Chetan uppara

CSCI Final Project

Data Set and the goal of the project

For my final project I've picked a data set that uses a dataset about diamonds. This dataset contains 10 variables. The goal of the project is to to predict the price of the diamond with the given variables and see how else the price changes.

#	# carat	cut	color	clarity	# depth	# table	# price
Index counter	Carat weight of the diamond	Describe cut quality of the diamond. Quality in increasing order Fair, Good, Very Good, Premium, Ideal	Color of the diamond, with D being the best and J the worst	How obvious inclusions are within the diamond: (in order from best to worst, FL = flawless, I3= level 3 inclusions) FL,IF,	depth % :The height of a diamond, measured from the culet to the table, divided by its average girdle diameter	table%: The width of the diamond's table expressed as a percentage of its average diameter	the price of the diamond
		Ideal 40%	G 21%	SI1 24%		58.60 - 61.20 Count: 13,230	
		Premium 26%	E 18%	VS2 23%			
1 – 53.9k	0.2 – 5.01	Other (18598) 34%	Other (32851) 61%	Other (28617) 53%	43 – 79	43 – 95	326 – 18.8
1	0.23	Ideal	E	SI2	61.5	55	326
2	0.21	Premium	E	SI1	59.8	61	326
3	0.23	Good	E	VS1	56.9	65	327
4	0.29	Premium	I	VS2	62.4	58	334
5	0.31	Good	J	SI2	63.3	58	335
6	0.24	Very Good	J	VVS2	62.8	57	336
7	0.24	Very Good	I	VVS1	62.3	57	336
8	0.26	Very Good	H	SI1	61.9	55	337
9	0.22	Fair	E	VS2	65.1	61	337
10	0.23	Very Good	H	VS1	59.4	61	338
11	0.3	Good	J	SI1	64	55	339
12	0.23	Ideal	J	VS1	62.8	56	340
13	0.22	Premium	F	SI1	60.4	61	342
14	0.31	Ideal	J	SI2	62.2	54	344
15	0.2	Premium	E	SI2	60.2	62	345
16	0.32	Premium	E	I1	60.9	58	345
17	0.3	Ideal	I	SI2	62	54	348

Hypothesis

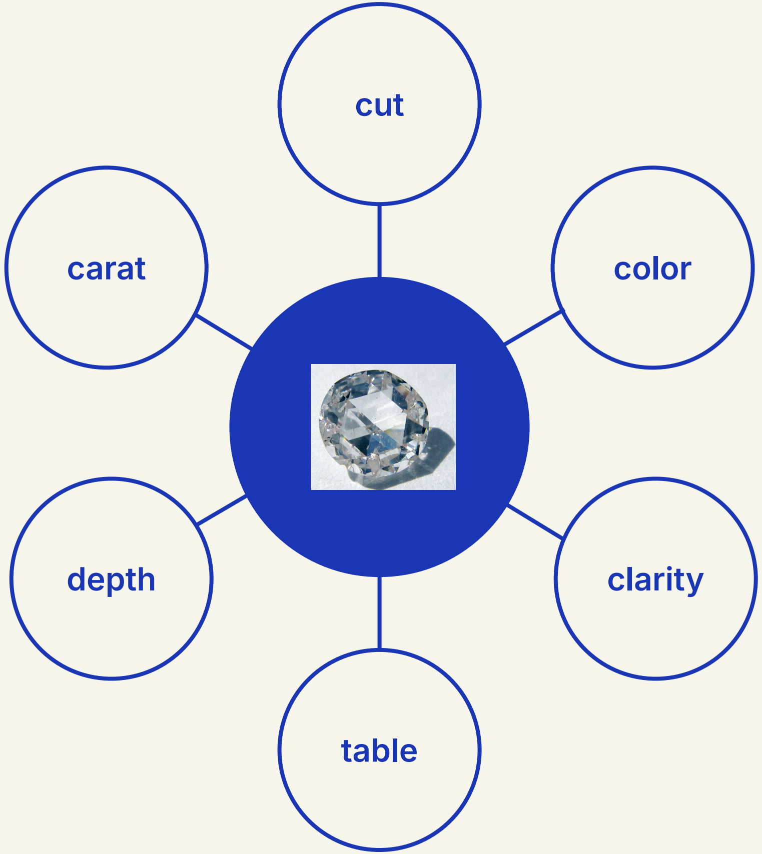
Null Hypothesis

where Data set has a correlation where the clarity, carrots, weight, and quality of the diamond can increase the value of the diamond

Alternate Hypothesis

where the data set does not correlate between total depth percentage and the price of the diamonds

