Machine Learning Engineer Nanodegree

Capstone Project

Kariman K. Gabaa

Nov 10th, 2018

I. Project Overview

Diamond is allotrope of carbon with a crystalline structure (It consists only from carbon element under high pressure and heat), its located deep in the earth, It can be obtained from the sediment collected at river waters and, It can be found with other metals, It's a strong stone and precious. It has many types:

- South Africa Diamond
- Florentine diamond
- Shah's Diamond
- Star of Africa Diamond
- Regent Diamond

Diamond is known for its charming effect on ladies due to its value and shine also it's known as a symbol of romanticism and elegance, In addition to other factors like it's rarity and high demand -because it's not affected by external factors and its ability to maintain its shape and characteristics-, It's so expensive, so it's important to have a tool to determine the price of diamonds, The price of diamond varies according to different factors like form, color, weight or degree of purity ..etc.

Problem Statement

We want to predict the price of diamonds, We found that the price varies according to different factors since it is very delicate and expensive trade, Diamonds price prediction help us to know the real price according to its specifications and prevent fraud, we can predict the price by regression algorithms.

Evaluation Metrics

To evaluate the model, we will use the coefficient of determination (R2), it's used to measure the error of the regression model, It takes value from 0 to 1.

The Formula:

The total sum of squares:

The sum of squares of residuals:

The coefficient of determination(R2):

It compares the real value and the predictive value to find out the match between them. We can also use the mean square error, it takes a square root of the mean squared error, and it's the measure of the deviation of the predicted value from the actual value. The formula:

It also compares the real value and the predicted value to find out the match between them.

II. Analysis

Data Exploration

Before expploring the data, we need to import the basic libraries (numpy, pandas, matplotlib, seaborn) and the dataset

1- We show part of data, we note there is a column called unnamed, it seems to be unuseful, and so we drop it.

2- From the first glance, the data is much unclean

	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
5	6	0.24	Very Good	J	VVS2	62.8	57.0	336	3.94	3.96	2.48
6	7	0.24	Very Good	I	VVS1	62.3	57.0	336	3.95	3.98	2.47
7	8	0.26	Very Good	H	SI1	61.9	55.0	337	4.07	4.11	2.53
8	9	0.22	Fair	E	VS2	65.1	61.0	337	3.87	3.78	2.49
9	10	0.23	Very Good	H	VS1	59.4	61.0	338	4.00	4.05	2.39

We dropped column by:

```
# we note there is column(Unnamed:0) unusefull
data.drop(data.columns[0],axis-1,inplace-True)
```

Now we have:

Data columns: become 10 columns. Data rows: from 0 to 53939 rows that mean 53940 entries. Memory usage: 4.1+ MB.

The Features:

• carat: it’s weight of the diamond.

• cut: it’s quality of the cut.

• color: it’s diamond color.

• clarity: it’s the measurement of how clear the diamond is.

• x: it’s length in mm.

• y: it’s width in mm.

• z: it’s depth in mm.

• depth: it’s total depth percentage = z / mean(x, y) = 2 * z / (x + y).

• table: it’s width of the top of diamond relative to the widest point.

The target value:

• price: it’s price in US dollars.

We noticed, there are 3 categorical columns (cut, color, clarity) and 7 numerical columns.

When describing the dataset by:

```
data.describe()
```

We can note:

1- The mean of ”depth” column nearly to 25% and 50%, This indicates that its effect is minimal on price and also the mean of “ table ” column nearly to 50%.

