Price Prediction Modeling for Diamonds

Fırat Canberk IŞIKLI
Bahcesehir University
firatcanberk.isikli@bahcesehir.edu.tr

Hasan Fahri YETİM
Bahcesehir University
hasanfahri.yetim@bahcesehir.edu.tr

***Abstract*—In this paper, we intended to draw a model for predicting diamond prices by exploration of the dataset from Kaggle. In doing so, we aimed to increase the correlation between price and the attributes of diamonds that effect it. For this goal, firstly the role of outliers is examined and then to avoid multicollinearity, simple multiplication of three dimensions are added to the dataset as a new column, i.e. 'volume' to mimic the real volumes of them. When linear, ridge and Lasso regression models were applied, we achieved 92%, 92%, and 88% accuracy rates.**

I. ATTRIBUTES IN THE DATASET

As the industry standard, there are 4 characteristics taken into account for describing a diamond which are known as 4 C's: Color, clarity, cut, and carat weight.

Color is a categorical attribute. It is ranked on a scale between D and Z. In our dataset, diamonds with color range from D to J as 7 different categories were included.

Clarity is a measurement that indicates how spotless a diamond is. In our dataset there were items from 8 different groups in terms of this attribute as I1, SI2, SI1, VS2, VS1, VVS2, VVS1, IF; as I1 considered to be the worst and IF the best.

Cut attribute defines the quality of craftsmanship while a diamond is shaped. In our dataset, there were items from 5 categories as Fair, Good, Very Good, Premium, and Ideal in an increasing quality order.

Carat is a unit of mass which is equal to 0.2 grams and used especially in the context of gemstones. Diamonds in our dataset had carat values from 0.2 to 5.01.

Price is the attribute of a diamond for which we aim to develop a model in the frame of this paper and is expressed in US dollars in our dataset.

Other attributions given place in our data frame are dimensions of each diamond given in millimeters under x, y, and z columns, table which is the ratio of top of a diamond to the widest length of it, and depth which is the total depth percentage calculated with the formula 2 * z / (x + y) .

II. MAKING SENSE OF AND CLEANING THE DATASET

Firstly, we referred to info() function to have a basic insight of our data in hand.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 11 columns):
Unnamed: 0    53940 non-null int64
carat         53940 non-null float64
cut           53940 non-null object
color         53940 non-null object
clarity       53940 non-null object
depth         53940 non-null float64
table         53940 non-null float64
price         53940 non-null int64
x             53940 non-null float64
y             53940 non-null float64
z             53940 non-null float64
dtypes: float64(6),int64(2),object(3)
```

We observed that our data is a table with 53490 rows and 11 columns, where 8 of them are filled with numerical values and 3 with categorical. Also, there were no null cells.

Then, we applied head() function on our data frame, we saw that it looked like in the following format:

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

Among columns, "Unnamed: 0" was included only for indexing the items, thus, it was irrelevant in terms of our price prediction aim. We dropped it and checked the new version with the following code:

```
df.drop('Unnamed: 0', inplace=True, axis=1)
df.info()
```

The attributes x, y, and z were values indicating the dimensions of diamonds in millimeters, table and depth were ratios based on those values, carat was the weight and price was the money value; so, they should not have been equal to zero. We checked our dataset in line with this insight:

```
print("Number of rows with carat == 0: {}
".format((df.carat == 0).sum()))
print("Number of rows with x == 0: {}
".format((df.x == 0).sum()))
print("Number of rows with y == 0: {}
".format((df.y == 0).sum()))
print("Number of rows with z == 0: {}
".format((df.z == 0).sum()))
print("Number of rows with depth == 0: {}
".format((df.depth == 0).sum()))
print("Number of rows with table == 0: {}
".format((df.table == 0).sum()))
print("Number of rows with price == 0: {}
".format((df.price == 0).sum()))
```

