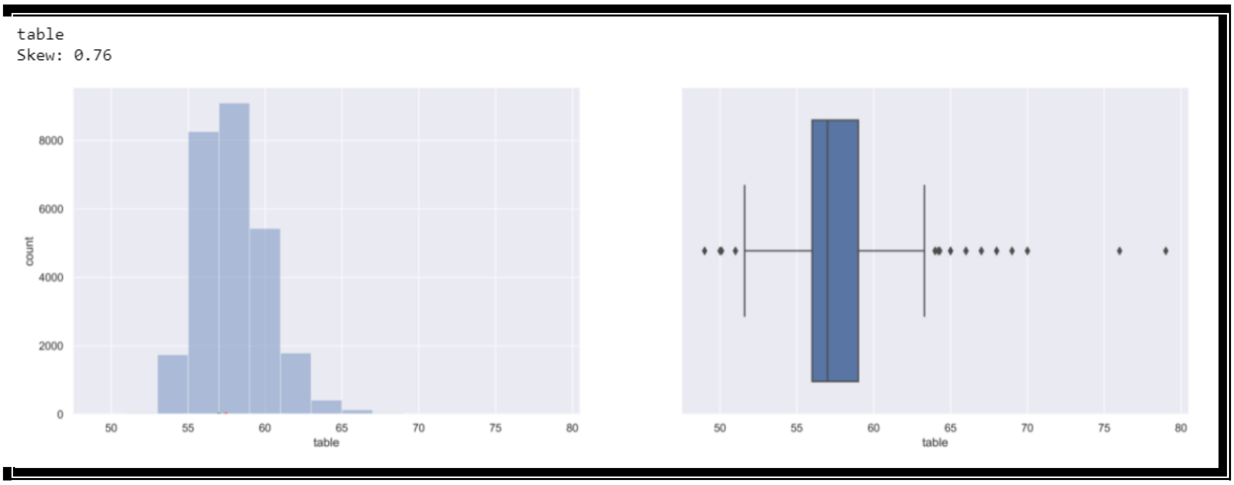
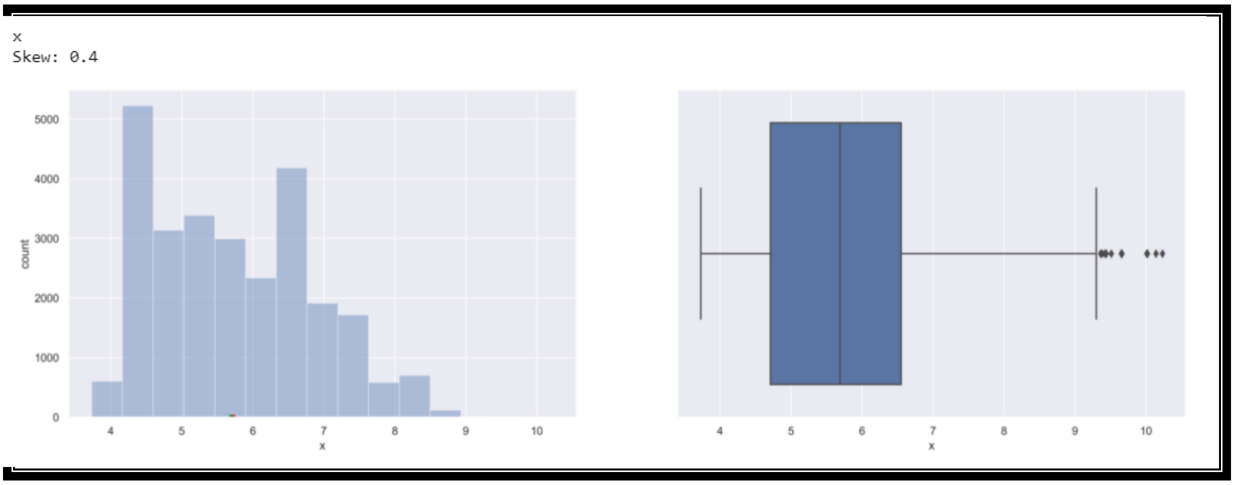


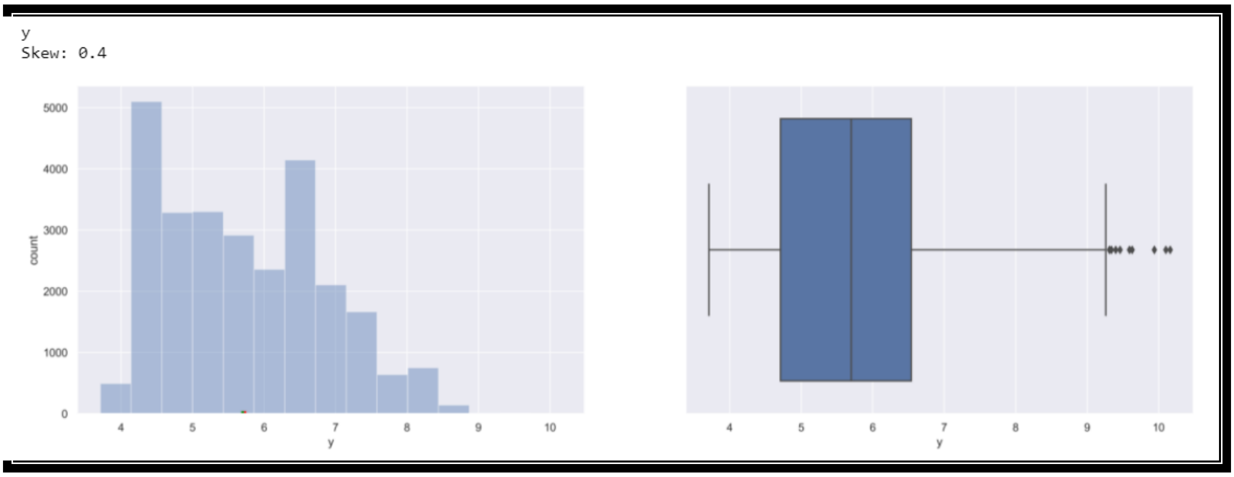
table
Skew: 0.76

Figure 1.9: Distplot and Boxplot of variable 'table'



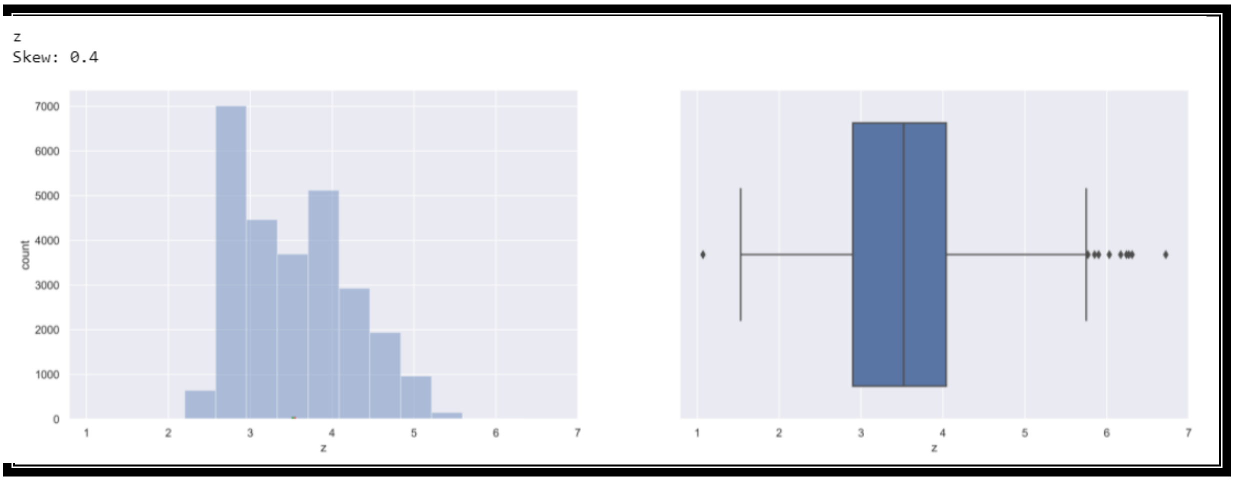
x
Skew: 0.4

Figure 1.10: Distplot and Boxplot of variable 'x'



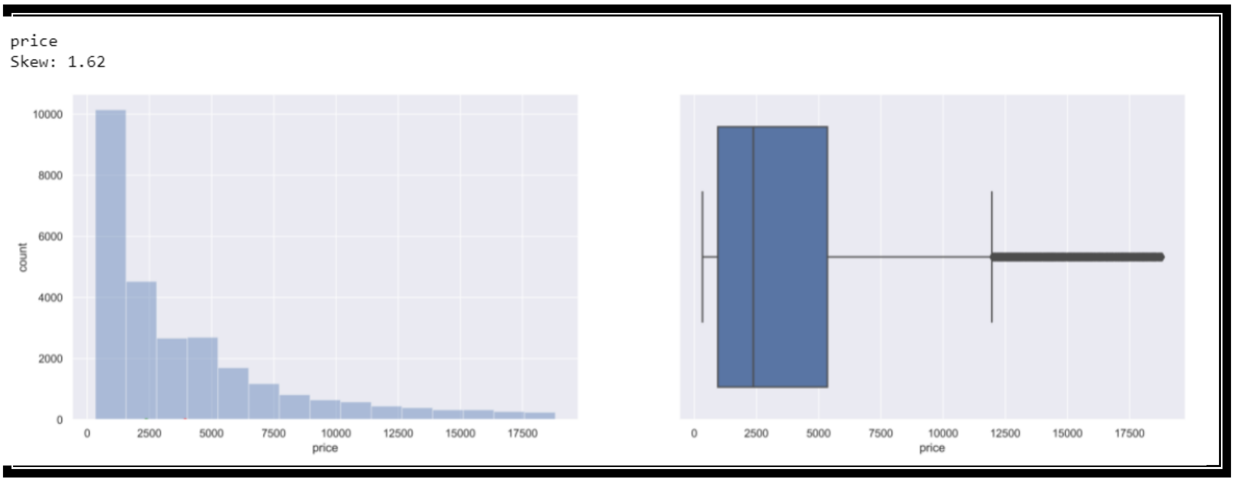
y
Skew: 3.89

Figure 1.11: Distplot and Boxplot of variable ‘y’



z
Skew: 2.64

Figure 1.12: Distplot and Boxplot of variable ‘z’



price

Skew: 1.62

Figure 1.13: Distplot and Boxplot of variable ‘price’

<u>**Observation:**</u>

- From above distplot and boxplot we see that variable ‘depth’ shows normal distribution.
- Carat, table, x, y, and z are all right-skewed.
- All the predictors /independent variables are showing good amount of outliers.
- Even the target variable ‘price’ is right skewed and is showing significant number of outliers.

<u>**Outliers:**</u> Outliers are the data points possibly different from the rest. They represent errors in measurements, bad data collection, or simply shows variable not considered when collecting the data. In Regression Model, it is very important to do outlier treatment. But after going thoroughly through this Zircon diamond data we feel otherwise. We are not going to treat them due to following reasons:

The outlier data points in carat which are higher than showing as outliers are just 655 data points which is around 2% of outliers present in the dataset. Which we feel is very low. And might not be impactful to the data while performing regression. We know that ‘carat’ is very significant variable when it comes to zircon diamonds, hence we will keep them.

Depth and Table are the average diameters of the zircon stones and can range from 0-100. If we look closely at the dataset we see that values of depth and table are very much in the range and hence not going to treat them either.

Variables x(length), y(width), and z(height) are the dimension of the zircon diamonds. And we know that these are directly correlated to carat of the diamonds. Looking at the data we feel that diamonds can have higher dimensions are non of the dimension seems to be out side of the box. The values which were higher has already been treated above. Hence, further no outlier treatment is required.

<u>Since this is a Regression Problem let us see the Scatter Plot of predictors with the target variables to check the Correlation</u>

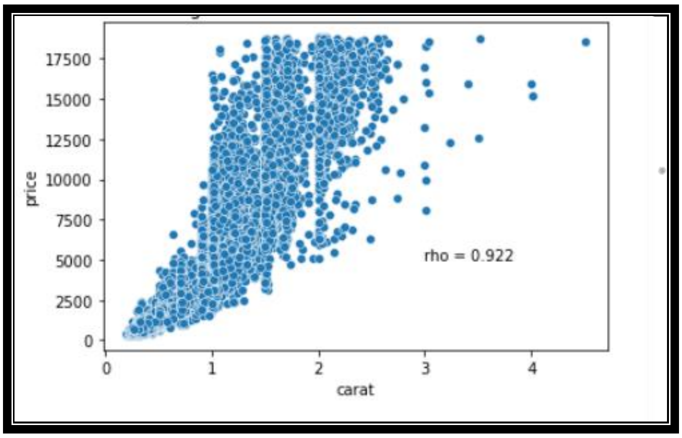


Figure 1.14: Correlation between price and carat

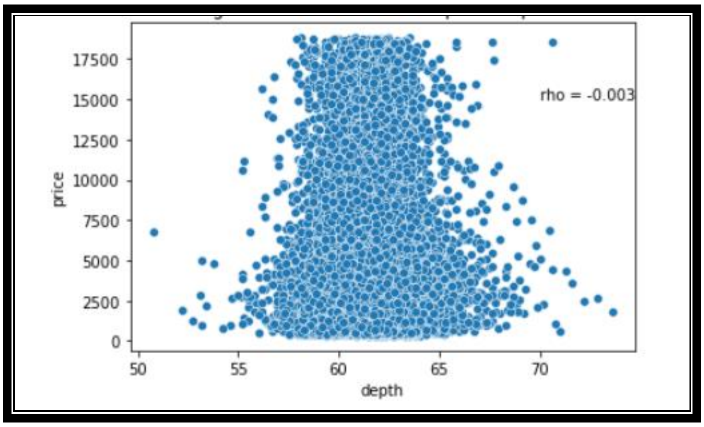


Figure 1.15: Correlation between price and depth

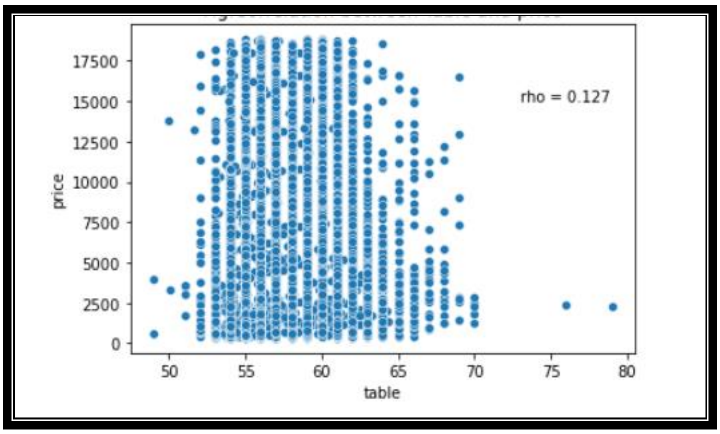


Figure 1.16: Correlation between price and table

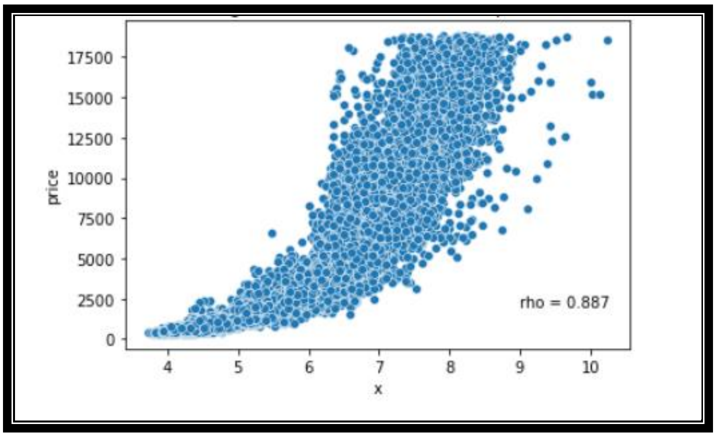


Figure 1.17: Correlation between price and variable x

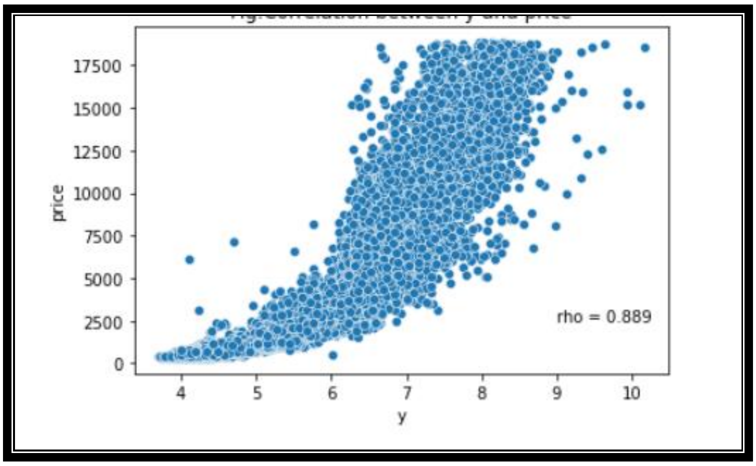


Figure 1.18: Correlation between price and variable y

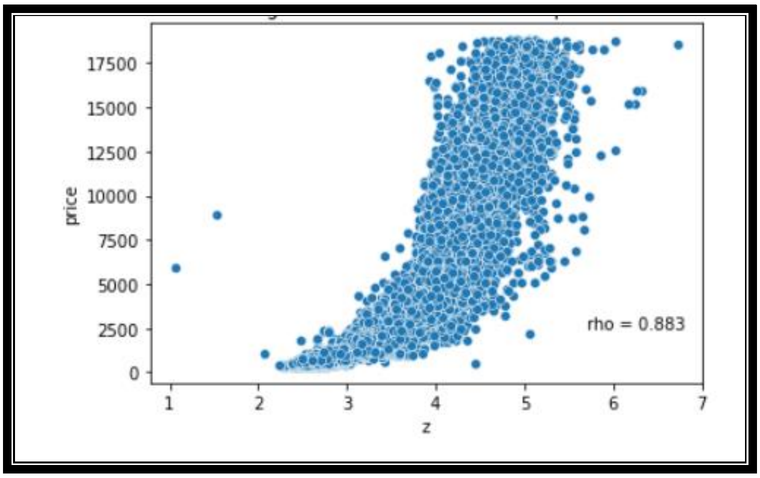


Figure 1.19: Correlation between price and variable z

Interpretation of Fig 1.14 to 1.19:

Fig. 1.14: Correlation between price (Y)and X_1(carat) is r=0.922. This indicates positive dependence between price and carat and as the carat increases, price (%) increases.

Fig 1.15: Correlation between price (Y)and X_1(depth) is r=-0.003. This indicates negligible dependence between price and depth.

Fig 1.16: Correlation between price (Y)and X_1(table) is r=0.127. This indicates a positive correlation between price and table but the numerical value is very small.

Fig 1.17: Correlation between price (Y)and X_1(x) is r=0.887. This indicates positive dependence between price and x variable and as the x dimension increases, price (%) increases.

Fig 1.18: Correlation between price (Y)and X_1(y) is r=0.889. This indicates positive dependence between price and y variable and as the y dimension increases, price (%) increases.