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

2- the min of length “x”, width ”y”, and depth ”z” columns is zero may be that Error collecting data.

price	x	y	z
53940.000000	53940.000000	53940.000000	53940.000000
3932.799722	5.731157	5.734526	3.538734
3989.439738	1.121761	1.142135	0.705699
326.000000	0.000000	0.000000	0.000000
950.000000	4.710000	4.720000	2.910000
2401.000000	5.700000	5.710000	3.530000
5324.250000	6.540000	6.540000	4.040000
18823.000000	10.740000	58.900000	31.800000

In cut column there is 5 type of diamond cut:

1. Ideal 21551
2. Good 4906
3. Premium 13791
4. Very Good 12082
5. Fair 1610

In color column there is 7 colors of diamond :

• G 11292

• E 9797

• F 9542

• H 8304

• D 6775

• I 5422

• J 2808

In clarity column there is 8 type of diamond clarity:

• SI1 13065

• VS2 12258

• SI2 9194

• VS1 8171

• VVS2 5066

• VVS1 3655

• IF 1790

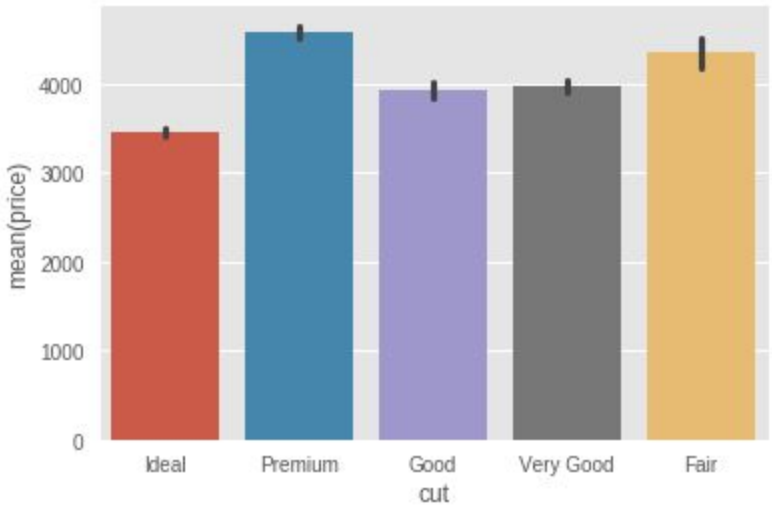
• I1 741

Visualization

1- we drew a relationship between cut and price by barbot :

```
a=barplot.sns(data.cut,data.price)
```

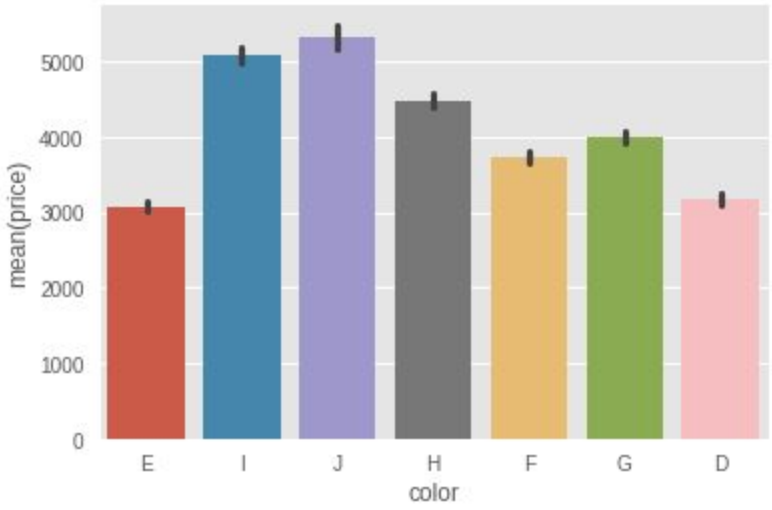
we can see that Premium is the highest price :



2- we drew a relationship between color and price by barbot

```
a=sns.barplot(data.color,data.price)
```