```
Number of rows with carat == 0: 0
Number of rows with x == 0: 8
Number of rows with y == 0: 7
Number of rows with z == 0: 20
Number of rows with depth == 0: 0
Number of rows with table == 0: 0
Number of rows with price == 0: 0
```

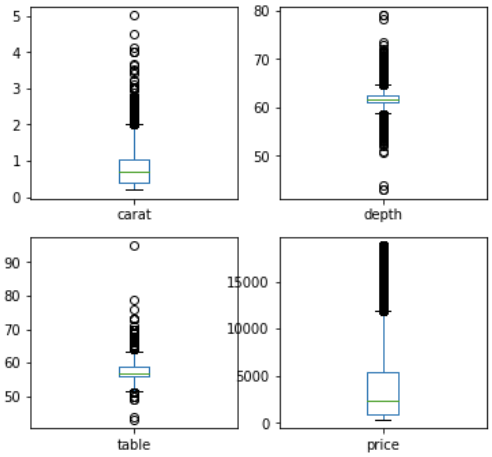
It turned out that there were dimension attributes which were included in our dataset as equal to zero. We excluded the items with this controversial aspect:

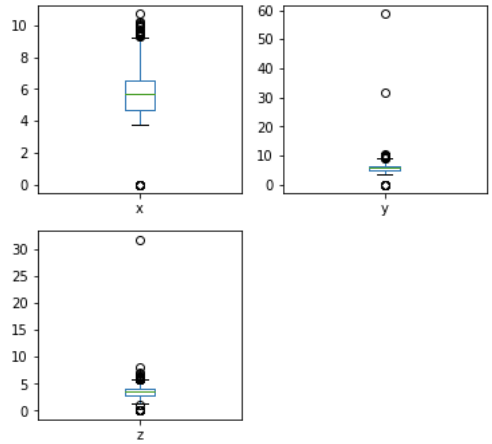
```
df = df[(df.x != 0) & (df.y != 0) & (df.z != 0)]
df.info()
```

As the result, only 20 items were discarded from our dataset. For this is a negligible amount given that we had more than 53,000 items in our dataset, we ignored the effect caused by this deletion.

III. VISUAL EXPLORATION OF THE DATASET AND DEALING WITH OUTLIERS

As seen by the figure below, we boxplotted each attribute to see the distribution of them:





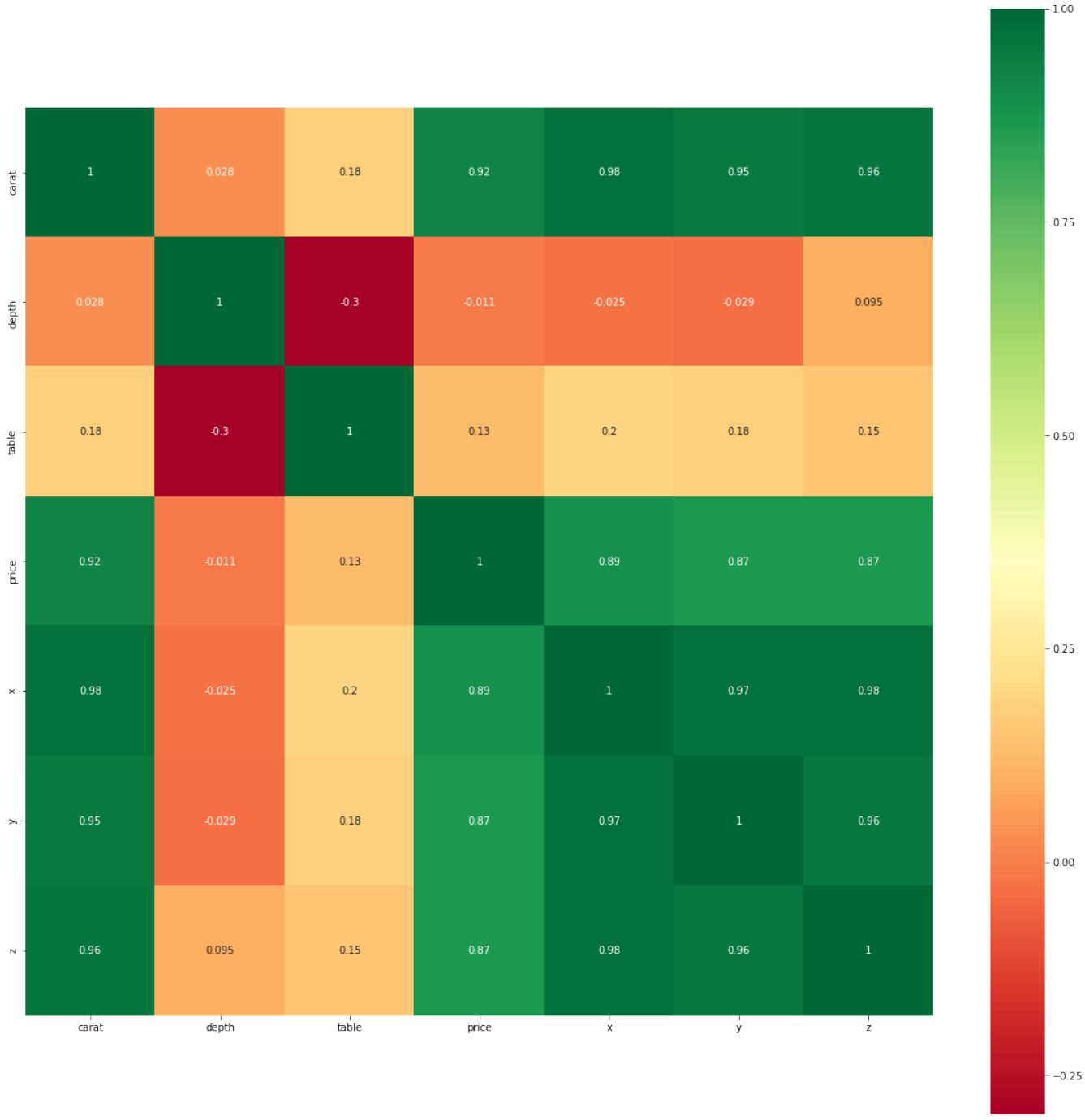
We observed that carat and price attributes had outliers above median values and there were outliers both above and below median values for other attributes.

Then, we created two heatmaps which the first one had outliers remained the same and the second had them transformed to mean values for each attribute. Then, we compared the correlation coefficients between the two. The code and the visual results of this process are as below:

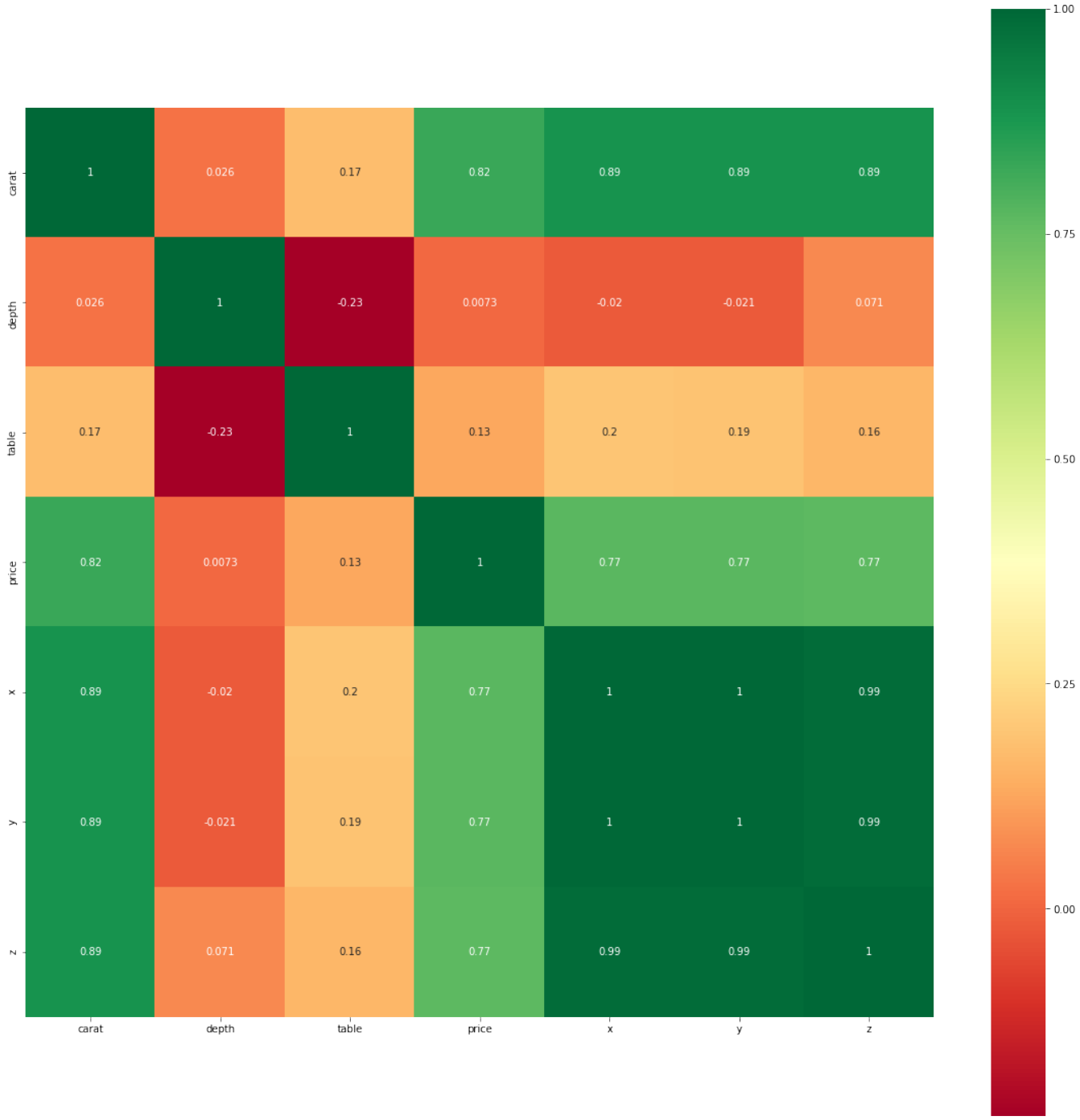
```
plt.figure(figsize=(20,20))
p=sns.heatmap(df.corr(),
annot=True,cmap='RdYlGn',square=True)
def outliers(var):
    a = []
    q1 = df[var].quantile(.25)
    q2 = df[var].quantile(.5)
    q3 = df[var].quantile(.75)
    iqr = q3-q1
    ulim = float(q3+(1.5*iqr))
    llim = float(q1-(1.5*iqr))