Fig 1.19: Correlation between price (Y)and X_1(z) is r=0.883. This indicates positive dependence between price and z variable and as the z dimension increases, price (%) increases.

Heatmap of the correlations

	carat	depth	table	x	y	z	price
carat	1.000000	0.035301	0.181530	0.977906	0.976836	0.976481	0.922388
depth	0.035301	1.000000	-0.293401	-0.018027	-0.021584	0.100582	-0.002526
table	0.181530	-0.293401	1.000000	0.197531	0.191445	0.157622	0.126975
x	0.977906	-0.018027	0.197531	1.000000	0.998512	0.991130	0.887448
y	0.976836	-0.021584	0.191445	0.998512	1.000000	0.990739	0.888980
z	0.976481	0.100582	0.157622	0.991130	0.990739	1.000000	0.882559
price	0.922388	-0.002526	0.126975	0.887448	0.888980	0.882559	1.000000

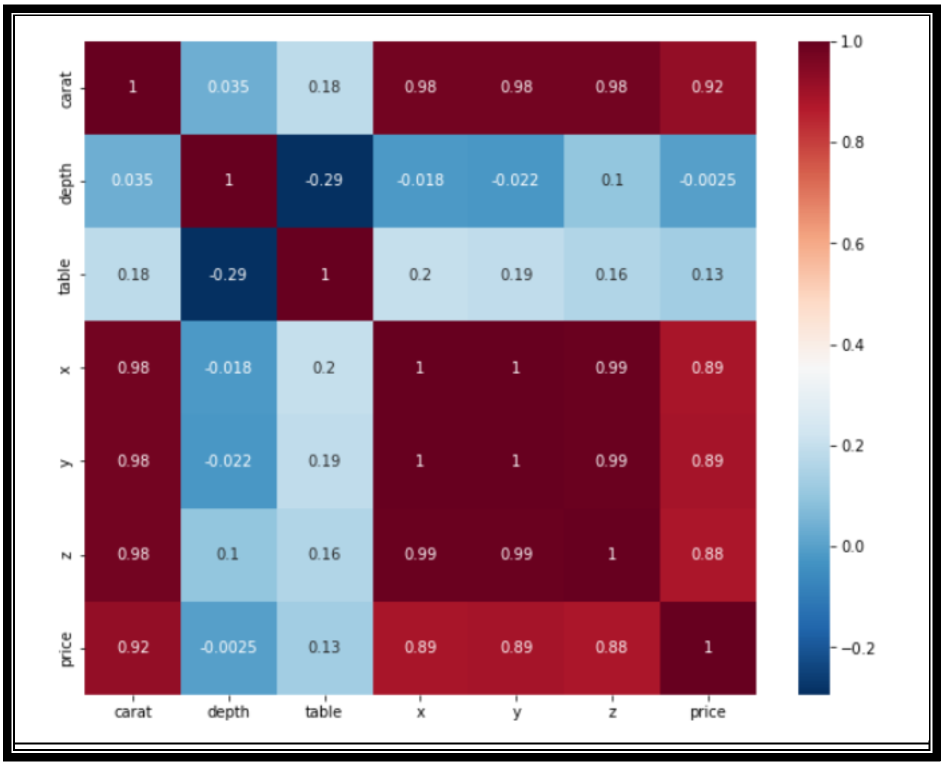


Figure 1.20: HeatMap-Correlation Matrix

Positive Correlation

- Price – carat, x, y, z.
- z – carat, depth,x,y,price.
- y – carat, depth,x,z,price
- x-carat, depth,y,z,price
- depth-z
- carat-x, y, z, price

Negative Correlation

are not very high

Strong Positive correlations between carat,x,y, and z.

This violates the non-multicollinearity assumption of Linear regression.

Multicollinearity hinders the performance and accuracy of our regression model.

To avoid this, we must get rid of some of these variables by doing feature selection.

Note: Correlation is only useful in determining linear relationship between two variables. Zero correlation may imply no linear dependence, but that does not preclude any other form of dependence (such as, polynomial) between the variables concerned. Further, correlation does not indicate any cause and effect relationship. Correlation simply quantifies how well two variables are related, if at all, and its direction (i.e. positive or negative)

PairPlot- Checking pairwise distribution of the continuous variables

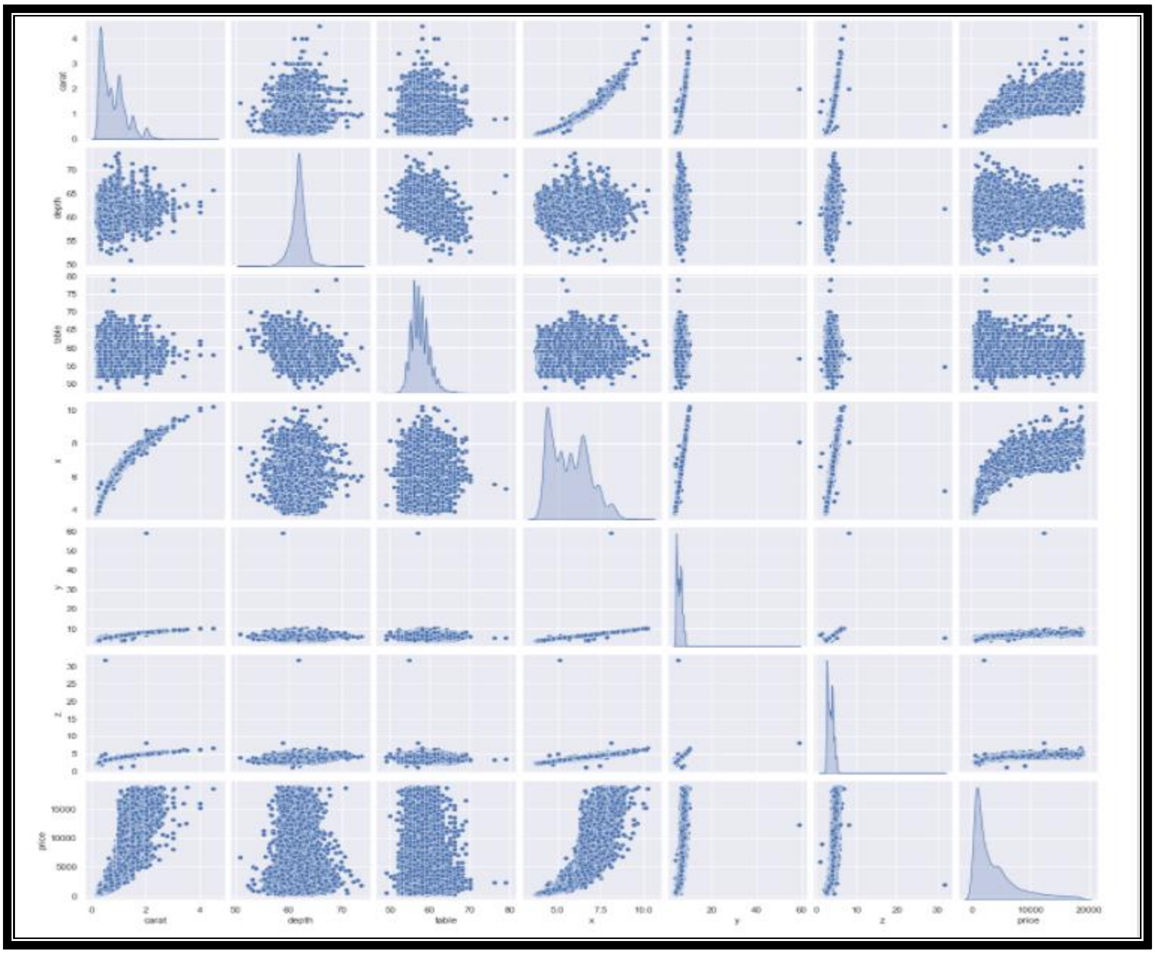


Figure1.21 : Pairplot of All Variable

EDA Conclusion:

We have now a dataset of 26923 rows after treating duplicates, 'zero' values of variable x, y, and z. Also, treating the maximum dimensions in y and z variable(reason mentioned above).

One more important thing is 'scaling', In regression, it is often recommended to scale the features so that the predictors have a mean of 0. This **makes it easier to interpret the intercept term as the expected value of Y when the predictor values are set to their means**.

Another reason for scaling is when one predictor variable has very large scale. In that case, the regression coefficients may be on a very small order of magnitude which can be unclear to interpret. The reason that we standardize predictions primarily exists so that the units of the regression coefficients are the same. Scaling also helps to standardize the independent features present in the data in a fixed range. If feature scaling is not done, then a machine learning algorithm tends to weigh greater values, higher and consider the smaller values as the lower values, regardless of the unit of the values. To supress the effect, we need to bring all features to the same level of magnitude.

However, for this data, in our descriptive summary output we see that mean and std values aren't varying significantly for original numerical variables. Hence, when we scale the numbers, our model performance will not vary much. (further analysis on scaling is done in 1.2 and 1.3)

From our Correlation matrix we can identify that there is a strong correlation between independent variables i.e. carat, x y, and z. All these variables are strongly correlated with the target variable 'price'. This indicate that the data is suffering from case of Multicollinearity. Depth does not show any strong relation with price variable. Table also does not show any strong corelation with price variable. Hence, with this initial analysis we can hope to build a regression model without these variables hooping the model will give a good accuracy with less features.

1.2. **Build various iterations of the Linear Regression model using appropriate variable selection techniques for the full data.**

Let's identify the feature with the strongest linear relation with price!

```
zircon.corrwith(zircon.price)

carat     0.922388
depth    -0.002526
table     0.126975
x         0.887448
y         0.888980
z         0.882559
price     1.000000
dtype: float64
```

Let us create a **Simple Linear Regression** model between dependent and independent variable to find out the relationship between these two variables.

Simple linear regression relates the target variable Y to the single predictor variable X through a straight line. The mathematical formulation of the simple linear regression line is:

$$(Y)=\beta 0+\beta 1X$$

where,

Y: is the value of the continuous response (or dependent) variable,

$\beta 0$and $\beta 1$: are intercept and slope coefficients, respectively, and known as the regression parameters.

X: represents the independent (predictor) variable continuous in nature.

The simple linear regression model involves unknown parameters $\beta 0$and $\beta 1$, which need to be estimated from data. There are several different methods of estimating the parameters. The simplest and the most widely used method is known as the Ordinary Least Squares method (OLS)

OLS: Ordinary least squares (OLS) regression is a statistical method of analysis that estimates the relationship between one or more independent variables and a dependent variable; the method estimates the relationship by minimizing the sum of the squares in the difference between the observed and predicted values of the dependent variable.

Note: The objective here is to determine the dependence of 'price' on 'carat'. Though any one of the 6 continuous predictors may have been used as predictor, the choice is made based on a **higher correlation between the response and the predictor**. A scatterplot of price versus carat helps to get a visual impression about whether a linear function of carat will at all be suitable to describe price.

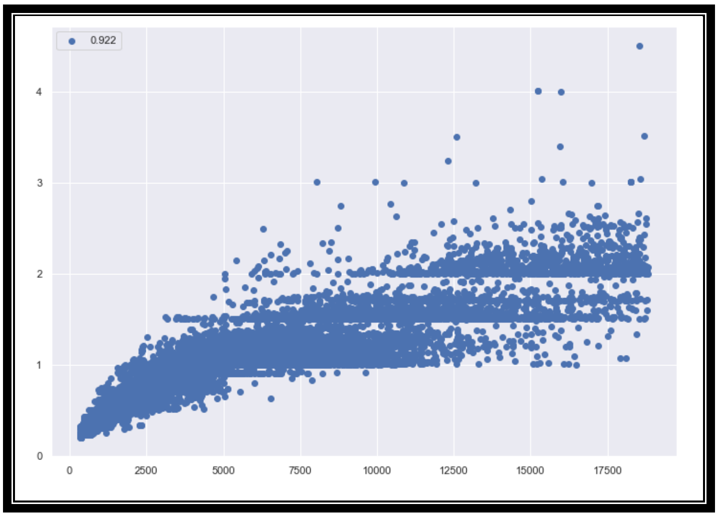


Fig. 1.22: Scatterplot of carat and price in SLR model

The scatterplot above suggests that a linear relationship between price and carat may exist since the majority of the points seem to fall on a straight line. We also expect the slope to be positive and hence, increase in carat is expected to increase the price. Recall that the correlation between price and carat is 0.922.

A linear regression model is fit with price as the target/response and carat as the predictor.

```
# Regression: Price on carat
mod_slr = ols('price ~ carat', data = zircon).fit()
intercept , carat_slope = mod_slr.params
equation = "\n Y = {}".format(round(carat_slope,2))+"*X +"+" {}".format(round(intercept,2))
print(equation)
```

```
Y = 7774.43*X + -2266.32
```

Thus, the OLS line has the form

Price^= 7774.43*Carat-2266.32

The hat symbol is used to indicate that the regression gives an estimate of the response.
We first note that the sign of $\beta1$is positive.
This shows that the two variables are positively related, that is, if one increases, the other increases too.
This confirms our expectation that the variables price and carat increase/decrease in equal directions and we get a straight line with positive slope.
The value of $\beta1$ indicates that if carat increases by 1 unit, the estimated price increases by 7774.43.
The intercept term is the estimated value of the response when the predictor is 0. However, the intercept term is not always interpretable, such as in this case.

Note: The sign of the regression slope and the correlation coefficient will always be the same. The regression slope is the measure of change in the response with one unit change in predictor. The sign of the regression slope indicates the direction of the change.

The following graph shows the OLS regression line(in blue) through the scatterplot.

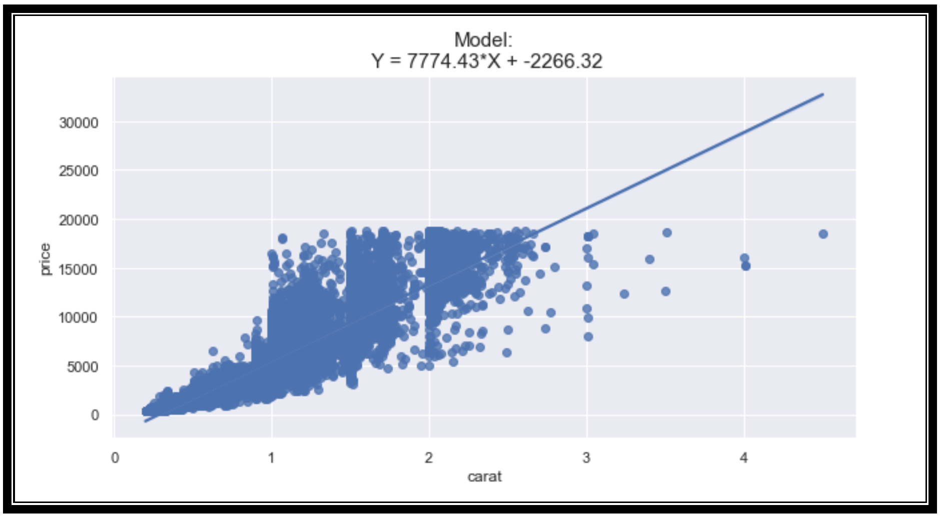


Figure 1.23: Scatterplot showing OLS line between 'price' and 'carat'

Fitting the regression model or simply estimating the regression coefficients is not enough.

We now need to know, is carat at all statistically significant in predicting the price.

Let us consider the test of hypothesis

$H0$: $\beta 1=0$ vs. $H1$: $\beta 1\neq 0$ where $\beta 1$is the regression coefficient of carat when price is regressed on carat.

```
#significance of regression
print(mod_slr.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.851
Model:                            OLS   Adj. R-squared:                  0.851
Method:                 Least Squares   F-statistic:                 1.535e+05
Date:                Sat, 29 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:15:43   Log-Likelihood:            -2.3603e+05
No. Observations:               26923   AIC:                         4.721e+05
Df Residuals:                   26921   BIC:                         4.721e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept  -2266.3232     18.444   -122.876      0.000   -2302.474   -2230.172
carat       7774.4294     19.842    391.810      0.000    7735.537    7813.321
==============================================================================
Omnibus:                     6766.544   Durbin-Watson:                   2.010
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            63448.633
Skew:                           0.941   Prob(JB):                         0.00
Kurtosis:                      10.281   Cond. No.                         3.63
==============================================================================

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

Summary Observation:

The Intercept value is -2266.3232

The Coefficient value of Carat is 7774.4294

R2 value is 85%

Adj R2 value is 85%

P_values of Intercept and carat showing in the summary is 0.000.

What are these statistical values? Lets, see what are these statistical information is all about:

Dependent variable: Dependent variable is one that is going to depend on other variables. In this regression analysis 'price' is our dependent variable because we want to analyse the effect of 'carat' on 'price'.

Model: The method of **Ordinary Least Squares(OLS)** is most widely used model due to its efficiency. This model gives best approximate of true population regression line. The principle of OLS is to minimize the square of errors.

Number of observations: The number of observations is the size of our sample, i.e. N = 26923.

Degree of freedom(df) of residuals:

Degree of freedom is the number of independent observations on the basis of which the sum of squares is calculated.

D.f Residuals = 26923 – (1+1) = 26922

Degree of freedom(D.f) is calculated as,

Degrees of freedom, $D.f = N - K$

Where, N *= sample size(no. of observations) and* K *= number of variables + 1*

Df of model:

Df of model = $K - 1 = 2 - 1 = 1$*,*

Where, K *= number of variables + 1*

Constant term: The constant terms is the intercept of the regression line. The intercept is -2266.3232. In regression we omits some independent variables that do not have much impact on the dependent variable, the intercept tells the average value of these omitted variables and noise present in model.

Coefficient term: The coefficient term tells the change in **'price'** for a unit change in **'carat'** i.e if 'carat' rises by 1 unit then 'price' rises by 7774.43.

Standard error of parameters: Standard error is also called the standard deviation. Standard error shows the sampling variability of these parameters.

t – statistics:

In theory, we assume that error term follows the normal distribution.

t – statistics are calculated by assuming following hypothesis –

- $H_0 : B_2 = 0$ *(variable X has no influence on Y)*
- $H_a : B_2 \neq 0$ *(X has significant impact on Y)*

p – values:
In theory, we read that p-value is the probability of obtaining the t statistics at least as contradictory to H_0 as calculated from assuming that the null hypothesis is true. In the summary table, we can see that P-value for both parameters is equal to 0. This is not exactly 0, but since we have very large t statistics (-122.9 and 391.8) p-value will be approximately 0. So, depending on the significance levels we can see that we can reject the null hypothesis at almost every significance level.

Confidence intervals:
There are many approaches to test the hypothesis, including the p-value approach mentioned above. The confidence interval approach is one of them. 5% is the standard significance level ($\propto$) at which C.I's are made.
While calculating p values we rejected the null hypothesis we can see same in C.I as well. Since 0 does not lie in any of the intervals so we will reject the null hypothesis.

R – squared value:

R^2 is the coefficient of determination that tells us that how much percentage variation independent variable can be explained by independent variable. Here, 85.1 % variation in Y can be explained by X. The maximum possible value of R^2 can be 1, means the larger the R^2 value better the regression.

Adj. R-squared: This is the modified version of R-squared which is adjusted for the number of variables in the regression. It increases/decreases only when we include attributes into the model that are weak or poor predictors of Y.

F – statistic:
F test tells the goodness of fit of a regression. The test is similar to the t-test or other tests we do for the hypothesis.

In the above regression summary, we observe that the p-value corresponding to carat is very small and thus the null hypothesis $\boldsymbol{H0{:}\beta1=0}$ is rejected which in turn indicates that carat is significant in explaining price.

Statistical significance alone, however, is not enough to decide whether the predictor is useful in explaining the variability in the response. Is carat enough to explain a large part of variation in price? This leads us to the concept of coefficient of determination, $R2$.

The coefficient of determination R2 is a summary measure that explains how well the sample regression line fits the data.

We have learnt in regression that not all predicted values of the response will be equal to the observed given value Y. In fact, it may well happen that none of the estimated values of the response coincides with the corresponding observed values. The difference between the observed and the estimated values of the response is called **residual**. **Residual is the estimated value of the unobserved error component in the regression equation.**

Residuals are estimated errors and are defined as $\hat{\epsilon}_i = Y_i - \hat{Y}_i$.
Residuals have many important properties and are employed to check various regression assumptions.

We notice that the $R2$ value in this case is not very low, in fact the model is showing quite a good performance stating carat is a big contributor toward the price factor of zircon diamonds

- around 85% variability in the dependent variable is being explained by the 'carat' variable.
- For Simple Linear Regression, the square of the Pearson's correlation is same as the value of the $R2$. Let us check it now.

