KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY (KIIT)
Deemed to be University U/S 3 of the UGC Act, 1956

DIAMOND PRICE PREDICITON

BY -

SABYASACHI GHOSH

TABLE OF CONTENT

INTRODUCTION

Diamond Price Analysis using Python aims to explore the factors influencing diamond prices and build a predictive model to estimate diamond prices based on various features. This project is essential for stakeholders in the jewellery industry to understand the market dynamics and make informed decisions regarding pricing strategies and inventory management.

DIAMOND PRICE ANALYSIS USING PYTHON

The dataset used in this project is sourced from Kaggle. It contains information on various features such as carat weight, cut, colour, clarity, and price for thousands of diamonds. Data preprocessing involved handling missing values, encoding categorical variables, and scaling numerical features to prepare the data for analysis.
You can download the dataset from: https://www.kaggle.com/shivam2503/diamonds . And python libraries: Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn.
Now we will start the task of Diamond Price Analysis by importing the necessary Python libraries and the dataset:

ANALYSIS:

[2]: #importing all the libraies

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split from
sklearn.tree import DecisionTreeRegressor from sklearn.metrics
import mean_squared_error from sklearn import metrics
from sklearn.metrics import mean_squared_error,mean_absolute_error
import plotly.express as px
import plotly.graph_objects as go
```

[3]:
```
# taking the data set input
data = pd.read_csv("C:\\Users\\KIIT\\Desktop\\diamondss.csv")
data.head(5)
```

```
[3]:   Unnamed: 0  carat    cut color clarity  depth  table  price     x     y  \
 0           1   0.23    Ideal    E     SI2   61.5   55.0    326  3.95  3.98
 1           2   0.21  Premium    E     SI1   59.8   61.0    326  3.89  3.84
 2           3   0.23     Good    E     VS1   56.9   65.0    327  4.05  4.07
 3           4   0.29  Premium    I     VS2   62.4   58.0    334  4.20  4.23
 4           5   0.31     Good    J     SI2   63.3   58.0    335  4.34  4.35
    0  2.43
    1  2.31
```