    for i in df[var]:
        if i > ulim:
            i=np.NaN
        elif i < llim:
            i = np.NaN
        else:
            i=i
        a.append(i)
    return a
for col in
df.select_dtypes(exclude='object').columns:
    df[col] = outliers(col)
for i in
df.select_dtypes(exclude='object').columns:
    df[i]=df[i].fillna(df[i].mean())

plt.figure(figsize=(20,20))
p=sns.heatmap(df.corr(), annot=True,cmap=
'YlGnBu',square=True)
plt.show()
```



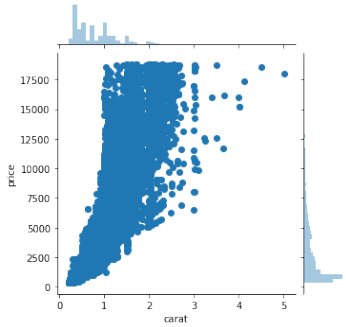
Heatmap with outliers remained the same



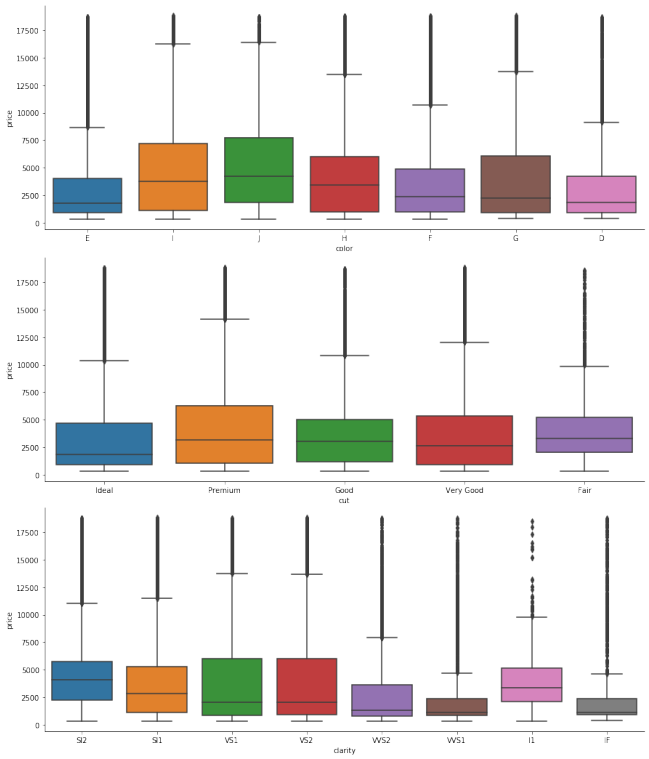
Heatmap where each outlier is transformed to the attribute mean

When we compared the correlation coefficients of price and other attributes between two heatmaps, we observed that the dataset where the outliers are kept the same had higher values as opposed to the other one where they are transformed to mean values and coefficients were decreased in return. Based on this comparison, we concluded that in a regression model, our outlier values were falling close to the original regression line of our dataset and it was statistically meaningful to keep them as they are while developing models for price prediction.

When all attributes related to price are examined, carat has the strongest correlation which is also positive. Therefore, as values of carat increase, price also increases.



The diamond dataset includes categorical data as well as numerical data, so examining categorical data in an isolated manner with boxplots would help get a grasp of the relation between each categorical attribute and the price. Below are given the boxplots for each of the three categorical variable against the price. When we examined the boxplots, we observed that the outlier values fell only above the interquartile range.



On the other hand, we observed that each of x, y, and z has with price, which can be seen in Figures 1 and 2. These variables are dimensions of the diamonds, and we decided to create a variable 'volume' that combines these variables for the sake of simplification. Considering three dimensions and analyzing their impact separately would provide us insights that we would need to consider making use of extensive technical details. Therefore, we decided to simplify in order to stay focused on the general relationship. While the volume of a diamond does not equal the multiplication of its three dimensions, we decided that we could use the simple multiplication as a proxy variable that would mimic the effect of the real volume on price.

In order to take the categorical data into account in the regression, converting them to numerical data is a way to discover the relationship between these categorical independent variables and the dependent variable, i.e. price. Integer encoding and one-hot encoding are common methods that can be used in such a transformation. In our case, both methods should be used in order to include every row in the calculation. Using only integer encoding would result in a poor analysis. Therefore, for each of the categorical variables in our dataset, i.e. 'cut', 'clarity' and 'color', we created dummy variables for every possible value that they could take.