```
from scipy.stats import pearsonr

P_S_corr = pearsonr(zircon['price'], zircon['carat'])[0]
P_S_corr

0.9223884059656058
```

Before, we build the Multiple Linear Regression model, let us play around with the data and try different kinds of variable transformation to see whether they improve performance.

Performing Scaling transformation of 'carat' and 'price'

```
scaled_carat = (zircon['carat']-np.mean(zircon['carat']))/np.std(zircon['carat'], ddof=1)
scaled_carat

0       -1.043484
1       -0.980597
2        0.214264
3       -0.791935
4       -1.022522
           ...
26962    0.654475
26963   -0.980597
26964   -0.603273
26965   -1.106372
26966    0.947950
Name: carat, Length: 26923, dtype: float64
```

```
scaled_price = (zircon['price']-np.mean(zircon['price']))/np.std(zircon['price'],ddof=1)
scaled_price

0       -0.854809
1       -0.734186
2        0.585203
3       -0.709813
4       -0.785171
           ...
26962    0.366093
26963   -0.701855
26964   -0.567055
26965   -0.809296
26966    0.305905
Name: price, Length: 26923, dtype: float64
```

```
mod_slr_exp = SM.ols(formula='scaled_price~scaled_carat',data=zircon).fit()
mod_slr_exp.summary()
```

OLS Regression Results

Dep. Variable:	scaled_price	**R-squared:**	0.851
Model:	OLS	**Adj. R-squared:**	0.851
Method:	Least Squares	**F-statistic:**	1.535e+05
Date:	Sun, 30 Jan 2022	**Prob (F-statistic):**	0.00
Time:	00:28:10	**Log-Likelihood:**	-12591.
No. Observations:	26923	**AIC:**	2.519e+04
Df Residuals:	26921	**BIC:**	2.520e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-2.005e-14	0.002	-8.52e-12	1.000	-0.005	0.005
scaled_carat	0.9224	0.002	391.810	0.000	0.918	0.927

Omnibus:	6766.544	**Durbin-Watson:**	2.010
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	63448.633
Skew:	0.941	**Prob(JB):**	0.00
Kurtosis:	10.281	**Cond. No.**	1.00

Figure 1.25: SLR Model Summary of Scaled variable price and carat

Note: Value of R2 and adj R2 remain the same as compared to above SLR summary before scaling.

Hence, We can say that scaling a variable for Linear Regression will give us the same values as compared to the unscaled variables.

Now, next step is then to examine whether inclusion of the other predictors contribute towards explanation of the variability in the response, and if so, to what degree. Let's go ahead and build a **Multiple Linear Regression** Model with all the predictors.

The formal definition of Multiple Linear Regression: Multiple regression is a statistical technique used to analyse relationship between a single dependent variable and several predictors simultaneously.

The mathematical formulation of multiple linear regression line is:

$$(Y)=\beta 0+\beta 1X1+\beta 2X2+\cdots+\beta kXk$$

where,
Y: is the value of the continuous response (or dependent) variable,
$\beta 0$: is the intercept
Xj: represents the jth independent (predictor) variable continuous in nature. j = 1, ..., k
βj: represents the coefficient of the jth independent (predictor) variable.

In case of multiple regression also the regression coefficients are estimated by minimizing the error sum of squares.
Before, running regression model it is important to look at correlations of all variables with respect to each other.

First, let's look at the Categorical Variables in this data. That are 'cut', 'color' and 'clarity'.

- Here our Categorical variables following an ordered ranking(information provided in Data Dictionary). So, here we will treat our Categorical variable by doing Ordinal Encoding.

Why? In ordinal encoding, each unique category value is assigned and integer values.

For e.g. "Fair" is 0, "Good" is 1, "Very Good" is 2 and so on.

This is called and ordinal encoding or an integer encoding and is easily reversible. Often integer values starting at zero are used. The integer values have a natural ordered relationship between each other and machine learning algorithms may be able to understand and harness this relationship.

Below, is the code we have used to assign the integer to labels in the order that is informed to us in the Problem Data Dictionary.

Below is how we encoding our Categorical data.

```
## We are coding up the 'cut','color','clarity' variable in an ordinal manner

zircon['cut']=np.where(zircon['cut'] =='Ideal', '4', zircon['cut'])
zircon['cut']=np.where(zircon['cut'] =='Premium', '3', zircon['cut'])
zircon['cut']=np.where(zircon['cut'] =='Very Good', '2', zircon['cut'])
zircon['cut']=np.where(zircon['cut'] =='Good', '1', zircon['cut'])
zircon['cut']=np.where(zircon['cut'] =='Fair', '0', zircon['cut'])
zircon['color']=np.where(zircon['color'] =='D', '0', zircon['color'])
zircon['color']=np.where(zircon['color'] =='E', '1', zircon['color'])
zircon['color']=np.where(zircon['color'] =='F', '2', zircon['color'])
zircon['color']=np.where(zircon['color'] =='G', '3', zircon['color'])
zircon['color']=np.where(zircon['color'] =='H', '4', zircon['color'])
zircon['color']=np.where(zircon['color'] =='I', '5', zircon['color'])
zircon['color']=np.where(zircon['color'] =='J', '6', zircon['color'])
zircon['clarity']=np.where(zircon['clarity'] =='IF', '0', zircon['clarity'])
zircon['clarity']=np.where(zircon['clarity'] =='VVS1', '1', zircon['clarity'])
zircon['clarity']=np.where(zircon['clarity'] =='VVS2', '2', zircon['clarity'])
zircon['clarity']=np.where(zircon['clarity'] =='VS1', '3', zircon['clarity'])
zircon['clarity']=np.where(zircon['clarity'] =='VS2', '4', zircon['clarity'])
zircon['clarity']=np.where(zircon['clarity'] =='SI1', '5', zircon['clarity'])
zircon['clarity']=np.where(zircon['clarity'] =='SI2', '6', zircon['clarity'])
zircon['clarity']=np.where(zircon['clarity'] =='I1', '7', zircon['clarity'])
```

Table 4: Categorical Data encoding

Note:

1. cut: Quality is increasing order Fair, Good, Very Good, Premium, Ideal.
2. Colour: D being the best and J the worst.
3. Clarity: In order from Best to Worst-IF, VVS1, VVS2, VS1, VS2, Sl1, Sl2, l1

Next let's check the dtypes of the variable:

```
zircon.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 26923 entries, 0 to 26966
Data columns (total 10 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   carat    26923 non-null  float64
 1   cut      26923 non-null  object
 2   color    26923 non-null  object
 3   clarity  26923 non-null  object
 4   depth    26923 non-null  float64
 5   table    26923 non-null  float64
 6   x        26923 non-null  float64
 7   y        26923 non-null  float64
 8   z        26923 non-null  float64
 9   price    26923 non-null  int64
dtypes: float64(6), int64(1), object(3)
memory usage: 3.3+ MB
```

Since, the model works on numerical variables, let us convert the dtype to 'float64'.

```
## Converting the categorical variable to numeric

zircon['cut'] = zircon['cut'].astype('float64')
zircon['color'] = zircon['color'].astype('float64')
zircon['clarity'] = zircon['clarity'].astype('float64')
zircon.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 26923 entries, 0 to 26966
Data columns (total 10 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   carat    26923 non-null  float64
 1   cut      26923 non-null  float64
 2   color    26923 non-null  float64
 3   clarity  26923 non-null  float64
 4   depth    26923 non-null  float64
 5   table    26923 non-null  float64
 6   x        26923 non-null  float64
 7   y        26923 non-null  float64
 8   z        26923 non-null  float64
 9   price    26923 non-null  int64
dtypes: float64(9), int64(1)
memory usage: 3.3 MB
```

Table 5: Dtypes of All variable after Transformation

Let us now check the correlation amongst the predictor variables just to make sure that the predictor variables are not highly correlated amongst themselves.

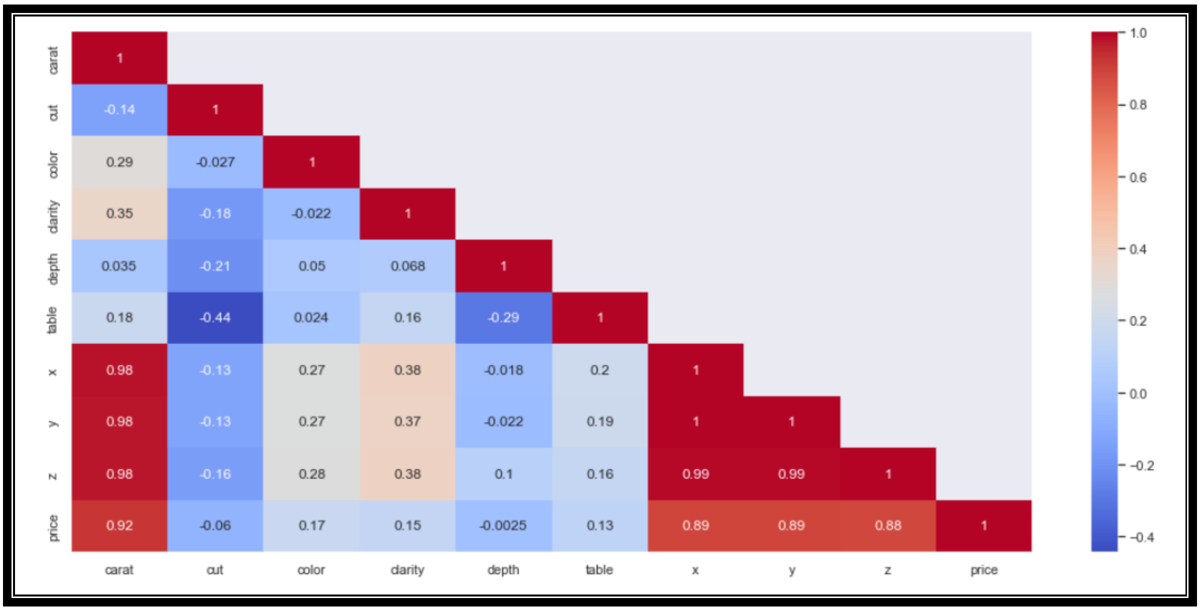


Fig. 1.24: Heatmap showing correlations among all variables after Label Encoding

It maybe observed that carat is positively correlated with x with correlation coefficient 0.98. carat is also positively correlated with y (98%) and positively correlated (98%) with z. All these are moderately high correlations. Likewise, other correlations can also be observed.

Although almost all correlations have been shown to be statistically significant, we will treat only those which are above 0.4 or below −0.4 to be of any importance. Once we agree to impose this restriction, only a few variable pairs show substantial correlation.

Next we perform multiple linear regression. We will build a model with all the variables first. The output is presented as below.

Model1: Price Vs All Variables

```
mod1 = ols('price ~ carat+cut+color+clarity+depth+table+x+y+z', data = zircon).fit()
coefficients = mod1.params
print(coefficients)
```

```
Intercept     1737.568652
carat        11030.948077
cut            127.105698
color         -327.606307
clarity       -494.076718
depth           41.580734
table          -21.763313
x            -1856.856951
y             2136.958108
z            -2005.494085
dtype: float64
```

The explicit form of linear equation is:

Y= 1737.6+ 11030.9carat + 127.1cut – 327color – 494.1clarity+41.6depth−21.8table-1856.8x+2136.9y-2005.5z

```
print(mod1.summary())

                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.909
Model:                            OLS   Adj. R-squared:                  0.909
Method:                 Least Squares   F-statistic:                 2.999e+04
Date:                Sat, 29 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:15:52   Log-Likelihood:            -2.2933e+05
No. Observations:               26923   AIC:                         4.587e+05
Df Residuals:                   26913   BIC:                         4.588e+05
Df Model:                           9
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   1737.5687    929.660      1.869      0.062     -84.614    3559.751
carat       1.103e+04     77.560    142.224      0.000    1.09e+04    1.12e+04
cut          127.1057      8.151     15.594      0.000     111.129     143.082
color       -327.6063      4.580    -71.530      0.000    -336.583    -318.629
clarity     -494.0767      4.986    -99.088      0.000    -503.850    -484.303
depth         41.5807     13.395      3.104      0.002      15.325      67.836
table        -21.7633      4.230     -5.145      0.000     -30.054     -13.472
x          -1856.8570    138.498    -13.407      0.000   -2128.320   -1585.393
y           2136.9581    138.514     15.428      0.000    1865.464    2408.452
z          -2005.4941    198.802    -10.088      0.000   -2395.157   -1615.832
==============================================================================
Omnibus:                     5954.362   Durbin-Watson:                   2.018
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           281517.274
Skew:                          -0.091   Prob(JB):                         0.00
Kurtosis:                      18.840   Cond. No.                     1.09e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.09e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Figure 1.26: MLR Model 1 Summary

Sign of the coef indicates in which direction the response will change, given the predictor increases/decrease by a unit amount.

Any positive coefficient means that a unit increase in the corresponding predictor increases the response by the numerical value of the coefficient provided all other predictors are held at constant level. Any negative coefficient means that a unit increase in the corresponding predictor decreases the response by the value of the coefficient, provided all other predictors are held at constant level.

Note that the sign of carat has not changed, but the numerical value is very different. In general, whether the sign of the regression coefficient of a predictor will remain unchanged in both SLR and MLR cannot be determined beforehand. The sign depends on the correlations among the predictors. Sufficiently high correlations among the predictors can result in disturbance in the sign of the regression coefficient.

In multiple regression, if one or more pairs of explanatory variables is highly correlated among themselves, then the phenomenon is known as **multi-collinearity**.

Effects of Multi-collinearity: multi-collinearity is not desirable. It leads to inflated standard errors of the estimates of the regression coefficients, which in turn affects significance of the regression parameters. Often the signs of the regression coefficients may also change. As a result, the regression model becomes non-reliable or lacks interpretability.

Multicollinearity can be detected via various methods. In this problem, we will focus on the most common one – **VIF (Variable Inflation Factors).**

VIF determines the strength of the correlation between the independent variables. It is predicted by taking a variable and regressing it against every other variable. "

Or

VIF score of an independent variable represents how well the variable is explained by other independent variables.

R^2 value is determined to find out how well an independent variable is described by the other independent variables. A high value of R^2 means that the variable is highly correlated with the other variables. This is captured by the VIF which is denoted below:

$$\text{VIF} = \frac{1}{1-R^2}$$

So, the closer the R^2 value to 1, the higher the value of VIF and the higher the multicollinearity with the particular independent variable.

We now calculate the VIF of each predictor variable.

```
carat  VIF =  25.13
cut  VIF =  1.51
color  VIF =  1.12
clarity  VIF =  1.24
depth  VIF =  6.4
table  VIF =  1.64
x  VIF =  446.51
y  VIF =  440.14
z  VIF =  351.47
```

Figure 1.26.1: VIF of Model 1-All variables

We observe that among all continuous predictors, variable x,y, and z has a sufficiently high VIF's (x=446.51,y=440.14,z=351.47) indicating it is substantially correlated with the other predictor variables. Let's first remove variable 'x' from the model.

Model2: Price Vs All Variables Minus variable 'x'

```
mod2 = ols('price ~ carat+cut+color+clarity+depth+table+y+z', data = zircon).fit()
coefficients = mod2.params
print(coefficients)
```

```
Intercept    -1721.018049
carat        10866.247654
cut            116.331741
color         -327.948775
clarity       -501.138137
depth          102.594942
table          -30.212944
y             1035.965411
z            -3125.009692
dtype: float64
```

```
print(mod2.summary())
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.909
Model:                            OLS   Adj. R-squared:                  0.909
Method:                 Least Squares   F-statistic:                 3.349e+04
Date:                Sat, 29 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:15:53   Log-Likelihood:            -2.2942e+05
No. Observations:               26923   AIC:                         4.589e+05
Df Residuals:                   26914   BIC:                         4.589e+05
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept  -1721.0180    896.113     -1.921      0.055   -3477.447      35.411
carat       1.087e+04     76.835    141.423      0.000    1.07e+04     1.1e+04
cut          116.3317      8.138     14.294      0.000     100.380     132.283
color       -327.9488      4.595    -71.369      0.000    -336.955    -318.942
clarity     -501.1381      4.975   -100.735      0.000    -510.889    -491.387
depth        102.5949     12.640      8.117      0.000      77.819     127.370
table        -30.2129      4.197     -7.199      0.000     -38.439     -21.987
y           1035.9654    111.915      9.257      0.000     816.606    1255.325
z          -3125.0097    181.014    -17.264      0.000   -3479.806   -2770.214
==============================================================================
Omnibus:                     5869.235   Durbin-Watson:                   2.017
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           265418.871
Skew:                          -0.084   Prob(JB):                         0.00
Kurtosis:                      18.381   Cond. No.                     1.04e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.04e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Figure 1.27: MLR Model 2 Summary

Note that the coefficients of the different predictor values have changed. We check the VIFs of the new predictors.

```
carat  VIF =  24.5
cut  VIF =  1.5
color  VIF =  1.12
clarity  VIF =  1.22
depth  VIF =  5.66
table  VIF =  1.6
y  VIF =  285.43
z  VIF =  289.46
```

Figure 1.27.1: VIF of Model 2-w/o variable x

We can see that after removing 'x', there are still predictors with sufficiently high VIF(y=285.43, z=289.46). Our problem of Multicollinearity still exists in the model. Hence let's go ahead and drop variable 'z' and check the VIF values of the predictors again.

```
carat  VIF =  23.98
cut  VIF =  1.5
color  VIF =  1.12
clarity  VIF =  1.22
depth  VIF =  1.41
table  VIF =  1.59
y  VIF =  24.04
```

Figure 1.27.2: VIF of Model 2-w/o variable x & z

We can see that after removing ‘z’, there are still predictors with sufficiently high VIF(y=24.04, carat=23.98). Our problem of Multicollinearity still exists in the model. Hence let’s go ahead and drop variable ‘y’ and check the VIF values of the predictors again.