```
2   2.31
3   2.63
4   2.75
```

[4]:
```
# Shape
data.shape
```

[4]: (53940, 11)

Data Pre-processing :

- ✓ Steps involved in Data Preprocessing
- ✓ Data Cleaning
- ✓ Identifying and removing the outliers
- ✓ Encoding categorical variables

[5]:
```
# Removing the first column because it is the same as index
data.drop(["Unnamed: 0"], axis = 1) #Axis 1 for selecting columns
data.describe()
```

[5]:

	Unnamed: 0	carat	depth	table	price
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000
mean	26970.500000	0.797940	61.749405	57.457184	3932.799722
std	15571.281097	0.474011	1.432621	2.234491	3989.439738
min	1.000000	0.200000	43.000000	43.000000	326.000000
25%	13485.750000	0.400000	61.000000	56.000000	950.000000
50%	26970.500000	0.700000	61.800000	57.000000	2401.000000
75%	40455.250000	1.040000	62.500000	59.000000	5324.250000
max	53940.000000	5.010000	79.000000	95.000000	18823.000000

	x	y	z
count	53940.000000	53940.000000	53940.000000
mean	5.731157	5.734526	3.538734
std	1.121761	1.142135	0.705699
min	0.000000	0.000000	0.000000
25%	4.710000	4.720000	2.910000
50%	5.700000	5.710000	3.530000
75%	6.540000	6.540000	4.040000
max	10.740000	58.900000	31.800000

Now removing the “x”,“y”,“z”, values which are zero because they are faulty values

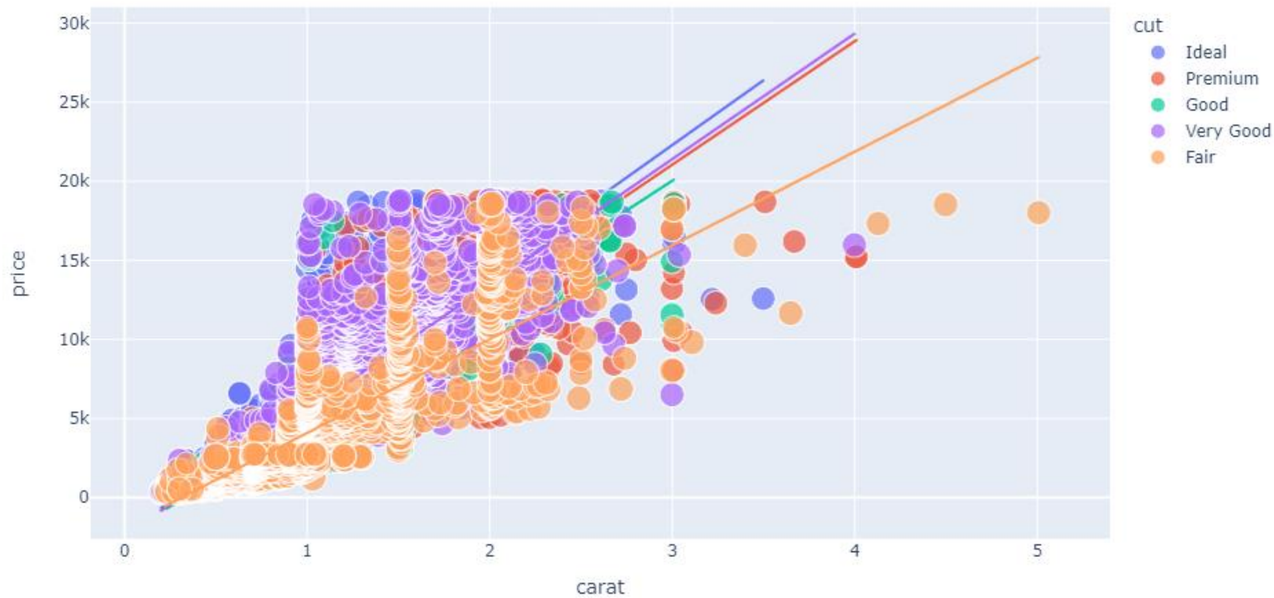
[6]:
```
data = data.drop(data[data["x"]==0].index)
data = data.drop(data[data["y"]==0].index)
data = data.drop(data[data["z"]==0].index)
data.shape     #data shape after removing the "x","y","z" values having 0
```

[6]: (53920, 11)

We lost 20 data points by deleting the dimensionless(2D or 1D) diamonds.

Exploratory Data analysis

[7]:
```
figure = px.scatter(data_frame = data, x = "carat", y= "price",size =