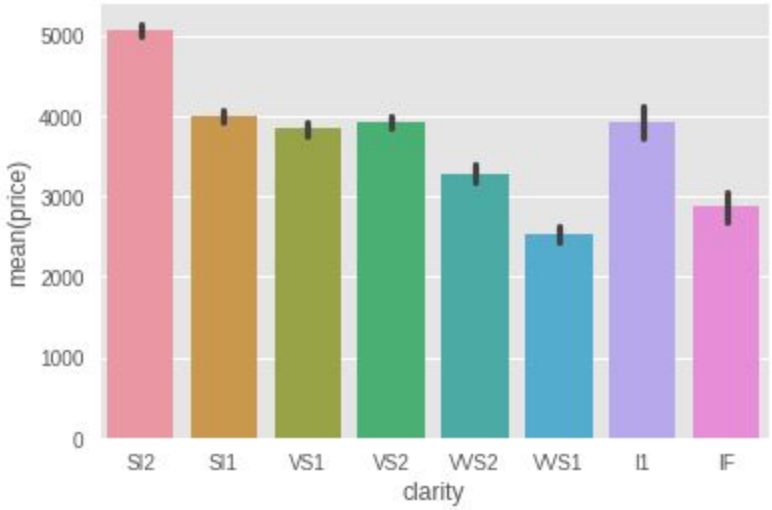
We can see that J is the highest price :



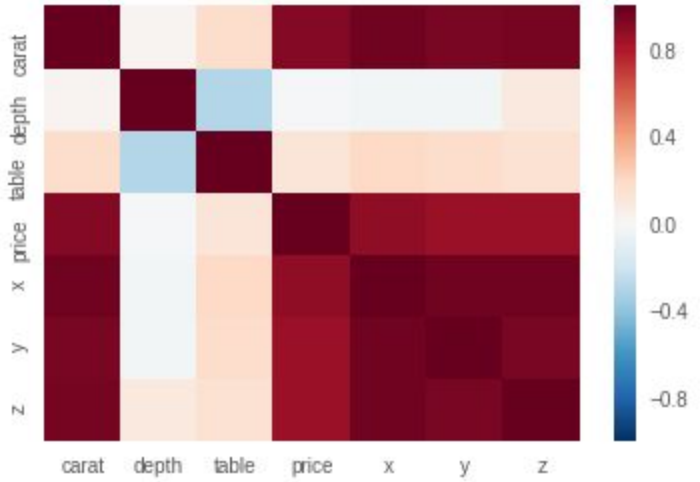
3- We drew a relationship between clarity and price by barbot

```
a=sns.barplot(data.clarity,data.price)
```

We can see that SI2 is the highest price:



The heatmap below shows the relationship between all numerical columns, each other and target value:



III. Methodology

Data Preprocessing & Feature Engineering

We can notice that:

• we have 3 category columns(cut , color, and clarity)

• x,y,z have zero value and that illogical.

1- so, we don't used the row of the zero value of x or y or z by excluding zero data :

data=data.loc[(data['x']!=0)&(data['y']!=0)&(data['z']!=0.0)]

2- we convert all categorical into numerical by replace every categorical value to numerical value :

The all data became numerical:

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	1	7	2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	2	7	3	59.8	61.0	326	3.89	3.84	2.31
2	0.23	4	7	5	56.9	65.0	327	4.05	4.07	2.31
3	0.29	2	2	4	62.4	58.0	334	4.20	4.23	2.63
4	0.31	4	1	2	63.3	58.0	335	4.34	4.35	2.75

Models

Before choosing the algorithms, we first import libraries required and split the data to training and testing and then Scaling the values in data .

we import from sklearn.model_selection train_test_split to spilt data and sklearn.preprocessing StandardScaler to transform the data.

after splitting data :

X_train= data except for the target value" price".

X_test= target value "price".

test_size= 20% of data.

```
# split data
X_train,X_test,y_train,y_test=train_test_split(data.drop(['price'],axis=1
),data['price'],test_size=0.20,random_state=10)
scale = StandardScaler()
X_train_scaled = scale.fit_transform(X_train)
X_test_scaled = scale.transform(X_test)
print('Number of rows in the total set: {}'.format(data.shape))
print('Number of rows in the training set: {}'.format(X_train.shape))
print('Number of rows in the test set: {}'.format(X_test.shape))
```

Number of rows in the total set: (53920, 10)

Number of rows in the training set: (43136, 9)

Number of rows in the test set: (10784, 9)

IV. Results

Model Evaluation and Validation

1-Benchmark Model: naive Bayes, we import its libraries and records R2 score and mean square error:The smallest score

r2_score: 0.8747099357220705

mean square error: 2059994.3889094954

2- Models: we chose

Linear regression:

r2_score:0.902853789970007.

mean square error:1597258.7189490432.