```
carat  VIF =  1.3
cut  VIF =  1.49
color  VIF =  1.12
clarity  VIF =  1.2
depth  VIF =  1.32
table  VIF =  1.59
```

Figure 1.27.3: VIF of Model 2-w/o variable x,y,z

We can see that after removing x(length),y(width),z(height), all the predictors have low VIF (below 2). So the problem of multi-collinearity has been eliminated. In all our subsequent discussions, we can consider this multiple linear regression model with (x,y,z) removed. Let’s go ahead and build the regression model and check the summary.

Model3: Price Vs All Variables Minus variable ‘x’ ,’y’, ‘z’

```
mod3 = ols('price ~ carat+cut+color+clarity+depth+table', data = zircon).fit()
coefficients = mod3.params
print(coefficients)
```

```
Intercept    3885.277100
carat        8822.589639
cut           120.390883
color        -323.644095
clarity      -522.477917
depth         -44.889760
table         -28.919420
dtype: float64
```

```
print(mod3.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.906
Model:                            OLS   Adj. R-squared:                  0.906
Method:                 Least Squares   F-statistic:                 4.303e+04
Date:                Sat, 29 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:15:54   Log-Likelihood:            -2.2987e+05
No. Observations:               26923   AIC:                         4.598e+05
Df Residuals:                   26916   BIC:                         4.598e+05
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   3885.2771    549.246      7.074      0.000    2808.726    4961.828
carat       8822.5896     18.014    489.754      0.000    8787.281    8857.899
cut          120.3909      8.264     14.568      0.000     104.193     136.589
color       -323.6441      4.670    -69.299      0.000    -332.798    -314.490
clarity     -522.4779      5.006   -104.375      0.000    -532.289    -512.666
depth        -44.8898      6.199     -7.241      0.000     -57.041     -32.739
table        -28.9194      4.256     -6.795      0.000     -37.262     -20.577
==============================================================================
Omnibus:                     5226.785   Durbin-Watson:                   2.011
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            98453.868
Skew:                           0.419   Prob(JB):                         0.00
Kurtosis:                      12.331   Cond. No.                     6.17e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 6.17e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Figure 1.28: Model 3 Summary

In above Model summary all the continuous predictor's p-value in the regression table is less than a pre-fixed level α. If the predictors p-value was greater than α then we could have simply eliminated the variable from the regression equation. However, we cannot do that in case.

The coefficient of determination, R2 shows 90.6% whereas adj R2 shows 90.6%. Both are showing equivalent values in the model.

Imp Note: Recall that in simple linear regression, $R2$ is the square of the pairwise correlation coefficient between the single predictor X and the response Y. In MLR no such interpretation of $R2$ holds. In MLR adj R2 value is important. Adjusted R2measure involves an adjustment based on the number of predictors relative to the sample size.

Though we have got good values of vif for below predictors, Still we can drop 'table' variable as that has a comparatively high vif among all.

carat VIF = 1.3
cut VIF = 1.49
color VIF = 1.12
clarity VIF = 1.2
depth VIF = 1.32
table VIF = 1.59

Dropping variables means losing out on information. That can hamper the predictive as well as the descriptive power of the model. Let's drop table and build the model and check the performance.

Model 4: Price Vs All Variables Minus variable 'x', 'y', 'z' and table

```
mod4 = ols('price ~ carat+cut+color+clarity+depth', data = zircon).fit()
coefficients = mod4.params
print(coefficients)
```

```
Intercept      991.446836
carat         8808.302621
cut            149.663244
color         -323.690912
clarity       -524.608324
depth          -25.990087
dtype: float64
```

```
print(mod4.summary())
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.905
Model:                            OLS   Adj. R-squared:                  0.905
Method:                 Least Squares   F-statistic:                 5.154e+04
Date:                Sat, 29 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:15:54   Log-Likelihood:            -2.2990e+05
No. Observations:               26923   AIC:                         4.598e+05
Df Residuals:                   26917   BIC:                         4.599e+05
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    991.4468    347.122      2.856      0.004     311.069    1671.824
carat       8808.3026     17.906    491.913      0.000    8773.206    8843.400
cut          149.6632      7.058     21.204      0.000     135.829     163.498
color       -323.6909      4.674    -69.251      0.000    -332.853    -314.529
clarity     -524.6083      5.000   -104.919      0.000    -534.409    -514.808
depth        -25.9901      5.545     -4.687      0.000     -36.859     -15.121
==============================================================================
Omnibus:                     5240.846   Durbin-Watson:                   2.011
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            97806.991
Skew:                           0.426   Prob(JB):                         0.00
Kurtosis:                      12.298   Cond. No.                     2.86e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.86e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Figure 1.29: Model 4 Summary

Let's check the VIF of remaining predictors after dropping 'table'

```
carat  VIF =  1.28
cut  VIF =  1.09
color  VIF =  1.12
clarity  VIF =  1.19
depth  VIF =  1.05
```

Figure 1.29.1: VIF of Model 4-w/o x,y,z & table

We will drop 'depth' variable and build the model to check the performance of the model.

Model 5: Price Vs All Variables Minus variable 'x', 'y', 'z', table and depth

```
mod5 = ols('price ~ carat+cut+color+clarity', data = zircon).fit()
coefficients = mod5.params
print(coefficients)
```

```
Intercept    -627.991627
carat        8810.064116
cut           156.392374
color        -324.807588
clarity      -525.482927
dtype: float64
```

```
print(mod5.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.905
Model:                            OLS   Adj. R-squared:                  0.905
Method:                 Least Squares   F-statistic:                 6.436e+04
Date:                Sat, 29 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:15:54   Log-Likelihood:            -2.2991e+05
No. Observations:               26923   AIC:                         4.598e+05
Df Residuals:                   26918   BIC:                         4.599e+05
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   -627.9916     33.382    -18.812      0.000    -693.422    -562.561
carat       8810.0641     17.909    491.929      0.000    8774.961    8845.167
cut          156.3924      6.913     22.622      0.000     142.842     169.943
color       -324.8076      4.670    -69.553      0.000    -333.961    -315.654
clarity     -525.4829      4.999   -105.126      0.000    -535.280    -515.685
==============================================================================
Omnibus:                     5230.225   Durbin-Watson:                   2.012
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            98831.512
Skew:                           0.419   Prob(JB):                         0.00
Kurtosis:                      12.349   Cond. No.                         26.5
==============================================================================

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

Figure 1.30: Model 5 Summary

```
carat  VIF =  1.28
cut  VIF =  1.04
color  VIF =  1.12
clarity  VIF =  1.19
```

Figure 1.30: VIF of Model 5-w/o x, y, z, table& depth

Compare:

comparing mod3, mod4 and mod5 the adj R2 and R2 value has not changed at all. its same. Hence, eliminating variable x, y, z in mod3 has definitely removed the multicollinearity problem from our model. Later eliminating variables depth and table which had very less correlation with our predictor we saw that it is not affecting the regression model at all.

- There is almost no change in the $R2$ values.
- While adding or subtracting variables from a regression model to refine the model, we need to be very careful about the Adjusted $R2$ values. Adding any particular value which is not significant can increase the $R2$ value but the Adjusted $R2$ changes by the addition or the subtraction of significant variables.

Next step is making prediction based on our model.

```
mod3_pred = mod3.fittedvalues
mod4_pred = mod4.fittedvalues
mod5_pred = mod5.fittedvalues
mod5_pred
```

```
0        -311.625115
1        1774.083891
2        6238.077385
3        1471.740844
4        1553.599645
            ...
26962    6018.519269
26963    1605.668678
26964    1907.578901
26965     362.929404
26966    6277.505482
Length: 26923, dtype: float64
```

The fitted values of the response and the residuals can be extracted directly from the model.

Below is the code and output of the same:

```
#Extraction of fitted response
model_fitted_y = pd.DataFrame(mod5.fittedvalues,columns= ['Estimated'])

# Extraction of residuals
model_residuals = pd.DataFrame(mod5.resid , columns= ['Residual'])
d1 = pd.concat([zircon, model_fitted_y,model_residuals], axis=1, ignore_index=True)
d1.columns  = ['carat','cut','color','clarity','depth','table','x','y','z','price','Estimated','Residuals']
d1.loc[[0,1,3,7,8,14,20],['carat','cut','color','clarity','depth','table','price','Estimated','Residuals']]
#d1
```

	carat	cut	color	clarity	depth	table	price	Estimated	Residuals
0	0.30	4.0	1.0	5.0	62.1	58.0	499	-311.625115	810.625115
1	0.33	3.0	3.0	0.0	60.8	58.0	984	1774.083891	-790.083891
3	0.42	4.0	2.0	3.0	61.6	56.0	1082	1471.740844	-389.740844
7	0.50	3.0	1.0	5.0	61.5	62.0	1415	1293.995333	121.004667
8	1.21	1.0	4.0	5.0	63.8	64.0	5407	6261.933344	-854.933344
14	1.50	0.0	3.0	4.0	66.2	53.0	10644	9510.750077	1133.249923
20	1.04	3.0	0.0	2.0	61.1	60.0	10984	7952.686324	3031.313676

Table 6: DataFrame of actual and predicted values

Model Evaluation

There are three primary metrics used to evaluate linear models. These are: Mean absolute error (MAE), Mean squared error (MSE), or Root mean squared error (RMSE). Here we will use RMSE.

RMSE: Most popular metric, Root Mean Square Error (RMSE) is **the standard deviation of the residuals** (prediction errors). 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. Lower values of RMSE indicate better fit.

Below is model evaluation on adj R2 values.

	model_name	model_perf
0	SLR	0.850795
1	Model 2 All Predictors	0.909294
2	Model 3 w/o x,y,z	0.905561
3	Model 4 w/o x,y,z and table	0.905403
4	Model 5 w/o x,y,z,table and depth	0.905329

Table 6: Adj R2 values of All Models

Let's have a look at the **RMSE scores** of Model no. 3, 4, and 5

Model 3 - RMSE

```
metrics.mean_squared_error(zircon['price'], mod3_pred, squared=False)
```

```
95]: 1235.466728888604
```

Model 4 - RMSE

```
metrics.mean_squared_error(zircon['price'], mod4_pred, squared=False)
```

```
96]: 1236.5258904068312
```

Model 5 - RMSE

```
metrics.mean_squared_error(zircon['price'], mod5_pred, squared=False)
```

```
97]: 1237.030382390836
```

Table 7: RMSE values of Model 3, 4 and 5

Scatter Plot of Model 3, 4, and 5 predictions

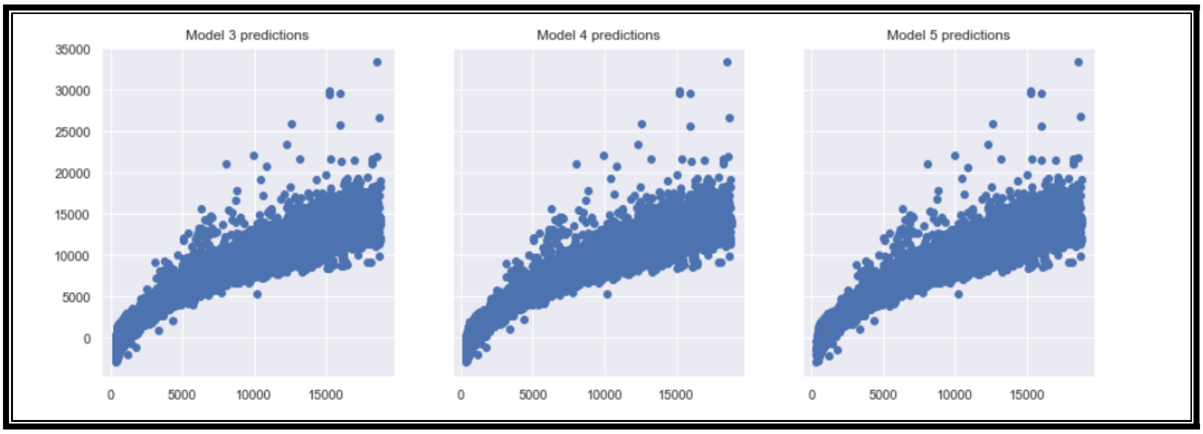


Figure 1.31 : Scatter plot of Model 3,4 and 5 prediction values

The scatter plot above is showing the relationship between actual and predicted values of all the 3 Model we are comparing. With regression analysis, we can use a scatter plot to visually inspect the data to see whether X and Y are linearly related. They seems to be positively related and the data points are very closely fitted to each other. Though data points are scattered and are taken as outliers but they are relevant points in the data.

In order to make valid inferences from our regression, the residuals of the regression should follow a normal distribution. The residuals are **simply the error terms, or the differences between the observed value of the dependent variable and the predicted value**.

Distplot of Model 3, 4, and 5 Residuals

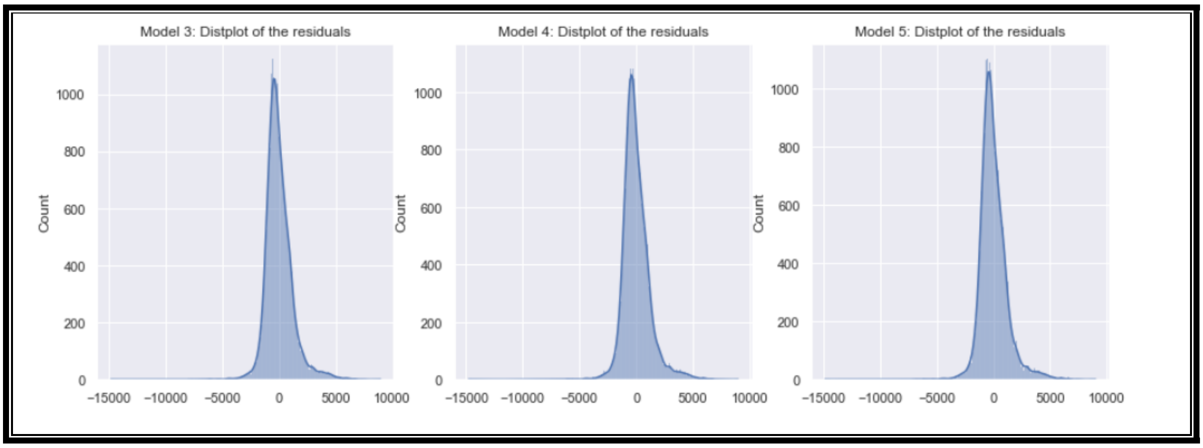


Figure 1.32 : Distribution plot of Model 3,4 and 5 prediction values

Above plot shows distribution plot of prediction values of the Model we build(final and comparing model) All the plot shows normal distribution. As we see from the plot the kurtosis is very high at a peak.

Boxplot of Model 3, 4, and 5 Residuals

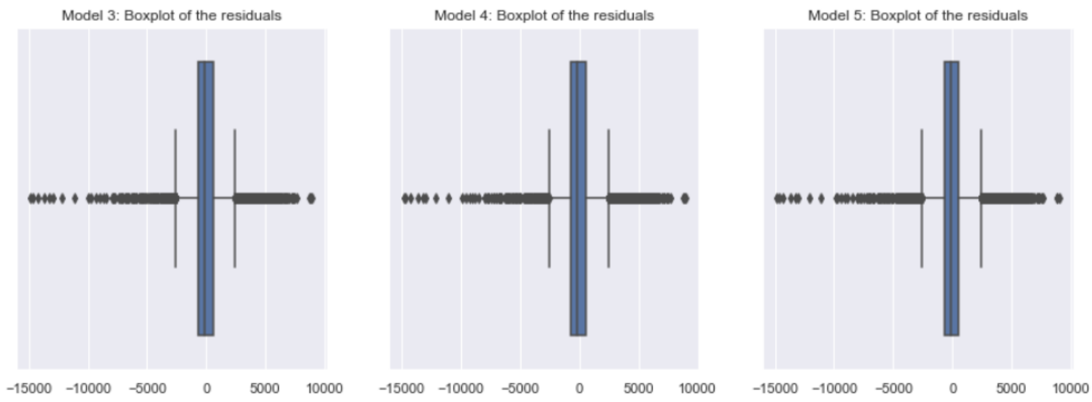


Figure 1.34 : Boxplot of Model 3,4 and 5 prediction values

It is very difficult to make out any difference in visual representation of the models we built.

The above boxplot shows the residuals to assess the overall accuracy of the model. All the model median looks same. All the boxplot shows outliers too. We have already discussed above as to why we did not treat the outliers and how the data given to us looks relevant.

Concluding above Model Building and Evaluation.

We have built SLR model between price and carat(most correlated independent variable with price). Then, we built various MLR model using statmode library without splitting the dataset in train and test set(we will do this in 1.3).

The overall idea of regression is to examine two things:

(1) does a set of predictor variables do a good job in predicting an outcome (dependent) variable?

(2) Which variables in particular are significant predictors of the outcome variable, and in what way do they–indicated by the magnitude and sign of the beta estimates–impact the outcome variable? These regression estimates are used to explain the relationship between one dependent variable and one or more independent variables.

After evaluating all the model above (refer to model Evaluation table), were we compared the adj R2 values of Model3, 4 and 5 and also RMSE values of Model 3, 4 and 5. In Model 3 we got rid of multicollinearity problem by removing the x, y, z variables. After that in Model 4 and 5 we removed table and depth variable but it did not showed much of progress or down gradation of adj R2 values in our model.

Hence, We can conclude that our Model 3 is the best model with good adj R2 value of 90.5% and low RMSE value of 1235.46. This model has the important predictors that is important in predicting our price and directly and indirectly effect our target variable i.e. price.

FINAL MODEL IS MODEL 3 with predictors carat, cut, color, clarity, depth and table

Thus, the final regression equation becomes:

price= 3885.277100+8822.589639*carat+ 120.390883*cut -323.644095*color -522.477917*clarity -44.889760*depth -28.919420*table

The above linear equation says that a unit increase in carat will increase price by 8822.58 unit
A unit increase in cut will increase the price by 120.39 unit.
A unit increase in color will decrease the price by 323.64 unit
A unit increase in clarity will decrease the price by 522.47 unit.
A unit increase in depth will decrease the price by 44.89 unit.
A unit increase in table will decrease the price by 28.9 unit.

The fitted values of the response and the residuals can be extracted directly from the model.

```
mod3_pred = mod3.fittedvalues
mod4_pred = mod4.fittedvalues
mod5_pred = mod5.fittedvalues
mod3_pred
```

```
0        -387.396589
1        1780.348301
2        6170.481366
3        1472.909626
4        1514.490053
            ...
26962    5982.243994
26963    1614.474611
26964    1885.390174
26965     422.239380
26966    6259.941187
Length: 26923, dtype: float64
```

Below we present some values of the predictors, the observed response, predicted response and the residuals.

	carat	cut	color	clarity	depth	table	price	Estimated	Residuals
0	0.30	4.0	1.0	5.0	62.1	58.0	499	-387.396589	886.396589
1	0.33	3.0	3.0	0.0	60.8	58.0	984	1780.348301	-796.348301
3	0.42	4.0	2.0	3.0	61.6	56.0	1082	1472.909626	-390.909626
7	0.50	3.0	1.0	5.0	61.5	62.0	1415	1167.986634	247.013366
8	1.21	1.0	4.0	5.0	63.8	64.0	5407	6059.225941	-652.225941
14	1.50	0.0	3.0	4.0	66.2	53.0	10644	9553.886257	1090.113743
20	1.04	3.0	0.0	2.0	61.1	60.0	10984	7899.057629	3084.942371

1.3 Split the data into training (70%) and test (30%). Build the various iterations of the Linear Regression models on the training data and use those models to predict on the test data using appropriate model evaluation metrics.