"depth",color = "cut",trendline = "ols")
figure.show()
```



The above figure concludes two features of diamonds:

1. Premium cut diamond are relatively larger than other diamonds.

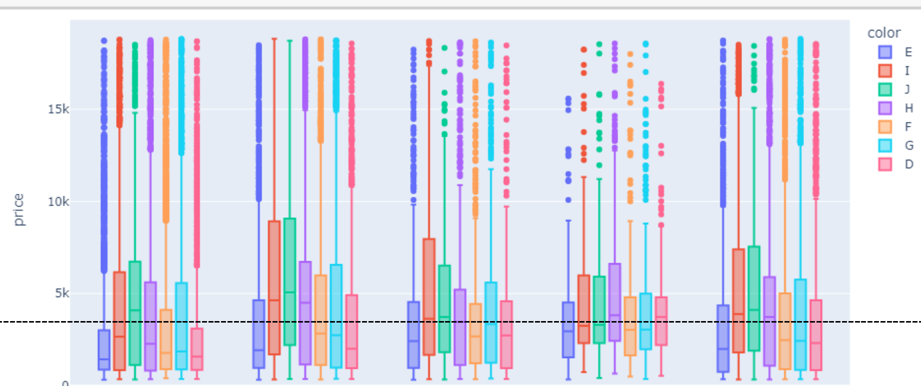
2. There's linear relationship between the size of all types of diamonds and their prices

Now let’s have a look at the price of all the types of all the types of diamonds based on their colour:

[8]:
```
fig = px.box(data, x = "cut", y = "price",color = "color")
fig.show()
```

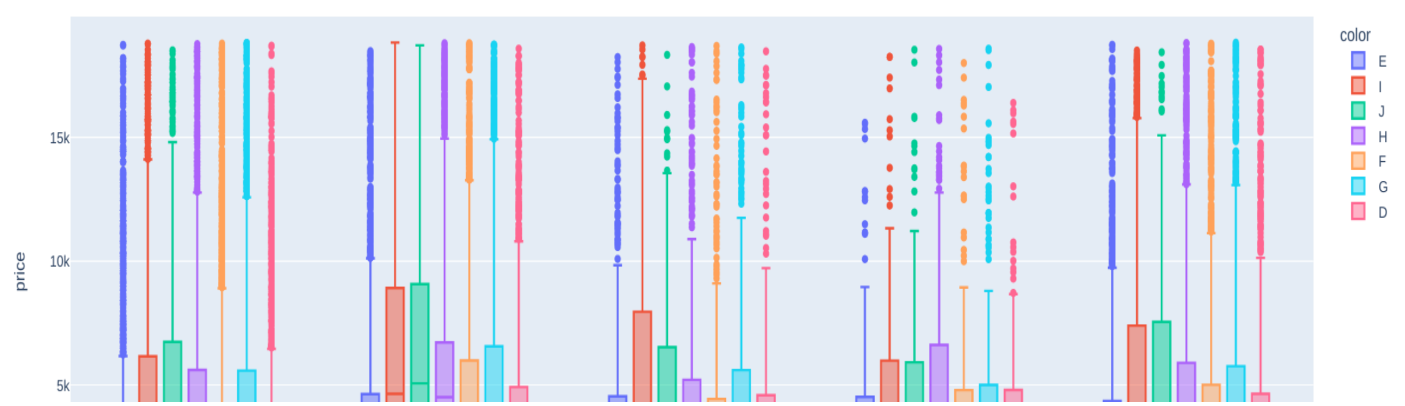
Now let’s have a look at the price of all the types of diamonds based on their clarity :

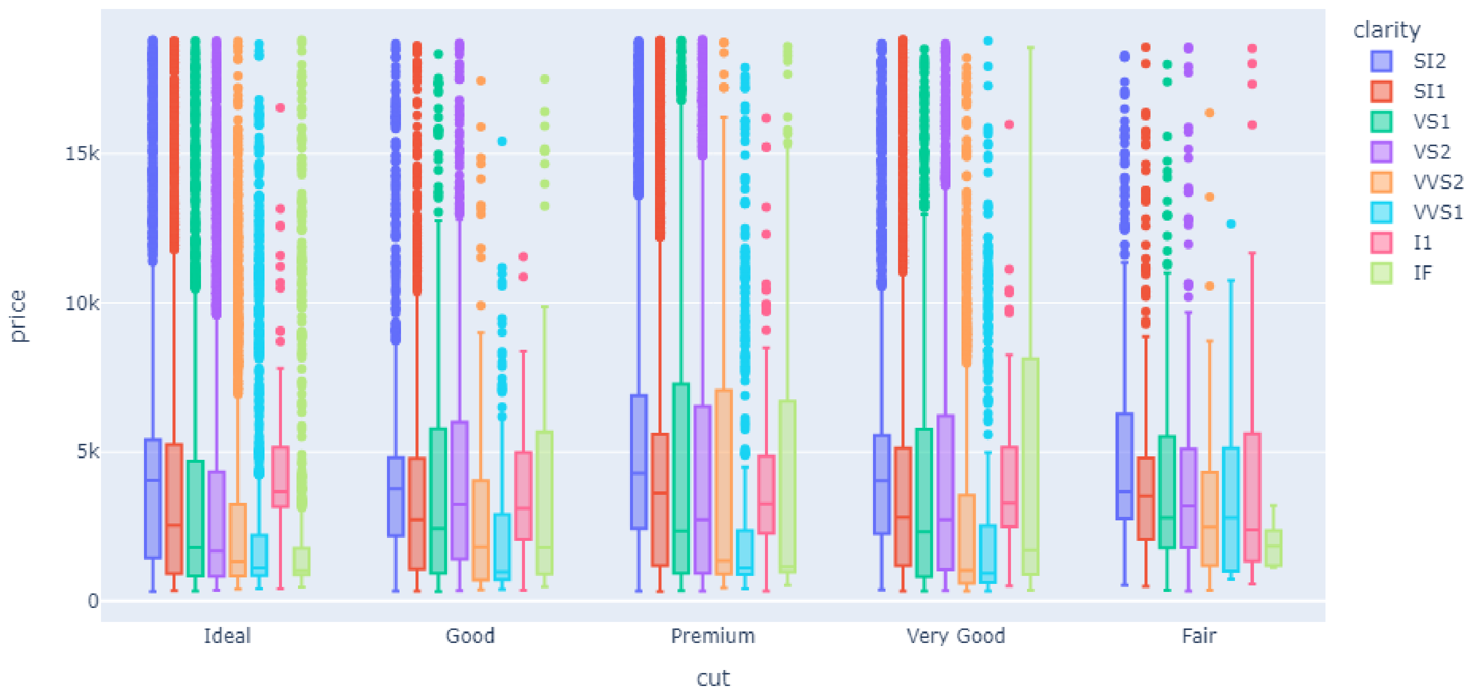
[10]:
```
fig = px.box(data, x = "cut", y = "price", color ="clarity")
fig.show()
```



Now let's have a look at the price of all the types of diamonds based on their clarity:

```
In [8]: fig= px.box(data, x = "cut",y = "price",color = "color")
        fig.show()
```





Data Pre-processing -2

```
[11]: data['cut'] = data['cut'].map({'Ideal':5,'Premium': 4, 'Very Good' :3,'Good':
      ↪2,'Fair':1})
      data['color'] = data['color'].map({'D':7,'E':6,'F':5,'G':4,'H':3,'I':2,'J':1})
      data['clarity']=data['clarity'].map({'IF':8,'VVS1':7,'VVS2':6,'VS1':5,'VS2':
      ↪4,'SI1':3,'ST2':2,'IF':1})
```

Correlation

Now let's have a look at the correlation between diamonds prices and other features in the dataset:

[12]:
```
data.corr()
```

carat	0.181646	0.921592	0.977779	0.953991	0.961048
cut	-0.433306	-0.053491	-0.126232	-0.122181	-0.150647
color	-0.026481	-0.172431	-0.270671	-0.263915	-0.270011
clarity	-0.060149	-0.063147	-0.167572	-0.164300	-0.165640
depth	-0.295733	-0.010729	-0.025017	-0.029069	0.095023
table	1.000000	0.127245	0.196097	0.184493	0.152483
price	0.127245	1.000000	0.887231	0.867864	0.868206

[12]:

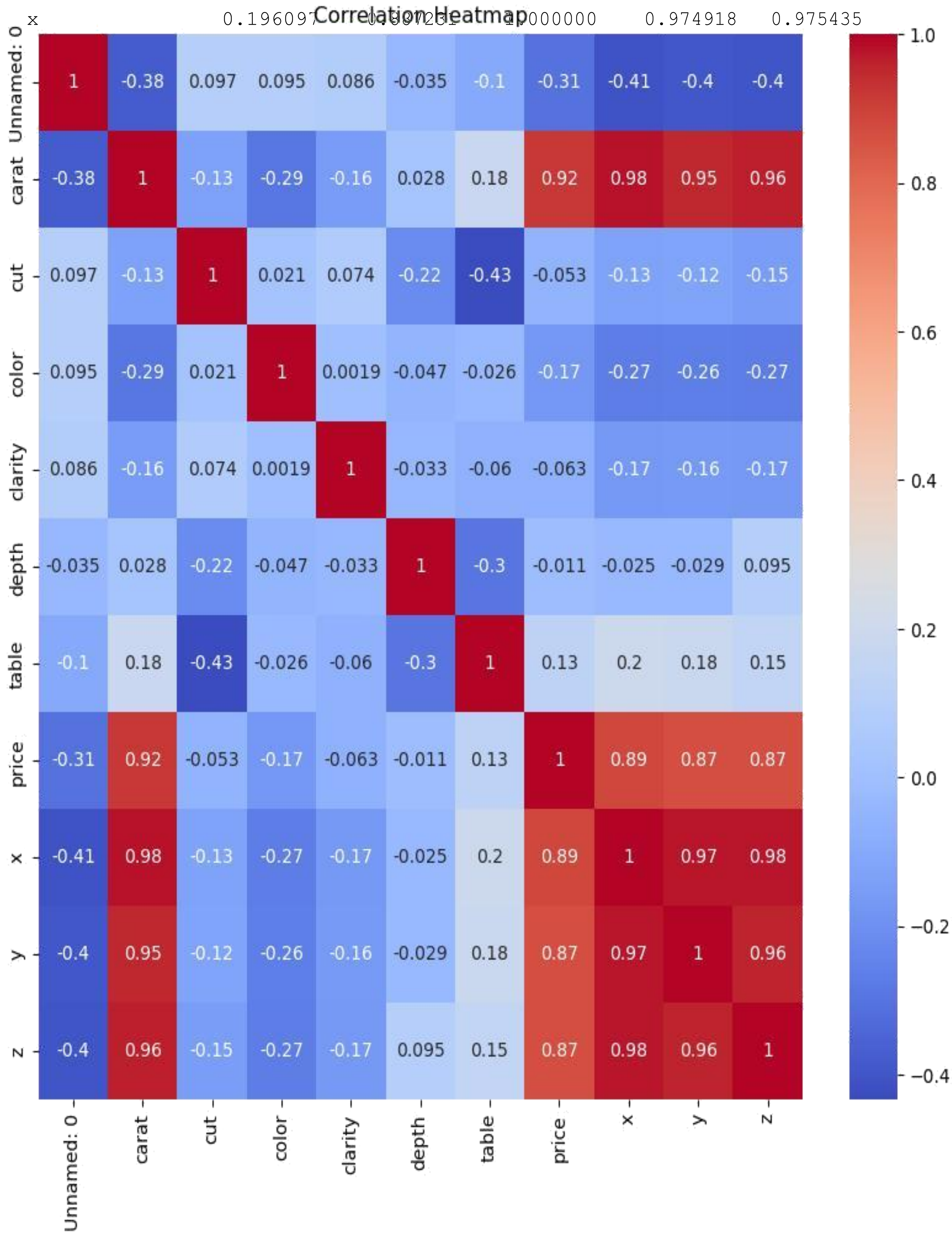