```
In [23]: df.drop(['x','y','z'], axis=1, inplace= True)

In [24]: df_hot =  pd.get_dummies(df)
         df_hot.head()
```

Out[24]:

	carat	depth	table	price	volume	cut_Fair	cut_Good	cut_Ideal	cut_Premium	cut_Very Good
0	0.23	61.5	55.0	326	38.202030	0	0	1	0	0
1	0.21	59.8	61.0	326	34.505856	0	0	0	1	0
2	0.23	56.9	65.0	327	38.076885	0	1	0	0	0
3	0.29	62.4	58.0	334	46.724580	0	0	0	1	0
4	0.31	63.3	58.0	335	51.917250	0	1	0	0	0

5 rows × 25 columns

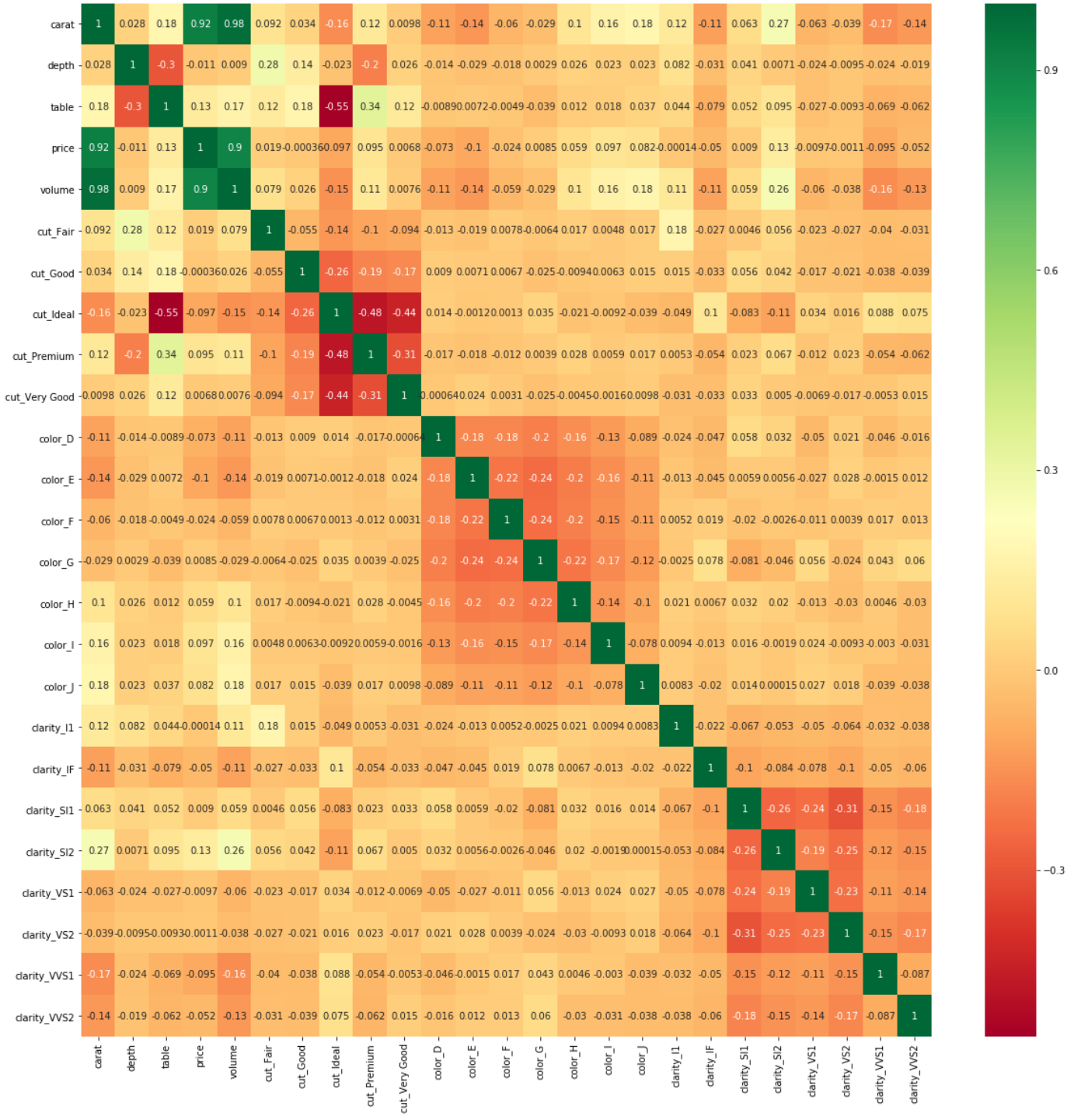
After the implementation of one-hot encoding, in order to get better results, the StandardScaler was used for each variable separately in an effort to standardize the dataset, i.e. transforming the data such that its mean becomes 0 and its standard deviation becomes 1.