Here first we are going to use Sklearn library to split and the training and test data set and building the Regression model.

```
from sklearn.linear_model import LinearRegression
```

```
X = zircon.drop('price', axis=1)
Y = zircon[['price']]
```

```
#Let us break the X and y dataframes into training set and test set. For this we will use
#Sklearn package's data splitting function which is based on random function

from sklearn.model_selection import train_test_split

# Split X and y into training and test set in 75:25 ratio

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.30 , random_state=1)
```

```
# invoke the LinearRegression function and find the bestfit model on training data

regression_model = LinearRegression()
regression_model.fit(X_train, Y_train)

LinearRegression()
```

Next, find the coefficient of the independent/predictor variables.

```
# # Let us explore the coefficients for each of the independent attributes

for idx, col_name in enumerate(X_train.columns):
    print("The coefficient for {} is {}".format(col_name, regression_model.coef_[0][idx]))

The coefficient for carat is 10726.656855882125
The coefficient for cut is 121.6914292151142
The coefficient for color is -326.3111489489219
The coefficient for clarity is -503.28199478158103
The coefficient for depth is 99.27697106836679
The coefficient for table is -21.624714273460683
The coefficient for x is -1532.7852151976497
The coefficient for y is 2490.5356570836216
The coefficient for z is -2901.08731996785
```

Next calculate the intercept. Pls, note it is difficult to interpret the intercept values.

```
# # Let us check the intercept for the model

intercept = regression_model.intercept_[0]

print("The intercept for our model is {}".format(intercept))

The intercept for our model is -2251.047116009021
```

The above intercept value say that when all the independent values are zero then the Y value becomes -2251 which doesn't make sense. Hence, it is difficult to interpret the intercept.

Next, calculate the **Accuracy Score of Train and Test Set.**

```
Train_score=regression_model.score(X_train, Y_train)
print("Train accuracy score:",Train_score)
Test_score=regression_model.score(X_test, Y_test)
print("Test accuracy score:",Test_score)

Train accuracy score: 0.9063123513457847
Test accuracy score: 0.9157489718279384
```

Table 8: Accuracy Score of Train and Test Set

The above accuracy score of Train and test are the coefficient of determinant i.e. R^2. R^2 is generally not considered for evaluating the model. hence, we use adj R^2.

Why? R^2 is not a reliable metric as it always increases with addition of more attributes even if the attributes have no influence on the predicted variable.

Instead, we use adjusted R^2 which removes the statistical chance that improves R^2.

Since this is regression, we will plot the predicted Y value vs actual Y values for the test data.
A good model's prediction will be close to actual leading to high R and R2 values

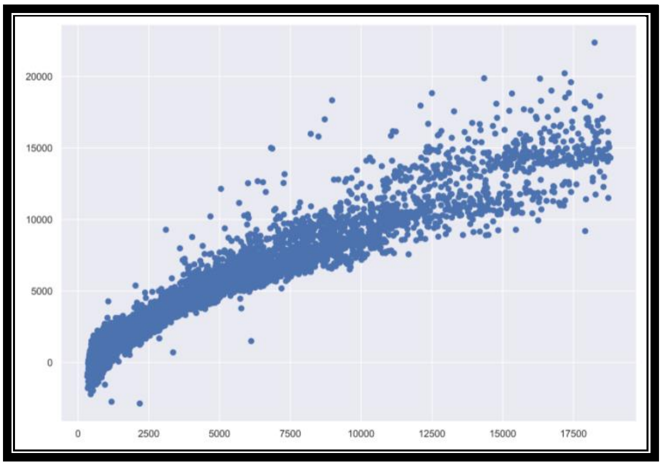


Figure 1.35: Scatterplot on test data between dependent and independent variables.

Note: Scikit does not provide a facility for adjusted R^2... so we use statsmodel, a library that gives results similar to what you obtain in R language.

This library expects the X and Y to be given in one single dataframe.

When we build the model, the model can end up in underfit and overfit zone. We want right fit. right fit is that situation when the model equally performs well on train and test data. Our train and test scores doesn't differ much. Hence this is an indication that the model is in right fit zone.

Note: When we use Sklearn library it builds linear Regression model but it avoids many statistical reason behind the linear regression models. Often Linear Regression models are taught as part of statistical learning. We will now build Linear Regression model using **Statsmodel** library. As Sklearn library doesn't give us all the statistical information of the data we use.

Here, lets concat X_train and Y_train in one dataframe.

```
zircon_train = pd.concat([X_train, Y_train], axis=1)
zircon_train.head()
```

	carat	cut	color	clarity	depth	table	x	y	z	price
2665	0.72	4.0	1.0	1.0	61.4	57.0	5.74	5.79	3.54	4864
7774	1.20	3.0	4.0	5.0	62.4	59.0	6.72	6.68	4.18	5592
9339	1.00	3.0	2.0	4.0	62.2	57.0	6.43	6.40	3.99	6296
1025	0.45	4.0	0.0	6.0	62.0	55.0	4.92	4.95	3.06	706
3558	1.10	3.0	3.0	4.0	62.8	58.0	6.60	6.58	4.14	6387

Let's build a LR model one by one to check the accuracy for Train and Test data.

<u>LR Model 1 Using all variables to build the model on the training data</u>

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.906
Model:                            OLS   Adj. R-squared:                  0.906
Method:                 Least Squares   F-statistic:                 2.025e+04
Date:                Sun, 30 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:22:07   Log-Likelihood:            -1.6078e+05
No. Observations:               18846   AIC:                         3.216e+05
Df Residuals:                   18836   BIC:                         3.217e+05
Df Model:                           9
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept  -2251.0471   1281.624     -1.756      0.079   -4763.146     261.052
carat       1.073e+04     93.499    114.725      0.000    1.05e+04    1.09e+04
cut          121.6914      9.934     12.250      0.000     102.219     141.164
color       -326.3111      5.568    -58.604      0.000    -337.225    -315.397
clarity     -503.2820      6.029    -83.477      0.000    -515.099    -491.465
depth         99.2770     19.086      5.201      0.000      61.866     136.688
table        -21.6247      5.130     -4.216      0.000     -31.679     -11.570
x          -1532.7852    182.279     -8.409      0.000   -1890.069   -1175.501
y           2490.5357    184.413     13.505      0.000    2129.070    2852.001
z          -2901.0873    293.751     -9.876      0.000   -3476.866   -2325.309
==============================================================================
Omnibus:                     4468.819   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           240416.674
Skew:                          -0.208   Prob(JB):                         0.00
Kurtosis:                      20.493   Cond. No.                     1.25e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.25e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Figure 1.36: LR Model 1 Summary

```
carat  VIF =  24.75
cut  VIF =  1.51
color  VIF =  1.12
clarity  VIF =  1.23
depth  VIF =  8.79
table  VIF =  1.65
x  VIF =  524.7
y  VIF =  529.24
z  VIF =  519.25
```

Figure 1.36.1: VIF of LR Model 1- All variables

Observation:

From above summary we see that the R2 and adj R2 values are good. The p_values of the independent variable are zero i.e. p_values <0.05 significant level, hence statistically their coefficients are very reliable. Hence, making the model reliable. But, as we are aware that independent variable x, y, z are highly correlated to independent variable(carat) itself. We calculated the VIF to eliminate the multicollinearity (explained in detail in 1.2) problem from the model. Hence we see that x, y, and z show extreme high VIF's. Let us build the model without these variables.

LR Model 2 Using variables of model 3 to build the model on the training data

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.903
Model:                            OLS   Adj. R-squared:                  0.903
Method:                 Least Squares   F-statistic:                 2.927e+04
Date:                Sun, 30 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:40:45   Log-Likelihood:            -1.6110e+05
No. Observations:               18846   AIC:                         3.222e+05
Df Residuals:                   18839   BIC:                         3.223e+05
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   3976.9604    663.692      5.992      0.000    2676.064    5277.856
carat       8801.9622     21.763    404.455      0.000    8759.306    8844.619
cut          114.2679     10.020     11.403      0.000      94.627     133.909
color       -321.9329      5.655    -56.925      0.000    -333.018    -310.848
clarity     -528.2921      6.034    -87.556      0.000    -540.119    -516.465
depth        -46.0315      7.493     -6.143      0.000     -60.718     -31.345
table        -28.2853      5.134     -5.510      0.000     -38.348     -18.223
==============================================================================
Omnibus:                     3711.382   Durbin-Watson:                   1.991
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            91796.882
Skew:                           0.310   Prob(JB):                         0.00
Kurtosis:                      13.794   Cond. No.                     6.17e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 6.17e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Figure 1.37: LR Model 2 Summary

```
carat  VIF =  1.3
cut  VIF =  1.49
color  VIF =  1.11
clarity  VIF =  1.2
depth  VIF =  1.31
table  VIF =  1.6
```

Figure 1.37.1: VIF of LR Model 2 w/o x,y,z

Observation:

The above Model 2 shows R2 and adj R2 value as 90.3%(no difference in their values). Also, the p_values are all zero showing statistically significant. And telling us that our model is a good fit. To check on Multicollinearity, we checked the VIF's and as we see the, all the predictors VIF's are below 2. Hence, making this Model2 an ideal model to conclude.

Just to have a clear picture of whether the model is a good fit or not, we built 2 more Model, one without table and one without depth to check on the adj R2 values. Below is the snippet of the same.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.903
Model:                            OLS   Adj. R-squared:                  0.903
Method:                 Least Squares   F-statistic:                 3.507e+04
Date:                Sun, 30 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:49:10   Log-Likelihood:            -1.6111e+05
No. Observations:               18846   AIC:                         3.222e+05
Df Residuals:                   18840   BIC:                         3.223e+05
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   1141.4368    419.414      2.722      0.007     319.348    1963.526
carat       8788.1083     21.634    406.225      0.000    8745.704    8830.512
cut          143.0157      8.562     16.704      0.000     126.234     159.797
color       -321.8771      5.660    -56.871      0.000    -332.971    -310.784
clarity     -530.4058      6.026    -88.016      0.000    -542.218    -518.594
depth        -27.4748      6.699     -4.102      0.000     -40.605     -14.345
==============================================================================
Omnibus:                     3716.453   Durbin-Watson:                   1.992
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            91239.783
Skew:                           0.316   Prob(JB):                         0.00
Kurtosis:                      13.761   Cond. No.                     2.86e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.86e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Table 9: Model 3 without table variable

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.903
Model:                            OLS   Adj. R-squared:                  0.903
Method:                 Least Squares   F-statistic:                 4.379e+04
Date:                Sun, 30 Jan 2022   Prob (F-statistic):               0.00
Time:                        19:50:14   Log-Likelihood:            -1.6112e+05
No. Observations:               18846   AIC:                         3.223e+05
Df Residuals:                   18841   BIC:                         3.223e+05
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   -570.7669     40.590    -14.062      0.000    -650.328    -491.206
carat       8790.2856     21.636    406.277      0.000    8747.877    8832.695
cut          149.9174      8.398     17.851      0.000     133.456     166.379
color       -322.8909      5.657    -57.081      0.000    -333.979    -311.803
clarity     -531.2257      6.025    -88.163      0.000    -543.036    -519.415
==============================================================================
Omnibus:                     3710.687   Durbin-Watson:                   1.992
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            92374.480
Skew:                           0.306   Prob(JB):                         0.00
Kurtosis:                      13.829   Cond. No.                         26.7
==============================================================================

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

Table 10: Model 4 without depth variable

Observation:

No change in the value of R2 and adj R2 i.e. 90.3% of both the model 3 and 4. P_values of all the predictors showing no change in their value i.e. zero.

Next, as one of the common step of the Model building we will go ahead and fit the model and predict on train and test data set to further check the accuracy and RMSE scores.

Model Evaluation

	adj R2
Model 1 with all variables	0.906
Model 2 w/o x,y,z variables	0.903
Model 3 w/o x,y,z,table variables	0.903
Model 4 w/o x,y,z,table,depth variables	0.903

RMSE Scores of all the Model we build, have a look below:

	RMSE Training Data	RMSE Test Data
Model 1 with all variables	1227.20	1174.72
Model 2 w/o x,y,z variables	1247.88	1206.40
Model 3 w/o x,y,z,table variables	1248.88	1207.58
Model 4 w/o x,y,z,table,depth variables	1249.44	1207.95

Table 11: RMSE scores of Training and Test Set

As per above RMSE score though Model 1 shows the lowest RMSE score, we can not select that as we are aware that in Model 1 x,y,z variable are highly corelated with other important predictors. Hence, the next best RMSE score is of Model2. Which will be our Final Model.

Scatter Plot of Model of Train data set of all the Model we built

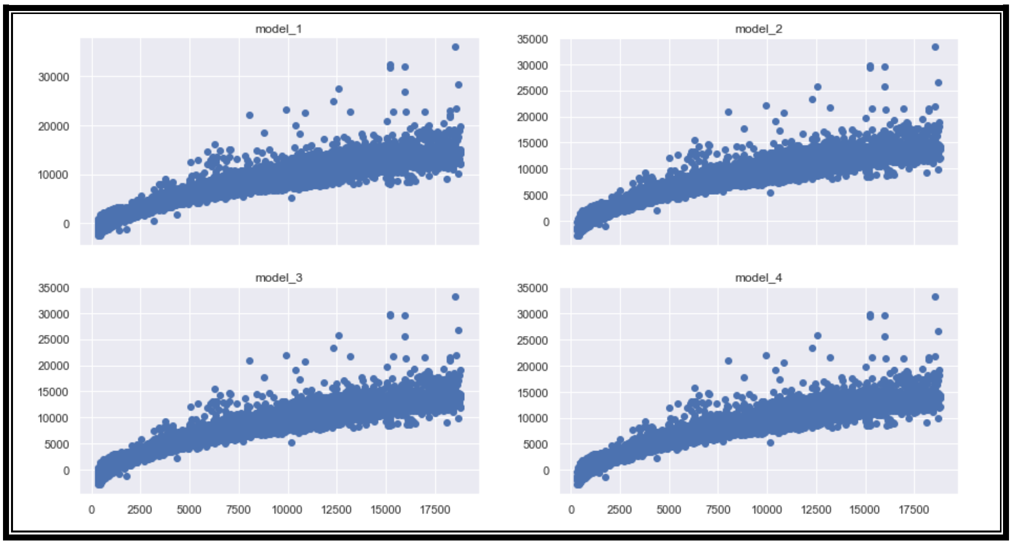


Figure 1.38: Scatter Plot of Train dataset of All Model

The above and below scatter plot of the all the model shows no major visual differences. All the model shows a positive relation between X and Y. The data points are very closely fitted to each other.

Scatter Plot of all Model of Test data set of all the Model we build

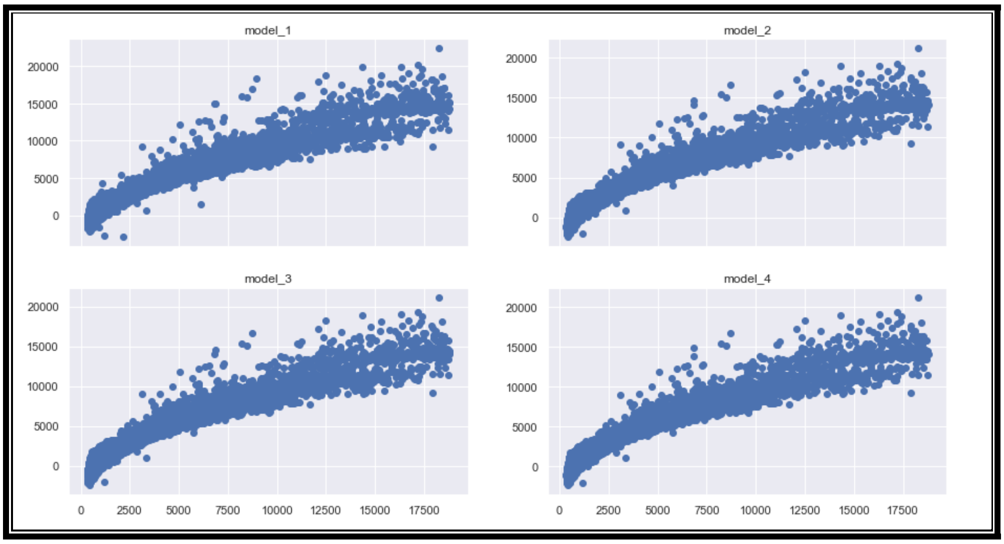


Figure 1.39: Scatter Plot of Test dataset of All Model

Final Model we Choose: Model 2 with predictors carat, cut, color, clarity, depth and table.

Lets do Scaling and check the effect on Train and Test data set: As we have seen in 1.2 earlier, with this specific dataset, we don't think we need to scale the data. However, to see its impact, let's quickly view the results post scaling the data.

We are going to us Z score to scale the data. The standard score (more commonly referred to as a z-score) is a very useful statistic because it (a) **allows us to calculate the probability of a score occurring within our normal distribution** and (b) enables us to compare two scores that are from different normal distributions.

Z-score is a **variation of scaling that represents the number of standard deviations away from the mean**. You would use z-score to ensure your feature distributions have mean = 0 and std = 1.

```
from scipy.stats import zscore

X_train_scaled  = X_train.apply(zscore)
X_test_scaled = X_test.apply(zscore)
Y_train_scaled = Y_train.apply(zscore)
Y_test_scaled = Y_test.apply(zscore)
```

Look at the below coefficients of the independent variable after scaling.

```
The coefficient for carat is 1.2730148188366235
The coefficient for cut is 0.03359090963055609
The coefficient for color is -0.13814678806785616
The coefficient for clarity is -0.20685514601440363
The coefficient for depth is 0.0344011332414558
The coefficient for table is -0.012066821572828278
The coefficient for x is -0.4295834749135074
The coefficient for y is 0.692908537076224
The coefficient for z is -0.5018979551300795
```

The values are scaled now.

After scaling the intercept value is -1.8361956875280148e-16, which is almost equal to Zero. Which is the result of scaling transformation.

The scaled Linear Regression equation will be, where the interpretability will be with an unit increase in Standard deviation and not the unit increase as in normal linear model.:

~ 0 + 1.27*carat + 0.33*cut – 0.14*color - 0.21*clarity + 0.03*depth – 0.01*table -0.43*x + 0.69*y – 0.50*z

Model score of train data set (90.6%) and test data set (91.6%) is exactly the same as before scaling the data. Hence, we can conclude that scaling does not impact our model at all.

RMSE is 0.29026. This means we have almost 29% variance of residual error or unexplained error in our model.