	Unnamed: 0	carat	cut	color	clarity	depth
Unnamed: 0	1.000000	-0.378173	0.096584	0.095062	0.085568	-0.035058
carat	-0.378173	1.000000	-0.134953	-0.291360	-0.157390	0.028259
cut	0.096584	-0.134953	1.000000	0.020517	0.074140	-0.218073
color	0.095062	-0.291360	0.020517	1.000000	0.001894	-0.047373
clarity	0.085568	-0.157390	0.074140	0.001894	1.000000	-0.033082
depth	-0.035058	0.028259	-0.218073	-0.047373	-0.033082	1.000000
table	-0.100872	0.181646	-0.433306	-0.026481	-0.060149	-0.295733
price	-0.307092	0.921592	-0.053491	-0.172431	-0.063147	-0.010729
x	-0.406331	0.977779	-0.126232	-0.270671	-0.167572	-0.025017
y	-0.396480	0.953991	-0.122181	-0.263915	-0.164300	-0.029069
z	-0.401758	0.961048	-0.150647	-0.270011	-0.165640	0.095023

	table	price	x	y	z
Unnamed: 0	-0.100872	-0.307092	-0.406331	-0.396480	-0.401758

Plotting the correlation heatmap :

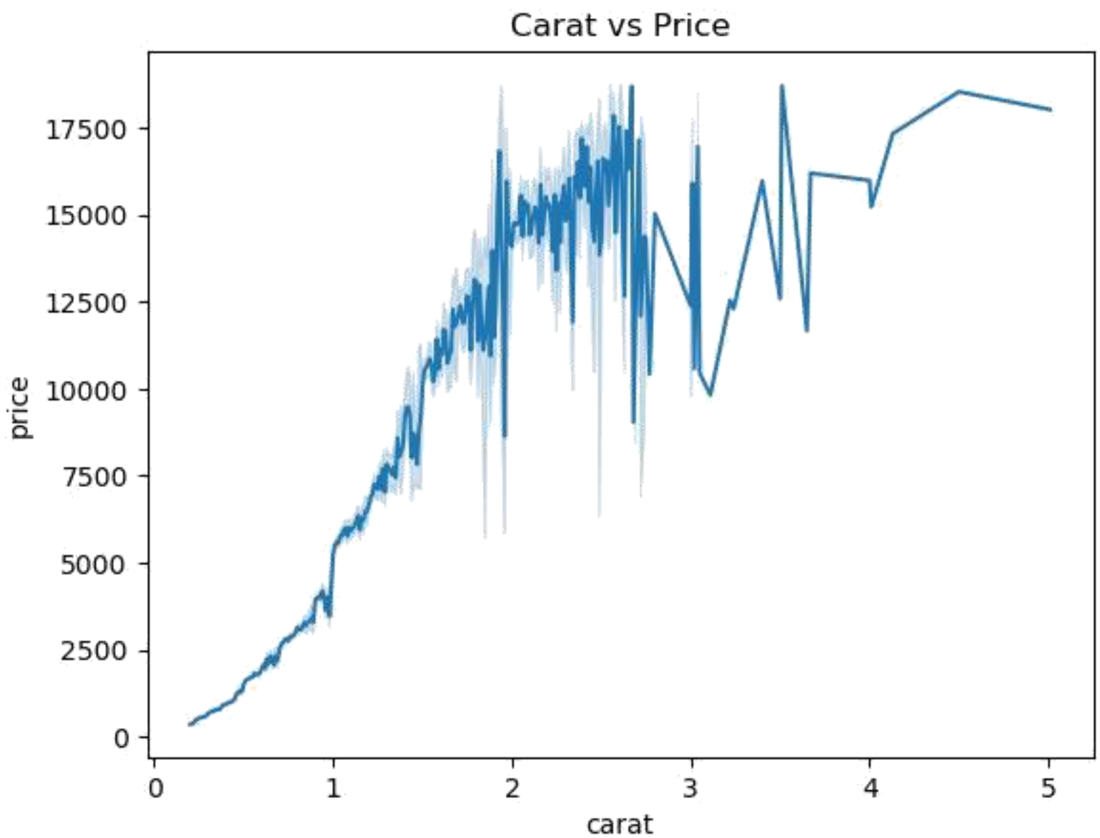
[13]:
```
plt.figure(figsize = (10,10))
sns.heatmap(data.corr(),annot = True, cmap = 'coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