```
In [28]: numeric.head()
```

Out[28]:

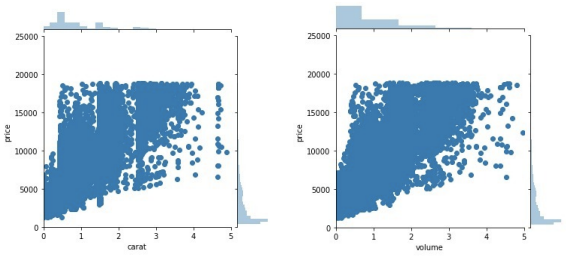
	carat	depth	table	volume
0	-1.198204	-0.174203	-1.099725	-1.172291
1	-1.240417	-1.361090	1.585988	-1.219546
2	-1.198204	-3.385781	3.376463	-1.173891
3	-1.071566	0.454149	0.243131	-1.063334
4	-1.029353	1.082501	0.243131	-0.996948

At the end of these processes, a revised heat-map with the transformed variables is created and given below. The 'volume' and 'carat' variables are seen to have a stronger relationship with price, which is shown by the higher coefficients.



Heatmap with one-hot encoding

This stronger relationship can also be observed from the following scatterplots:



In the end, three models, i.e. linear regression model, ridge model, and Lasso model, are evaluated.

The linear regression model was implemented, and the dataset was divided into training and test datasets of 80% and 20% of observations respectively. The result of the linear regression is shown below.

```
accuracy: 91.72685360494162%
Mean absolute error: 812.6702593939957
Mean squared error: 1360254.322692467
R Squared: 0.9172685360494163
Adjusted R Squared: 0.917091693700823
```

As figure shows, the vales of R-squared and adjusted R-squared are close and sufficiently high. Also, accuracy is 91.72%, and thus support our assumption.

Next regression model considered is a ‘ridge model’ that allows accounting for multicollinearity, which results in inaccurate results in linear regression. The outcomes of the ridge model are given below, which show that similar to linear regression, the explanatory power and accuracy of our model is sufficiently high.

```
accuracy: 91.72702208800212%
Mean absolute error: 812.5537116588846
Mean squared error: 1360226.6210418015
R Squared: 0.9172702208800212
Adjusted R Squared: 0.9170933821328316
```

Final model considered is Lasso regression, which accounts for overfitting by selecting variables by giving ‘0’ coefficients to some of the independent variables. The outcomes of the Lasso model are given below, which show that model accuracy is also sufficiently high, but a little lower than the accuracy figures reached on the other two models.

```
Lasso accuracy:  0.8776771215644659
[0.68160986 0.87590994 0.87723836 0.88154854 0.88524353]
mean = 0.8403100480995895, std = 0.07941842860503429
```

To sum up, we can establish that volume and carat are attributes that are highly correlated with the price of a diamond, while the effect of other variables are rather limited. Also, the models that we built, i.e. linear regression, ridge, and Lasso models, have high explanatory powers and high levels of accuracy.