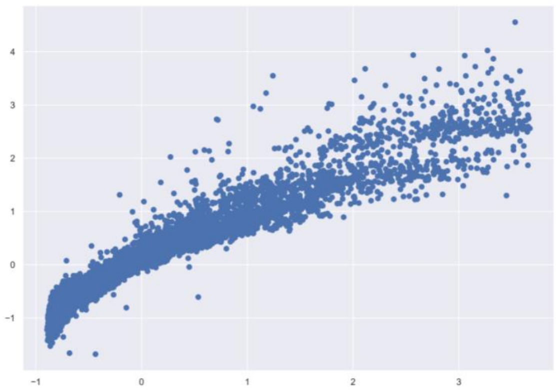


Fig: After scaling Scatterplot on test data between dependent and the independent variables.
The scatterplot also showing similar pattern like before scaling the data. Hence, our conclusion is right.

Note: Final Model selected in Q1.2 and Q1.3 are giving same scores(before and after scaling model performance) and we have got same predictors as well. Performing different transformation and scaling of data has not given any different adj R2 values, RMSE value and accuracy values.

Insight and Recommendation:

Insight:

We have a dataset in which we saw that there is high correlation between the independent variables. Hence, while collecting the data company should take care of the same in order to handle the issue of Multicollinearity which affects the model performance.

Multicollinearity makes it difficult to understand how one variable influences the target variable. However, it does not affect the model accuracy as we saw above while model building with all variables and dropping variables too.

While doing EDA analysis, we saw that carat is a very strong independent variable having very high correlation with x, y, z variables. Carat also showed very low correlation with other variables like table, depth, cut, color, clarity as well. There, we could establish that carat is our the most strong predictor. Even after encoding the categorical variable, carat still showed the same results.

With all the model we built we saw that carat was the most strong predictors among other independent variables, followed with cut, color, clarity, table and depth variable. Even after scaling the data we had the same results.

We also, looked into treating multicollinearity with the help of VIF score. As we mentioned earlier too that any variable with VIF score of greater than 10 has been accepted to indicate severe collinearity. Hence, with the help if VIF score we could eliminate whose variable with highest vif score in return handing the multicollinearity problem in the dataset.

For the business based model, the model we created for the test data set(future unseen data), the key variables/predictors that are likely to drive the change(increase/decrease) of our target variable i.e. price are-Carat, Cut, Color, Clarity, table and depth.

Recommendation:

- Diamond is a luxury which every women once in their life time would like to have it as a precious possession.
- As expected, Carat is a strong predictor to predict the price of the zircon diamonds.
- Variable cut refers to diamond's proportion, symmetry and polish. The beauty of diamond depends more on cut than any other factors. Diamond cut has three primary effects on appearance: brilliance, fire and scintillation. Therefore, the cut grade is so important. It allows the purchaser to identify those stones that were cut Fair to Ideal to gain carat weight. Thus, it is very important that the seller should focus more on cut of the zircon diamond.
- If company wants to increase their revenue and profit they should more focus on stones of color D,E, and F to charge relatively higher price which will in turn increase the sale. Color of diamond is a very important factor of diamond, a yellow tint will adversely affect the price. So, the company should always have high color grade diamonds to increase the revenue of the store.
- As we know a lower color diamond with a higher cut grade will have more sparkle and visual appeal than a higher color diamond with a lower cut grade. Hence, Company should always focus on Carat, cut and Clarity of the diamond as a important factor to increase the sale and profit of the company.

- Clarity is know to the inclusion and blemish of the diamond. A good clarity stone does helps the company to put a high price of the diamond. Hence, company should try to procure high grade clarity diamonds.
- Company should also focus on different cut, color of diamonds to attract not only the brides but women who would like to have different color and cut for different wearables.
- Not only wedding rings, company should also focus on have different jewellery with top notch quality of diamonds to increase their revenue. Like bracelet, bangle, earrings, middle finger ring, pendant of different color stones and different cut., etc.
- In my opinion, company should always have different range of different grade of diamonds to not only attract the customers for wedding or engagement but should make it available to those who would like to wear diamonds in other occasion too or even for gifting purpose.
- Company can also strategies and can segment their diamonds based on customer pay scale .

Problem 2: Logistic Regression

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.

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.

Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.

In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}\text{; 0 for y= 0, and infinity for y=1}$$

But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

Data Dictionary:

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

1.1 Exploratory Data Analysis

To start with the analysis let's look at the sample data and perform basic checks.

	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

Table1: Top 5 rows of the Dataset.

Inferences:
1. The dataset has a total of six independent variables -5 are continuous and 1 is categorical and 1 is the dependent/target variable.
2. Shape (dimension) of the Dataset is (872, 7).
3. No NULL values& duplicate values are present in the dataset.

Univariate analysis –

To perform Univariate analysis on continuous variables, let us start with looking at the summary statistics of the dataset.

```
holiday_data.describe(include='all')
```

	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

Table2: Summary Statistics of the Dataset.

Checking the Unique values of Categorical Variables in our Data.

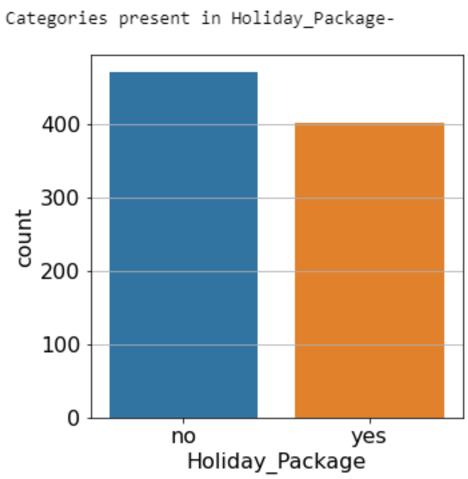


Figure 1: Countplot of Target Variable ‘Holiday_Package’

We see that 54% of the employees are not opting for the Holiday Packages and 46% are interested and opting for Holiday packages. This implies that we have dataset which is fairly balanced.

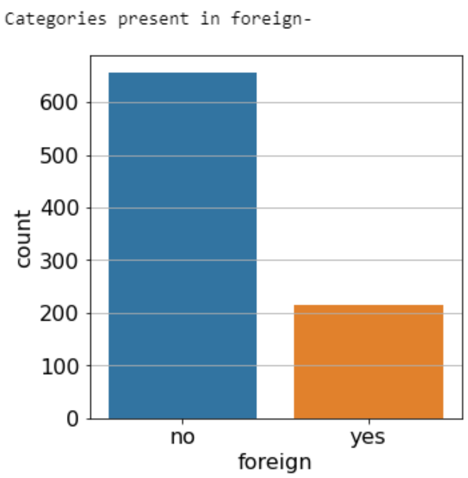


Figure 2: Countplot of Target Variable ‘foreign’

We see that 75% of employees are not Foreigner and rest 25% employees are Foreigners.

Analysing the Box plots & Distribution plots for Continuous Variables –

Starting with Numerical Variable

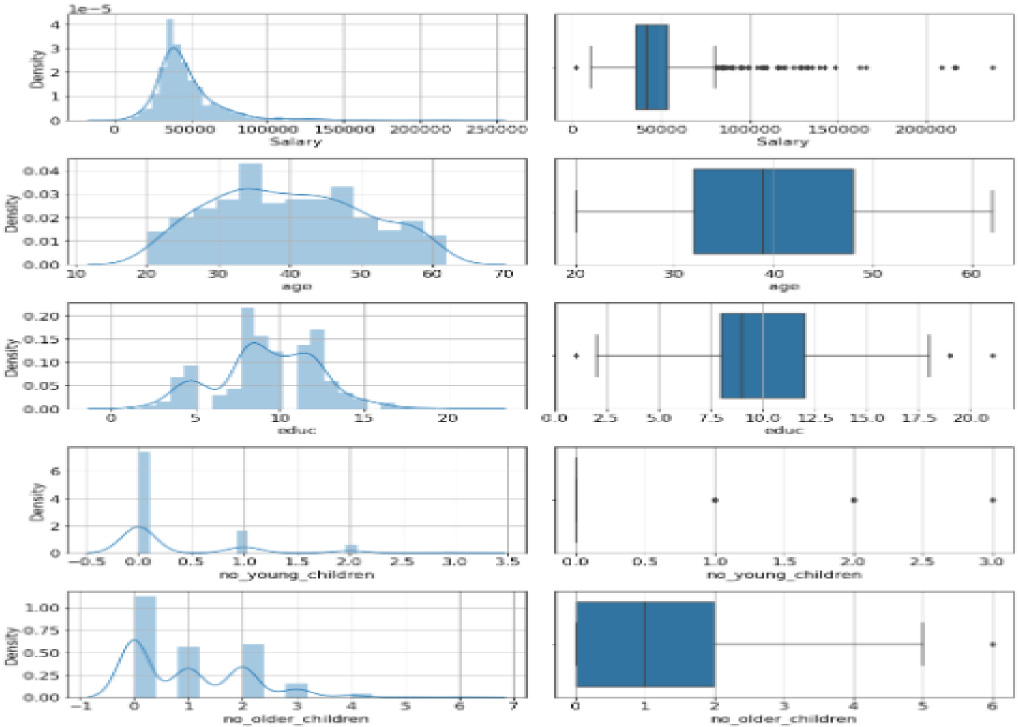


Figure3: Distribution & Box Plots for the variables

Observation: We observe that there are significant presence of outliers in variable "Salary", however there are minimum outliers in other variables like 'educ','no_young_children' & 'no_older_children'. There are no outliers in variable 'age'. For Interpretation purpose we would need to study the variables such as no. of young and elder children before we decide to treat the outliers.

Percentage of outliers-

	Outlier %
Holliday_Package	0.00
Salary	6.54
age	0.00
educ	0.46
no_young_children	23.74
no_older_children	0.23
foreign	24.77

Table3: Percentage of Outliers

Three columns have outlier values that too with more than 5% Percentage. In the current dataset, the proportion of outliers is very large, e.g. 20% for number of young children and foreigner.

‘Foreign’ variable is a categorical variable with values ‘yes’ and ‘no’ telling us if the employers are foreigner or not.

‘No_young_children’ is only telling us about employees have kids below 7yrs. And as per boxplot most of the data is very close to zero. We don’t see a point in treating this column

‘salary’ column has more than 6% outliers. Though the max salary value is high, we can say that the salary value of employee of higher age and higher education can be that high.

Looking at age and educ column we feel that salary is under justifiable range. Also, salary column will help us making recommendation to company for availing Holiday package. i.e. company can try and promote holiday package to employees who are earning more.

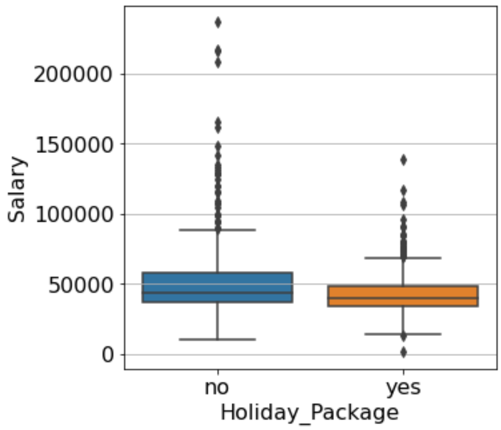
Hence, treating outlier in our opinion though its very important in Logistic regression model, we do not think these are Anomalies or wrong data captured.

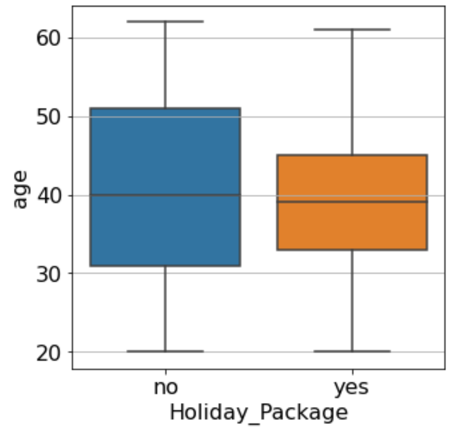
Note: We have tried to build model with treating outliers and its shows no significant difference hence, our analysis on no treating outlier in this dataset is correct.

Let’s have a look at the data more.

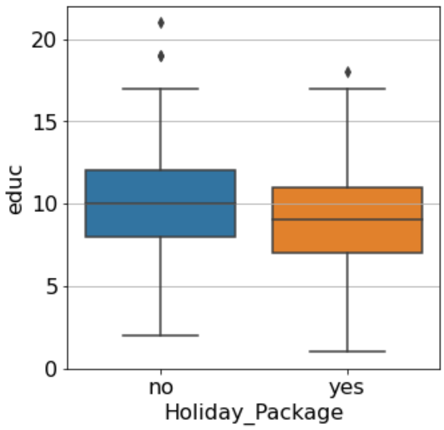
Bi-Variate Analysis with Target Variables

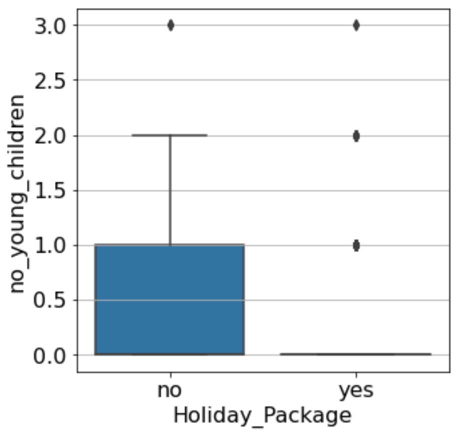
Holiday Package with Salary, age, educ, no_young_children and no_older_children





66





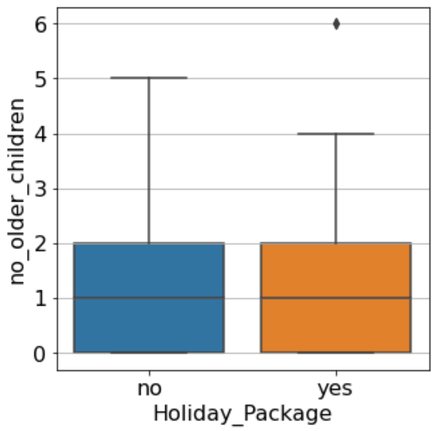


Figure4: Box Plots of Target variable vs Continuous variables

Observation:

- We observe that the Salary for employees opting for Holiday package and not opting for holiday package is very similar in nature. However, the distribution is fairly spread out for people not opting for holiday package.
- The distribution of Holiday_package with age is also very similar in nature. We can see that employee of middle age (34 to 45years) are going for holiday package as compared to older and younger employees.
- Education variable is also showing similar distribution pattern. This means education is likely not to be a variable influencing holiday packages for employees.
- We observe that employees with less year of education between 1-7 and higher educated employees are not opting for holiday package as compared to employee with formal education of 8-12 years.
- There is a significant difference in employees with younger children who are opting for holiday packages and employees who are not opting for holiday packages. Here, we see that employee with young children(below 7yrs) are not opting for holiday packages.
- Looking at variable no. of older children the distribution for opting or not opting holiday packages looks same. At this point we can say that this variable might not be a good predictor while creating the model.

Kurtosis & Skewness in Dataset –

	Kurtosis	Skewness
Holliday_Package	-1.98	0.16
Salary	15.85	3.10
age	-0.91	0.15
educ	0.01	-0.05
no_young_children	3.11	1.95
no_older_children	0.68	0.95
foreign	-0.63	1.17

Table4: Kurtosis & Skewness in Dataset

Inferences

- Continuous Variables age and no_older_children is having very low kurtosis values, near normal distributions.
- All variables have positive skewness except educ.
- All variable except age and educ are right skewed.
- Variable age and educ shows normal distribution.

Categorical Variable 'Foreign' with Salary, age, educ, no_young_children and no_older_children

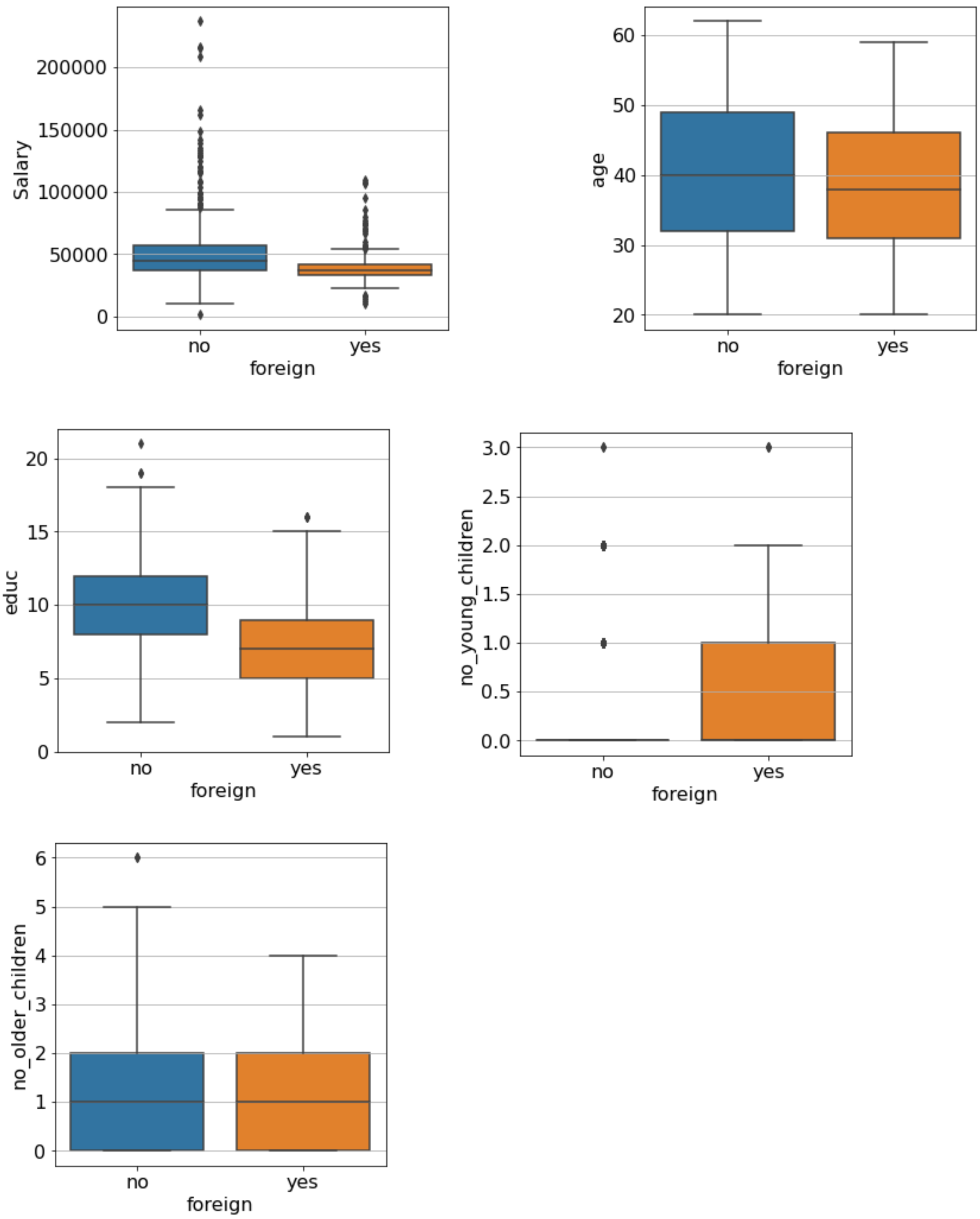


Figure5: Box Plots of Categorical variable 'foreign' vs Continuous variables

Observation-

Most of the non foreigner employees are having high salary. Majority of them are in the age bracket of 34-50. They are well educated enough. Only few on them have young children below 7yrs whereas rest are having older children most.