x 0.196097 0.307231 1.000000 0.974918 0.975435



Plotting the relationship between Price and Carat

[14]:
```
sns.lineplot(x = 'carat',y ='price', data = data)
plt.title('Carat vs Price')
plt
```



From the line-plot it is quite clear that the price of diamonds increase with the increase in the carat of the diamond. However, Diamonds with less carat also have a high price. This is because of the other factors that affect the price of the diamonds.

Train Test Split

[15]:
```
x_test,x_train,y_test,y_train =
 train_test_split(data.drop('price',axis =␣↪1),data['price'])
```

Model Building

Decision Tree Regressor

[16]:
```
 dt = DecisionTreeRegressor()
dt
```

[16]: DecisionTreeRegressor()

Train the model

```
[17]:  dt.fit(x_train,y_train)
```

```
[17]: DecisionTreeRegressor()
```

Train accuracy

```
[18]:  dt.score(x_train,y_train)
```

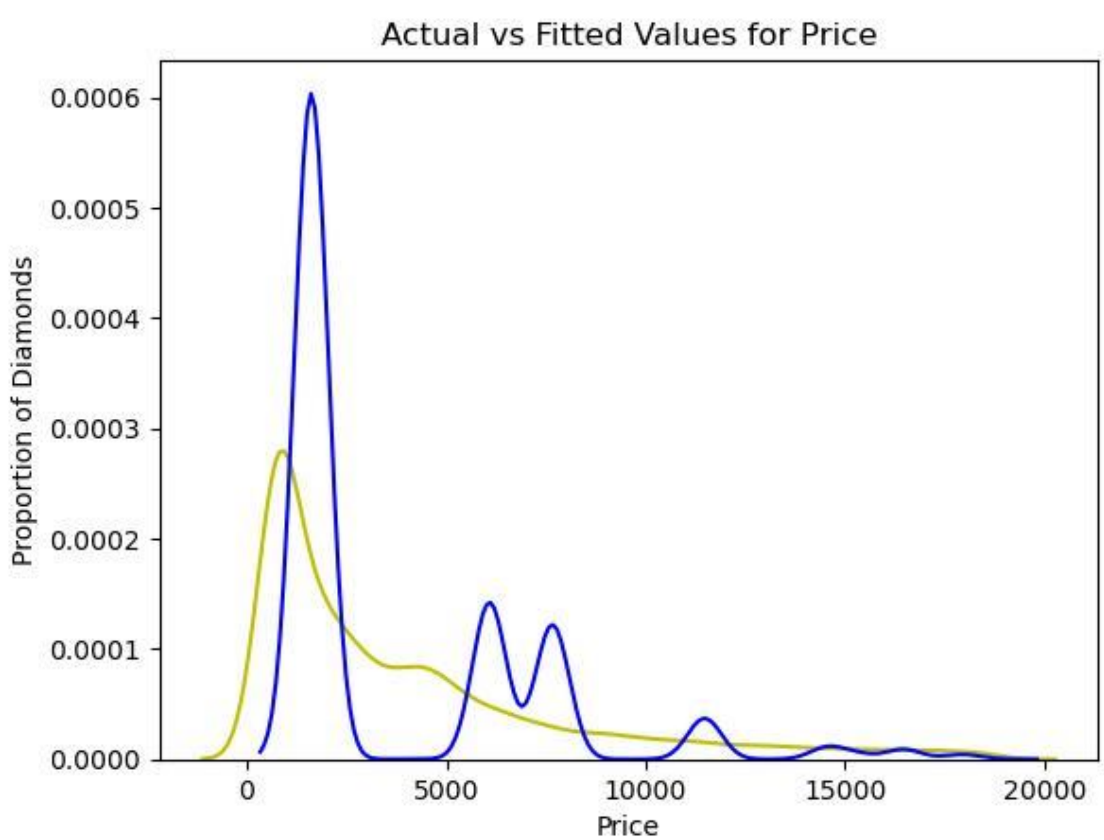
```
[18]: 0.7934132493503449
```

```
[19]:  #predicting the test set
dt_pred = dt.predict(x_test)
```

Model Evaluation

Distribution plot for actual and predict values

```
[20]: ax = sns.distplot(y_test, hist = False, color = 'y', label = 'Actual Value')
sns.distplot(dt_pred,hist = False, color = 'b', label = 'Fitted Values', ax =␣
 ↪ax)
plt.title('Actual vs Fitted Values for Price')
plt.xlabel('Price')
plt.ylabel('Proportion of Diamonds')
plt.show()
```



```
[21]: print("Decision Tree Regressor RMS:" , np.
      ↪sqrt(mean_squared_error(y_test,dt_pred)))
      print('Decision Tree Regressor ACCURACY:',dt.score(x_test,y_test))
      print('Decision Tree Regressor MAE:', mean_absolute_error(y_test,dt_pred))
```

```
Decision Tree Regressor RMS: 1810.2308988961279
Decision Tree Regressor ACCURACY: 0.8951691025250194
Decision Tree Regressor MAE: 1064.8658494661686
```

OBSERVATION: Model Accuracy : 0.895169102525019

CONCLUSION: In conclusion, this project provided valuable insights into the factors affecting diamond prices and developed an accurate predictive model for estimating prices based on various features. Understanding these factors can assist stakeholders in making informed decisions regarding pricing strategies and inventory management in the jewellery industry.

SUMMARY: In summary, the diamond price analysis reveals that carat weight, cut quality, colour, and clarity are key determinants of diamond prices. By leveraging machine learning techniques, businesses can gain valuable insights into market dynamics and make data-driven decisions to drive growth and success in the jewellery industry.