DecisionTrees Model:

r2_score:0.9655994693159025.

mean square error:565606.703078635.

Random Forest Model:

r2_score:0.9808131435311431.

mean square error:315466.4888588423.

Finally, we can see the highest measure is Random Forest Model and the smallest is naive Bayes.

Cross Validation

We used cross_val_score , it's function simply computes the R2 for each fold of the cross-validation by default and we split to 10 CVs , it's returns an array, we can measure the mean for it, we can see that in the cell below:

```
from sklearn.model_selection import cross_val_score
print("-------R2-------")
print("NaiveBayes:",np.mean(cross_val_score(gnb,X_train_scaled,y_train,cv
= 10)))
print("LinearRegression:",np.mean(cross_val_score(LR,X_train_scaled,y_trai
n,cv = 10)))
print("DecisionTrees:",np.mean(cross_val_score(regressor,X_train_scaled,y_
train,cv = 10)))
print("RandomForest:",np.mean(cross_val_score(rfg,X_train_scaled,y_train,c
v = 10)))
```

The result:

R2:

NaiveBayes: 0.013515326940720811

LinearRegression: 0.8921939236266118

DecisionTrees: 0.9651462536235369

RandomForest: 0.9789689546970992

We can see that the highest measure is Random Forest Model and the smallest is naive Bayes. And it can compute mean square error for each folder of cross validation, its return array; we can measure the mean for it, by the cell below:

```
print("------MSR------")
print("NaiveBayes:",np.mean(cross_val_score(gnb,X_train_scaled,y_train,sco
ring='neg_mean_squared_error',cv = 10)))
print("LinearRegression:",np.mean(cross_val_score(LR,X_train_scaled,y_trai
n,scoring='neg_mean_squared_error',cv = 10)))
print("DecisionTrees:",np.mean(cross_val_score(regressor,X_train_scaled,y_
train,scoring='neg_mean_squared_error',cv = 10)))
print("RandomForest:",np.mean(cross_val_score(rfg,X_train_scaled,y_train,s
coring='neg_mean_squared_error',cv = 10)))
```

The result: mean square error :

NaiveBayes:-1184445.413044155

LinearRegression: -1709047.521724832

DecisionTrees: -554852.0946947063

RandomForest: -329749.97791458486

We can see the highest measure is DecisionTrees Model and the smallest is naive bayes.

V. Conclusion

Reflection

This problem was very difficult especially with the data incorrect value and not cleaning but the hard part was cleaning the data and choosing the model to deal with it, we have tried many different solutions and many different models, but It was very bad in training and speed such as SVM. The interested in this project, they were Exploring data and Visualization, that help me more in understanding the nature of data and their problems. Also Pre-processing, it helps the scoring model.

Improvement

To improve this model in the future, we can use the neural network CNN, we can use some techniques to determine the best features such as PCA, we can use NLP for the text, make more analysis and test many other algorithms.

References

• https://en.wikipedia.org/wiki/Diamond

• https://www.lumeradiamonds.com/diamond-education/diamondcut?

• fbclid=IwAR2i52Kbfot3VWfNJIvyeFR_

• GvpLQjIvKSly4DWY0PPwepK5OuT__eGCok

• https://www.lumeradiamonds.com/diamond-education/diamondclarity?

• fbclid=IwAR02iBm6UJQZSM-t20eYktnLFT4dMC1SD5HFeJlIb_

• 1Wi_IsY5uIG-0DOU

• https://en.wikipedia.org/wiki/Coefficient_of_determination

• -https://en.wikipedia.org/wiki/Mean_squared_error

• -https://www.kaggle.com/shivam2503/diamonds

• -https://drive.google.com/open?id=1a3jafAJe9Idxx84jqmTS4gHsY_8dQKmB