Multi-variate Analysis-

Checking Pairwise distribution of the continuous Variable

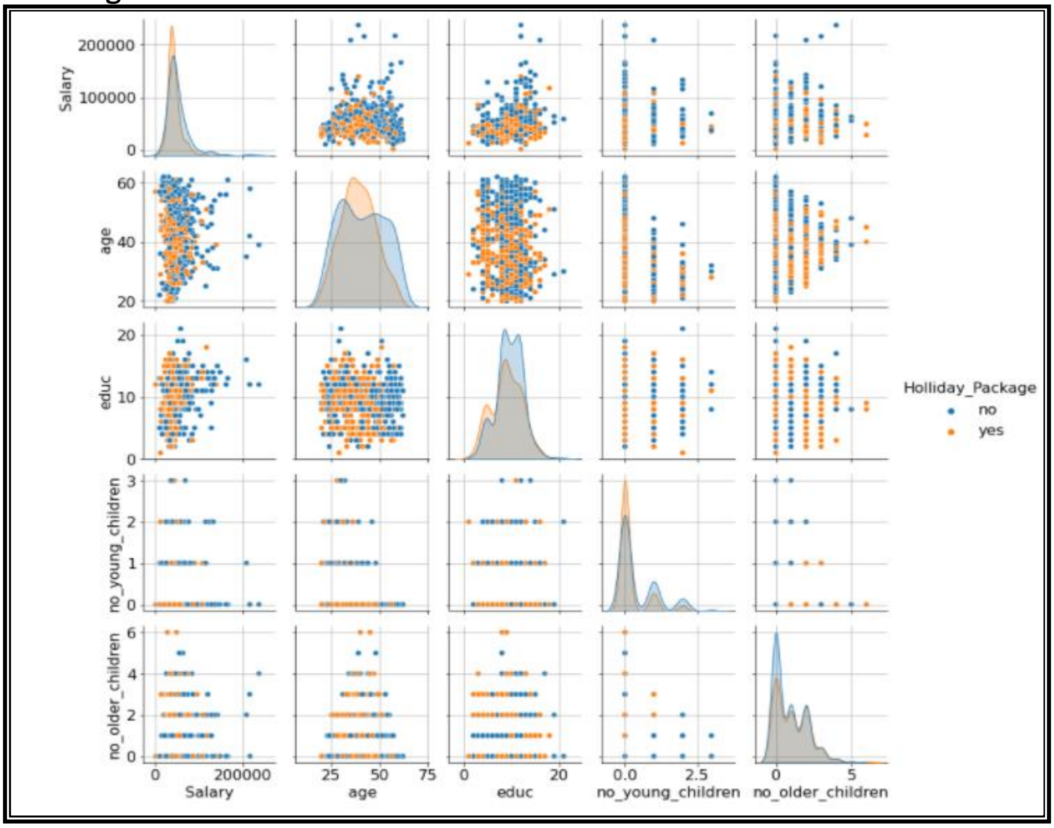


Figure 6: Pairplot of All Continuous Variable

The most important tool in data science tool box visualization is pairplot. In we do classification, we always look at the diagonals first. This pairplot is a square matrix, which means that no. of rows is equal to no. of columns. These rows are nothing but the different columns we have in our dataframe. So, it starts with Salary then age, then education, number of young children, number of older children. What we see in the diagonal is the kernel density estimates, its an estimated density distribution given the data we have. If we look at it all the orange and blue i.e 'yes' and 'no' are overlapping each other. Such classes are unable to discriminate between 2 classes. i.e. probability of opting for holiday package yes or not are almost equal such attributes are not good attributes from classification point of view. In, further correlation matrix and heatmap also we will see the same.

Analysing the relationship among continuous variables by **Correlation Heatmap**.

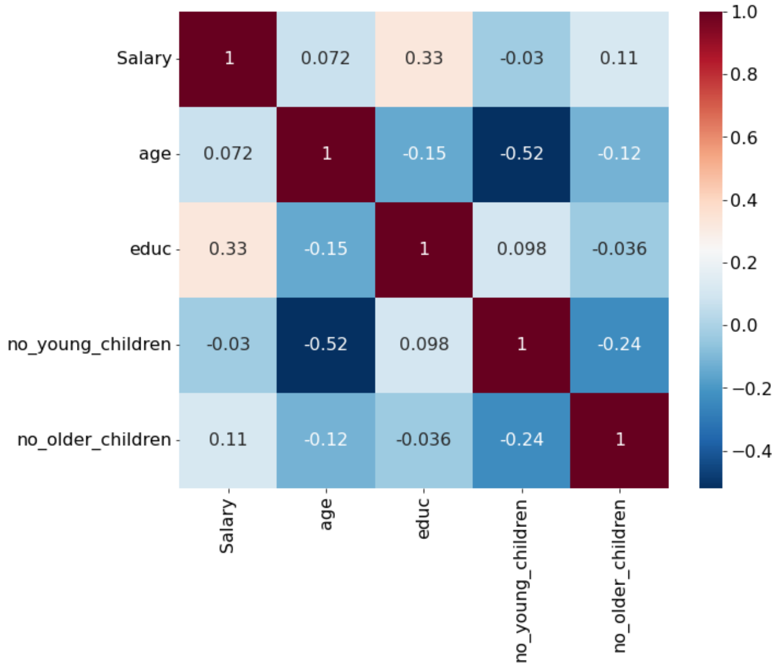


Figure 7: Correlation plot

Inference- There is no correlation among any of the independent variables. There are positive and negative correlation between the variables but they are very small.

Note -For practical purposes correlations in the range of[-0.4, 0.4] are not considered to be important.

In case of logistic regression, the response Y is always a nominal variable. Hence no correlation measure can be defined between the response and any of the predictors, be they continuous, nominal or ordinal.

1.2 Build various iterations of the Logistic Regression model using appropriate variable selection techniques for the full data. Compare values of model selection criteria for proposed models. Compare as many criteria as you feel are suitable.

Our Approach to Build Models

Descriptive Analysis

- Forward Selection
- Add columns and Check Adj Pseudo RSqaure
- Look at VIF values
- Remove column with high VIF value
- If no, multicollinearity is observed, remove columns based on p value

Let us first take care of the Categorical Variables 'Holiday_Package' and 'foreign' Here we are going to encode these variables into numerical values for the model creation.

'pd.Categorical' **can only take on only a limited, and usually fixed, number of possible values (categories)**. In contrast to statistical categorical variables, a Categorical might have an order, but numerical operations (additions, divisions, …) are not possible.

```
# Converting Categorical to Numerical Variable
for feature in holiday_data.columns:
    if holiday_data[feature].dtype == 'object':
        print("\n")
        print("feature:",feature)
        print(pd.Categorical(holiday_data[feature].unique()))
        print(pd.Categorical(holiday_data[feature].unique()).codes)

        holiday_data[feature] = pd.Categorical(holiday_data[feature]).codes
```

```


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


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

Let us check the values in the dataset to confirm the conversion.

```
holiday_data.head()
```

	Holiday_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

```
holiday_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 872 entries, 0 to 871
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Holiday_Package    872 non-null    int8
 1   Salary             872 non-null    int64
 2   age                872 non-null    int64
 3   educ               872 non-null    int64
 4   no_young_children  872 non-null    int64
 5   no_older_children  872 non-null    int64
 6   foreign            872 non-null    int8
dtypes: int64(5), int8(2)
memory usage: 35.9 KB
```

Table 5. Data head() and info() after encoding

Model Building- Using Forward Selection

Forward selection algorithm is a model building is an automated algorithm which selects one predictor at a time, conditional on which other predictors are already included in the model. This model is not expected to include the redundant predictors. Forward selection algorithm suggests a minimal set of predictors following certain optimality criteria. The forward selection algorithm is one of the most popular algorithms because of its easy interpretability.

As its mentioned above we selected one predictor at a time and built the model one by one by adding one-one variable in each iteration. (Refer to the python code notebook for the summary of the model)

By doing this we built 6 models and below is the Summary table of the **Final Model**

```
LG_model_6 = sm.logit(formula='Holiday_Package~Salary+age+no_young_children+foreign',data=holiday_data).fit()
LG_model_6.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.602653
         Iterations 6
```

Logit Regression Results

Dep. Variable:	Holiday_Package	No. Observations:	872
Model:	Logit	Df Residuals:	867
Method:	MLE	Df Model:	4
Date:	Tue, 01 Feb 2022	Pseudo R-squ.:	0.1265
Time:	11:06:22	Log-Likelihood:	-525.51
converged:	True	LL-Null:	-601.61
Covariance Type:	nonrobust	LLR p-value:	6.885e-32

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	2.6725	0.426	6.278	0.000	1.838	3.507
Salary	-1.664e-05	4.08e-06	-4.075	0.000	-2.46e-05	-8.64e-06
age	-0.0495	0.008	-5.843	0.000	-0.066	-0.033
no_young_children	-1.2946	0.169	-7.669	0.000	-1.625	-0.964
foreign	1.2124	0.183	6.634	0.000	0.854	1.571

Table 6: Summary Table of Final Model

There is no direct measure of goodness of fit for a logistic regression. For a linear regression the total sum of squares and the residual sum of squares are two well defined quantities. In case of logistic regression, these are not available. Hence it is difficult to quantify for a proposed logistic model, how much of the total variability in the data, it is able to explain. A few alternative quantifications similar, but not identical, to R2 statistic have been proposed for logistic regression model assessment. Two of them are more popular among them, namely **McFadden R2** and **Nagelkerke R2** .

$$\text{McFadden } R^2 = 1 - \frac{\log L_M}{\log L_N},$$

where the numerator is the model log-likelihood and the denominator is the log-likelihood of the null (intercept only) model. This quantity measures the improvement over the null model.

This statistic does not achieve 1 as the maximum value.

Nagelkerke R2 is also a function of the two log-likelihoods but has a complex form. This quantity has a range between 0 and 1.

```
# Calculate McFadden R-square
print('McFadden Psuedo R Squared (Model 1: Salary) =',round(LG_model_1.prsquared,2))
print('McFadden Psuedo R Squared (Model 2: Salary+age) =',round(LG_model_2.prsquared,2))
print('McFadden Psuedo R Squared (Model 3: Salary+age+educ) =',round(LG_model_3.prsquared,2))
print('McFadden Psuedo R Squared (Model 4: Salary+age+no_young_children) =',round(LG_model_4.prsquared,2))
print('McFadden Psuedo R Squared (Model 5: Salary+age+no_young_children+no_older_children) =',round(LG_model_5.prsquared,2))
print('McFadden Psuedo R Squared (Model 6: Salary+age+no_young_children+froeign) =',round(LG_model_6.prsquared,2))
```

```
McFadden Psuedo R Squared (Model 1: Salary) = 0.03
McFadden Psuedo R Squared (Model 2: Salary+age) = 0.03
McFadden Psuedo R Squared (Model 3: Salary+age+educ) = 0.04
McFadden Psuedo R Squared (Model 4: Salary+age+no_young_children) = 0.09
McFadden Psuedo R Squared (Model 5: Salary+age+no_young_children+no_older_children) = 0.09
McFadden Psuedo R Squared (Model 6: Salary+age+no_young_children+froeign) = 0.13
```

Table 7: McFadden R-square values of the Model

```
# Calculate Nagelkerke R-square
models=[LG_model_1,LG_model_2,LG_model_3,LG_model_4,LG_model_5,LG_model_6]
model_names= {LG_model_1: 'Model 1: only Salary',LG_model_2:'Model 2: Salary+age',LG_model_3: 'Model 3:Salary+age+educ',LG_mc
for i in models:
 ll_Intercept=i.llnull
 ll_Model = i.llf
 N= holiday_data.shape[0]
 num=(1- np.exp((ll_Model - ll_Intercept)*(-2/N)))
 den=( 1- np.exp((ll_Intercept)*(2/N)))
 nagelkerke_r2 = num/den
 print('Nagelkerke R Squared for {} ='.format(model_names[i]),round
(nagelkerke_r2,2))

Nagelkerke R Squared for Model 1: only Salary = 0.05
Nagelkerke R Squared for Model 2: Salary+age = 0.06
Nagelkerke R Squared for Model 3:Salary+age+educ = 0.06
Nagelkerke R Squared for Model 4: Salary+age+no_young_children = 0.15
Nagelkerke R Squared for Model 5: Salary+age+no_young_children+no_older_children = 0.15
Nagelkerke R Squared for Final Model 6: Salary+age+no_young_children+froeign = 0.21
```

Table 8: Nagelkerke R-square value of All Model

Since the range of the pseudo-R 2 is 0 to a number less than or equal to 1, the interpretation of the above values is not easy. Instead of taking them as an absolute number, it is better to look at their relative values among the models under consideration. Thus, it is clear that the model proposed as the Final Model has considerable **higher R2 values for both types**.

Final Model is Model 6 and the predictors are 'Salary', 'age', 'no_young_children', 'foreign'

In logistic regression the probability of the response being a success is predicted. To actually assign a binary value to the response, a threshold needs to be devised to partition the response space into success and failure.

Typically, the threshold is set at 50% level. If probability of success is 50% or above for a given combination of predictors, the value of response is taken to be 1, otherwise 0.

However, this threshold may be set at some other convenient level. Three measures of accuracy may be defined.

Let P be the total number of successes (positives) in the data and N be the total number of failures (negatives). If a success is predicted as success, it is an example of **True Positive (TP)**. If on the other hand a failure is predicted as failure, it is an example of **True Negative (TN)**.

In both cases, classification is correct. However, if a success is predicted as a failure, or if a failure is predicted as a success, they are misclassified.

Probability of misclassification = $FN+FP/n$, where n is the sample size.

For the perfect logistic regression, misclassification probability is 0; i.e. no observation would have been misclassified. This indicates overfit of the model and not to be recommended, since such a model will not have good predicting power.

A few other quantities are equally important.

TP (True Positive): We predicted positive (1) and its actual value is also Positive (1).

TN (True Negative): We predicted Negative (0) and its true (1).

False Positive (type1 error) (FP): We predicted positive (1) and its false (0).

False Negative (type2 error) (FN): We predicted Negative (0) and its false (0).

Precision = $TP/\ TP+FP = TP\ /P$, i.e. among all the successes (positives) in the data, how many are identified as positive by the logistic regression.

Specificity = $TN/TN+FP = TN/N$, i.e. among all failures (negatives) in the data, how many are actually identified as negative by the logistic regression

Sensitivity or Recall = $TP\ /TP+FN$, i.e. among all the predicted successes, how many are actually success.

The **F-score** of the model is defined as $2(Precision*Recall)\ Precision+Recall$. F is between 0 and 1, and the closer is it to 1, the better is the model Among two competing logistic regressions, the one that maximizes all the accuracy measures, is the one of choice.

However, it may not be possible to maximize all criteria simultaneously.

	predicted_prob	
Label	0	1
Holiday_Package		
0	353	118
1	173	228

Table 9: Confusion Matrix of Final Model 6 at threshold level 0.5

If the cut-off threshold is set at 0.5, then misclassification probability is (118 + 173)/872 = 33.3%

Therefore, **accuracy** of the model is 1 – 0.3337 = 66.6%

Recall = 228 / (228 + 173) = 57%

Specificity = 353 / (353 + 118) = 67%

Precision = 228 / (228 + 118) = 66%

F-score = 2*0.569*0.659/(0.659+0.569) = 0.62 = 62%

accuracy	Precision	Recall	F-score
66.60%	66%	57%	62%

Table 10: Classification Matrix of Final Model 6

It is clear from the above statistics that precision of the model is not high. Among all the Holiday packages in the data, the model is able to correctly predict only 57% of the cases.

On the other hand, specificity 67% indicates that, among the non-holiday package, the model is able to correctly identify 67%.

Recall that once the probability of success is estimated through the logistic regression, the partition into two groups, success and failure, is controlled by placing the cut-off threshold. Currently it is set at 0.5. Suppose to improve precision, it is decided to set at 0.35

	predicted_prob	
Label	0	1
Holiday_Package		
0	216	255
1	62	339

Confusion Matrix of Final Model 6 at threshold level 0.35

, then misclassification probability is (255 + 62)/872 = 36.35%

Therefore, **accuracy** of the model is 1 – 0.4266 = 63.65%

Recall = 339 / (339 + 62) = 84.5%

Specificity = 216/ (216 + 255) = 45.86%

Precision = 339 / (339 + 255) = 57.07%

F-score = 2*0.845*0.571/(0.845+0.571) = 0.6815 = 68.15%

Note that, by changing the threshold value, as precision improves significantly, both specificity and recall values decreases by large amount. Accuracy and F-score changes marginally, the former increases and the latter decreases.

Since these measures are a function of the threshold, often the impact of the whole range of thresholds is investigated through the Receiver Operating Characteristic (ROC) curve. The curve is typically obtained by plotting 1 – specificity (False Positive Rate, FPR) on the x-axis and sensitivity (True Positive Rate, TPR) on the y-axis. However, there may be alternative representations of the same.

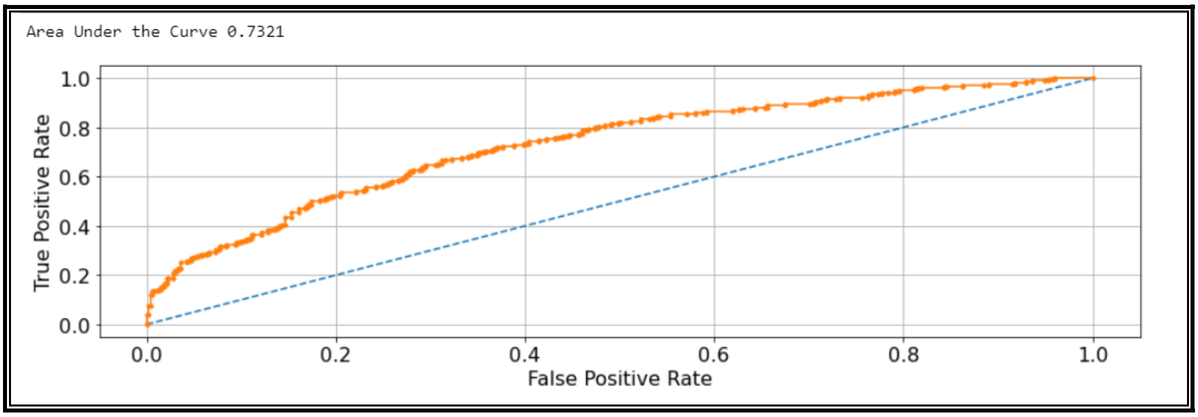


Figure 8: AUC-ROC of Final Model

Area under ROC curve is a performance measurement for any classification problem (not necessarily for logistic regression only) at various thresholds points. ROC is a probability curve and AUC represents the classification model's ability to separate the two classes. **The higher the AUC, the more powerful is the model to predict true class membership.**

The observations from Fig. 8 are:

1. There is a trade-off between sensitivity and specificity; if one increases, then the other decreases. 2. The closer the curve comes to the top left corner of the probability space, the more area it covers; hence the better is the model

3. Any curve below the diagonal line is worse than a random allocation mechanism.

Note: The model with higher AUC-ROC is expected to have better discretionary power.

1.3 Split the data into training (70%) and test (30%). Build the various iterations of the Logistic Regression models on the training data and use those models to predict on the test data using appropriate model evaluation metrics.

After the EDA and all adjustments and transformations were performed on the full data, it was randomly split into training and test sets in 70:30 ratio.

```
from sklearn.model_selection import train_test_split

Train,Test = train_test_split(holiday_data,test_size=0.3,random_state=1,stratify=holiday_data['Holiday_Package'])

Train.shape

(610, 7)

Test.shape

(262, 7)

print(Train['Holiday_Package'].value_counts(normalize=True),'\n')
print(Test['Holiday_Package'].value_counts(normalize=True))

0    0.539344
1    0.460656
Name: Holiday_Package, dtype: float64

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

Note: It is always a good idea to check that the success proportion of response is similar in both training and test data.

Model Building-Iteration 1

We are going to encode the target variable and build the logit Model.

In python codebook we have built 2 models (refer to the code book) Depending upon the p_values we saw that the predictor which were above significant level of 0.05 i.e. 'educ' and 'no_older_children' was dropped and made the Final model with predictor below significant level. Below is the Summary table for the same.

```
formula='Holiday_Package~Salary+age+no_young_children+foreign'

model2=sm.logit(formula,data=Train).fit()
print(model2.summary())

Optimization terminated successfully.
         Current function value: 0.601787
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:        Holiday_Package   No. Observations:                  610
Model:                          Logit   Df Residuals:                      605
Method:                           MLE   Df Model:                            4
Date:                Wed, 02 Feb 2022   Pseudo R-squ.:                  0.1279
Time:                        13:37:04   Log-Likelihood:                -367.09
converged:                       True   LL-Null:                       -420.93
Covariance Type:            nonrobust   LLR p-value:                 2.276e-22
=====================================================================================
                        coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------------
Intercept             2.9301      0.500      5.865      0.000       1.951       3.909
Salary            -1.421e-05   4.56e-06     -3.118      0.002   -2.31e-05   -5.28e-06
age                  -0.0579      0.010     -5.633      0.000      -0.078      -0.038
no_young_children    -1.2893      0.189     -6.833      0.000      -1.659      -0.920
foreign               1.1031      0.213      5.176      0.000       0.685       1.521
=====================================================================================
```

Table 11: Summary table of Final Model of Train dataset-Iteration1

Once a we built a satisfactory final model on the training data, we then went ahead and check the estimated accuracy on both training and test data.

The confusion matrix is an N x N table (where N is the number of classes) that contains the number of correct and incorrect predictions of the classification model.

```
Confusion Matrix on Train Set

                predicted_prob
Label                        0    1
Holiday_Package
0                          242   87
1                          121  160
```

Table 12: Confusion Matrix of Train Set-Iteration 1

Precision=160/(160+87)=64.77% i.e. 65%

Recall=160/(160+121)=56.93% i.e. 57%

F1Score=60.59% i.e. 61%

While, we can calculate this let's match it with our Classification Report below:

Classification Report on Train Set

	precision	recall	f1-score	support
0	0.67	0.74	0.70	329
1	0.65	0.57	0.61	281
accuracy			0.66	610
macro avg	0.66	0.65	0.65	610
weighted avg	0.66	0.66	0.66	610

Table 13: Classification Report on Train Set-Iteration 1

Observation:

The classification report visualizer displays the **precision, recall, F1, and support scores for the model**.

Accuracy- The model is showing accuracy of 66%.

Precision — the Train model is showing **65%** ***of our predictions are going to be correct.***

Recall – the model is telling 57% percent of the positive cases were a match i.e. our model could catch 51% of all positive (opting for holiday packages) instances.

F1 score- the model is telling 61% of positive prediction were correct.

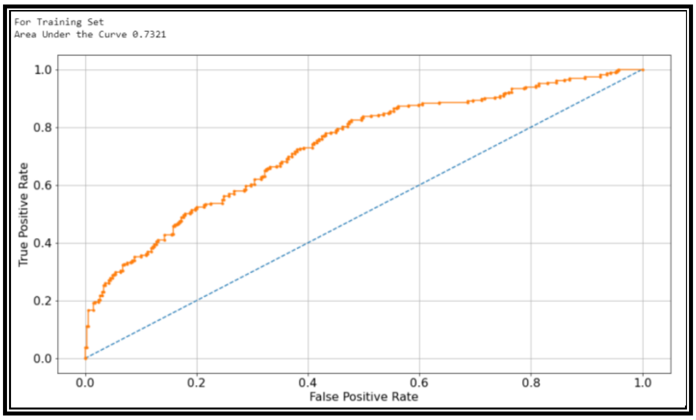


Figure 9: AUC-ROC curve for Train Set-Iteration 1

Once a model is decided upon, the same is applied to Test data. Here the model is NOT developed independently, but the same parameter estimates found from Training data are used. The goal is pure prediction accuracy; i.e., when new observation vectors are available, how accurately the model is expected to predict the probability of employees opting for Holiday packages. No model estimate is done for Test data.

Confusion Matrix on Test Set

	predicted_prob	
Label	0	1
Holiday_Package		
0	110	32
1	53	67

Table 15: Confusion Matric of Final Model of Test Set-Iteration 1

Precision=67/(67+32)=67.7% i.e. 68%

Recall=67/(67+53)=55.8% i.e. 56%

Recall=61.4% i.e. 61%

While, we can calculate this let's match it with our Classification Report below:

Classification Report on Test Set

	precision	recall	f1-score	support
0	0.67	0.77	0.72	142
1	0.68	0.56	0.61	120
accuracy			0.68	262
macro avg	0.68	0.67	0.67	262
weighted avg	0.68	0.68	0.67	262

Table 16: Classification Report on Final Model Test Set-Iteration 1

Observation: The classification report visualizer displays the **precision, recall, F1, and support scores for the model**.

Accuracy – the model accuracy is 68%.

Precision — the model is showing **68%** ***of our predictions were correct.***

Recall – the model is telling 56% percent of the positive cases were a match i.e. our model could catch 56% of all positive (opting for holiday packages) instances.

F1 score- the model is telling 61% of positive prediction were correct.

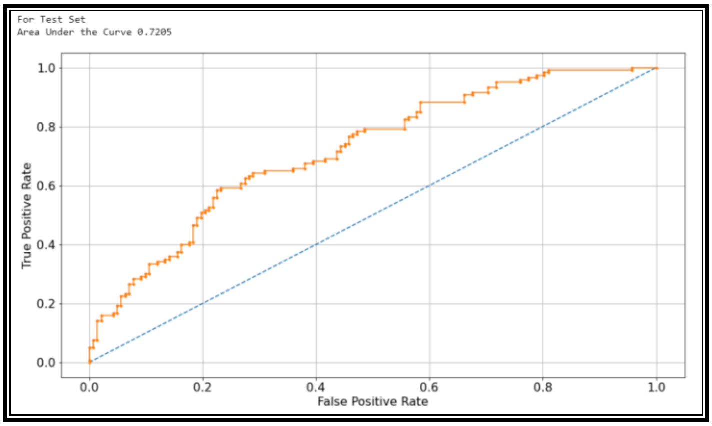


Figure 10: AUC-ROC plot for Test Set

Note: Accuracy of Train data Set: 66%

Accuracy of Test data set: 68%

As we compare Accuracy of Train and Test Data set, we see that accuracy of Training and Test data are very close. That is a support for consistency of the model building procedure.

Observation: ROC is a probability curve and AUC represents the classification model's ability to separate the two classes.

Comparing the AUC-ROC of final model in 1.2 (Fig6)and AUC-ROC of final model of Train(fig 9) and Test set(fig 10). We see no difference in the AUC value and ROC curve.

As we know the higher the AUC, the more powerful is the model to predict true class membership. Here, with 72% of AUC we can say that our model is predicting true class at 72% level.

Also, the important factor which will help the company to focus on particular employees are 'Salary', 'age', 'number of young childrens' and whether the employee is 'foreigner or not' as per the model. Looking at the coefficient value 'Foreign' variable is the only important factor that has emerged as a strong predictor.

In Logistic Regression we often hear a term "**Odd-Ratio**"

What is odd ratio? The odds ratio compares the odds of two events. The odds of an event are the probability that the event occurs(employee opting for holiday package) divided by the probability that the event does not occur(employee not opting for holiday package)

```
params = model2.params
conf = model2.conf_int()
conf['Coef']=np.round(model2.params.values,7)
conf['Odds Ratio'] = params
conf.columns = ['5%', '95%', 'Odds Ratio', 'Coef']
conf[['5%', '95%', 'Odds Ratio']]=np.exp(conf[['5%', '95%', 'Odds Ratio']])
conf
```

	5%	95%	Odds Ratio	Coef
Intercept	7.035222	49.860939	18.729196	2.930084
Salary	0.999977	0.999995	0.999986	-0.000014
age	0.924875	0.962924	0.943708	-0.057939
no_young_children	0.190307	0.398713	0.275459	-1.289315
foreign	1.984652	4.576034	3.013609	1.103138

- The coefficients and the odds ratios represent **the effect of each independent variable controlling for all of the other independent variables in the model** and each coefficient can be tested for significance.
- Odds ratios that are greater than 1 indicate that employees opting for holiday packages is more likely to occur as the predictor increases. Odds ratios that are less than 1 indicate that employee opting for holiday package is less likely to occur as the predictor increases.
- the important thing to remember about the odds ratio is that an odds ratio greater than 1 is a positive association (i.e., higher number for the predictor means group 1 in the outcome), and an odds ratio less than 1 is negative association (i.e., higher number for the predictor means group 0 in the outcome).
- Hence, the above figure says that for each increase in foreign value, the odds for employees opting for holiday package increases by a factor of 1.

<u>Model Building-Iteration 2</u>

In second iteration of model building, we took 3 models from question1.2 and built the model using Solver.

Solver-Provides option to choose solver algorithm for optimization. Usually default solver works great in most situations and there are suggestions for specific occasions such as: classification problems with large or very large datasets.

We can always monitor how solver works on test and train data by comparing different solver functions. This can help us to understand the fineness of different solvers.

Here, we are using ‘newton-cg’ solver which calculates Hessain explicitly which can be computationally expensive in high dimensions.

Penalty: Defines penalization norms. Certain solver objects support only specific penalization parameters so that should be taken into consideration.

‘none:’ Penalty regularization won’t be applied.

```
from sklearn.linear_model import LogisticRegression
```

```
LR = LogisticRegression(solver='newton-cg',penalty='none')
```

By taking variables of 3 models from question 1.2, we have First, import the Logistic Regression module and create a Logistic Regression classifier object using LogisticRegression() function.

Then, fit your model on the train set using fit() and perform prediction on the test set using predict().

Model Evaluation using Confusion Matrix using Heatmap:

A confusion matrix is a table that is used to evaluate the performance of a classification model. Here we can visualize the performance of an algorithm. The fundamental of a confusion matrix is the number of correct and incorrect predictions are summed up class-wise.

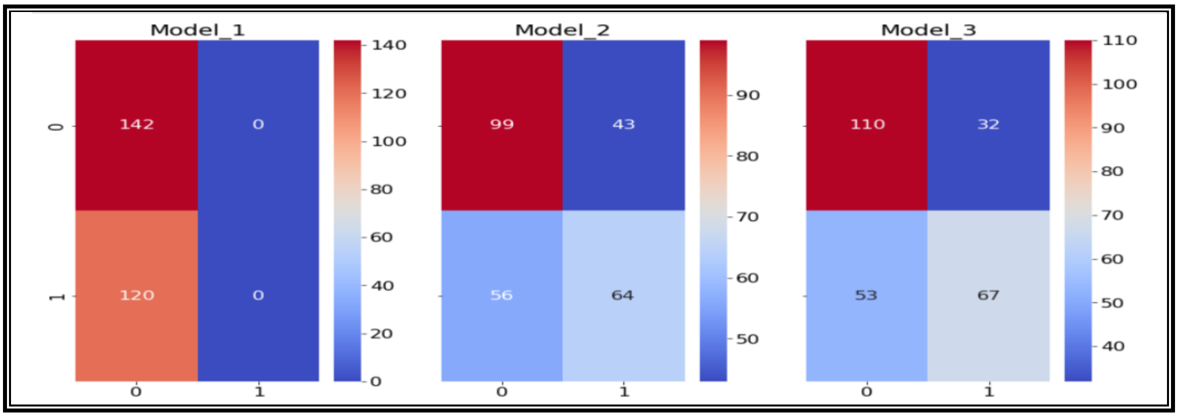


Figure 11: Confusion Matrix of three Model for Comparison-Iteration 2

Here, we can see the confusion matrix in the form of the array object. The dimension of this matrix is 2*2 because this model is binary classification. We have two classes 0 and 1. Diagonal values represent accurate predictions, while non-diagonal elements are inaccurate predictions. In the above output, in Model 3-110 and 67 are actual predictions, and 53 and 32 are incorrect predictions.

Lets, go ahead and evaluate these models using Model evaluation metrics such as accuracy, precision, recall and f1 scores.

Classification Report

Model 1

	precision	recall	f1-score	support
0	0.54	1.00	0.70	142
1	0.00	0.00	0.00	120
accuracy			0.54	262
macro avg	0.27	0.50	0.35	262
weighted avg	0.29	0.54	0.38	262

Model 2

	precision	recall	f1-score	support
0	0.64	0.70	0.67	142
1	0.60	0.53	0.56	120
accuracy			0.62	262
macro avg	0.62	0.62	0.62	262
weighted avg	0.62	0.62	0.62	262

Model 3

	precision	recall	f1-score	support
0	0.67	0.77	0.72	142
1	0.68	0.56	0.61	120
accuracy			0.68	262
macro avg	0.68	0.67	0.67	262
weighted avg	0.68	0.68	0.67	262

Table 17: Classification Metrics of 3 Model for Comparison-Iteration 2

ROC Curve

Receiver Operating Characteristic (ROC) curve is a plot of the true positive rate against the false positive rate. It shows the trade-off between sensitivity and specificity.

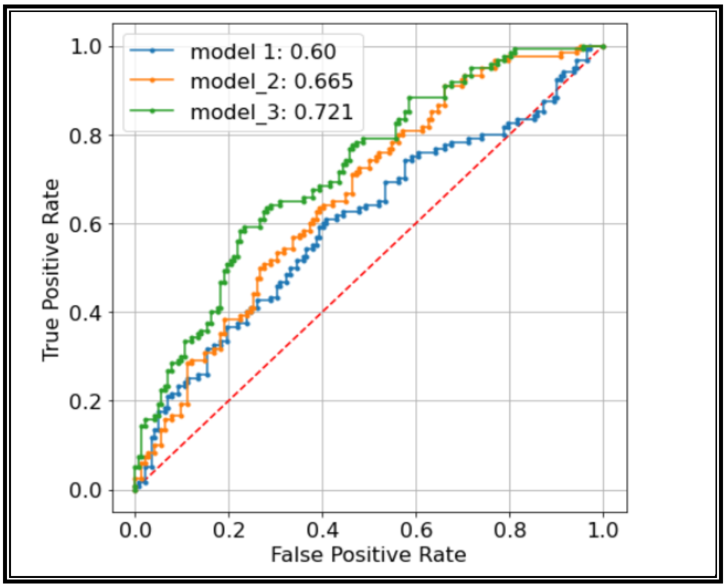


Figure 12 : AUC-ROC plot for 3 models-Iteration2

AUC score for the case is 1) model 1-0.60 2) model 2-0.665 and 3) model 3-0.721

Conclusion: With above Evaluation and Visualization we came to conclusion that Model 3 with predictors as 'Salary', 'age', 'no_young_children' and 'foreign' are giving best performance than others. Hence this is our Final Model. Among the predictors 'foreign' has emerged as a strong predictor with positive coefficient.

Note: Model build in 1.2 and 1.3 are very similar as they are using same variables/predictors for the business model. Comparing the classification metrics both the model doesn't show any difference in accuracy, precision, recall and f1scores.

As well as if we compare the AUC-ROC curve, the plot and the auc values are same. Hence, final model recommended as above.

Insight and Recommendation:

We were given a business problem where we were asked to predict the important factor on which company will focus on particular employees to sell their Holiday Packages. We did Logistic Regression Analysis on this problem to predict the important factor.

In our EDA analysis it clearly indicates certain criteria, if employee is not a foreigner and employee not having young children, chances of opting for holiday packages is good.

Employee having salary high salary are not opting for holiday packages.

Those employees who earns less than 50k have opted more for holiday packages.

Employees who are of age more than 50 are not taking holiday packages.

Whereas employee of middle age 30 to 50 with salary around 50k have opted more for holiday packages.

Employees having older children are not taking holiday packages.

There were no outliers present in age variable. Most of the variables were having very similar distribution for opting and not opting for holiday packages.

When we looked at the coefficient values of all the variable, we know that predictor variable shows the effect of a one-unit change in the predictor variable.

While model building and looking at the summary table we found that surprisingly Salary and age variable did not turn out to be an important predictor for our model. As they were showing negative coefficient values. Variable Foreign had positive coefficient value showing statistical significance on the target variable.

Outliers: As we have clarified in 1.1 we are not recommending treating outliers for our correct dataset. (Refer, to python code) Still we tried this method of treating the outliers and building the model. And we observed that we did not see any evidence that treating outliers will help us getting a good fit of the model. Hence, we conclude that treating outliers in our present data is not required at all.

Scaling: Its an advantage, logistic regression doesn’t require scaling. Logistic regression provides a probability score for observations. Because its very efficient and straightforward in nature, this doesn’t require high computation power, easy to implement, easy interpretable, used widely by data analyst and scientist.

Insight of Model Performance: While evaluating Model performances on metrics we saw that accuracy score of train and test data is not giving a best score but there is not much of difference in their scores. This happens, as we dropout variables, during training. Thus, training accuracy suffers. We also feel that to achieve high model performance we should have more data/more features/more insights from the company.

Recommendation:

- As we analysed in our Final Model that variable ‘Foreign’ is a strong predictor.
- Company should design holiday packages depending on employee being foreigner or not.
- Employee being foreigner might be interested in explorer holiday package more and company might get more conversion from these employees.
- To sell more Holiday Packages to employees above 50 ages, company should come up with exquisite Holiday Packages like
- -Pilgrimage holiday packages-can be chosen based on religious belief.
- -wellness/medical holiday packages(the favourite destination for medical holiday package is Kerala which offers Ayurveda as well as Allopathy packages).
- -Eco Holiday Packages-focused on beauty of underdeveloped, natural and culturally sensitive destination.

- Company should try and design Cruise holiday package for Employees with higher Salary. As they will look for Leisure kind of holiday packages.
- Company should promote holiday packages on employee Anniversaries for Family Holidays.
- To target employee with younger kids and older kids might be difficult dues to children's academic sessions. But company can specially curate summer and winter fun-family holiday packages to get more conversion.
- Company can also give them discounts and surprise holiday package coupons to avail their packages.

Thank you