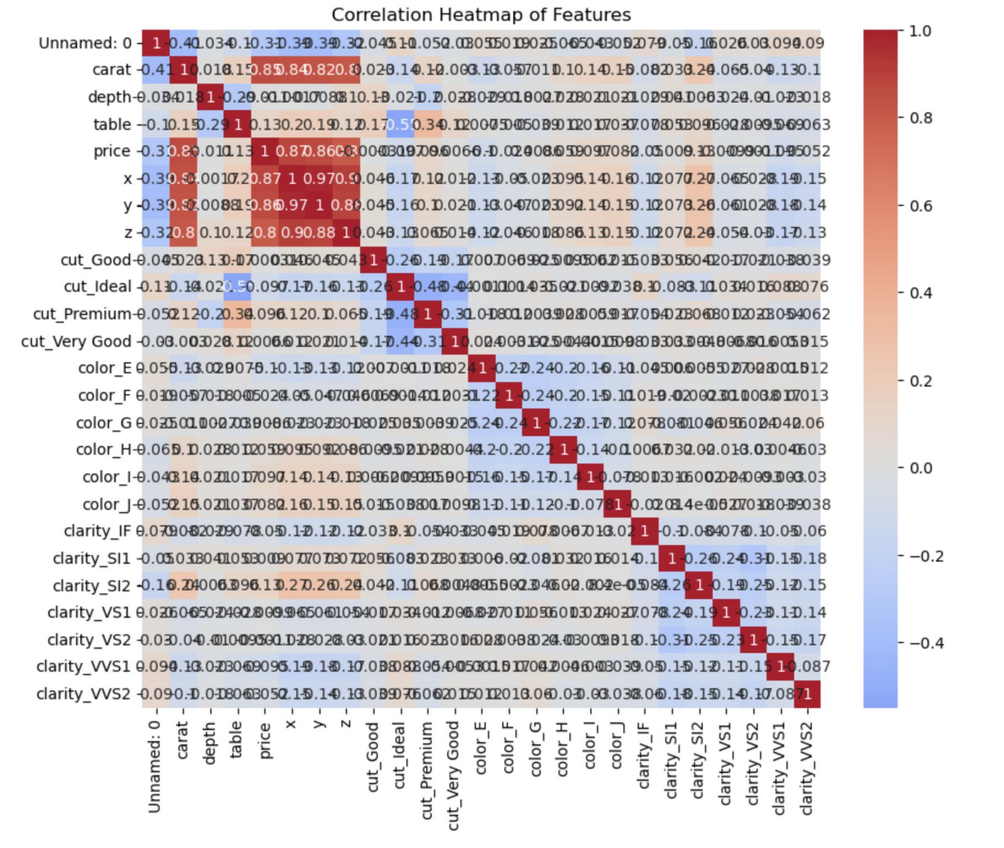
Variables



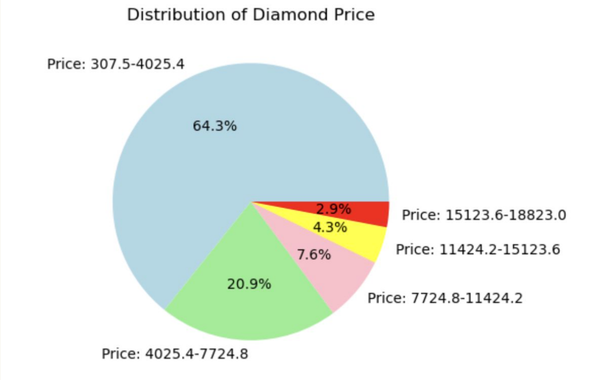
Data preprocessing

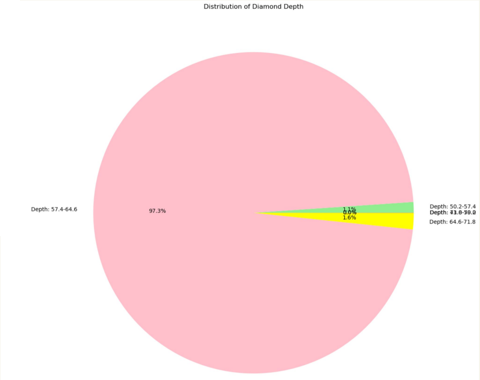
1. Replaced the null values with the median.
2. I used get dummies(enumerate the values) for my categorical columns.
3. I didn't standardize the data.
4. There is small amount of multicollinearity, but the variables are related.

```
def clean_data(data_frame):
    number_columns=["carat", "depth", "table", "x", "y", "z"]
    catagory_columns=["cut", "color", "clarity"]



    for col in number_columns:
        data_frame[col] = data_frame[col].fillna(data_frame[col].median())

    df = pd.get_dummies(data_frame, columns=catagory_columns, drop_first=True)

    df = df.replace([np.inf, -np.inf], np.nan).fillna(0)

    df = df.astype(int)

    return df
```



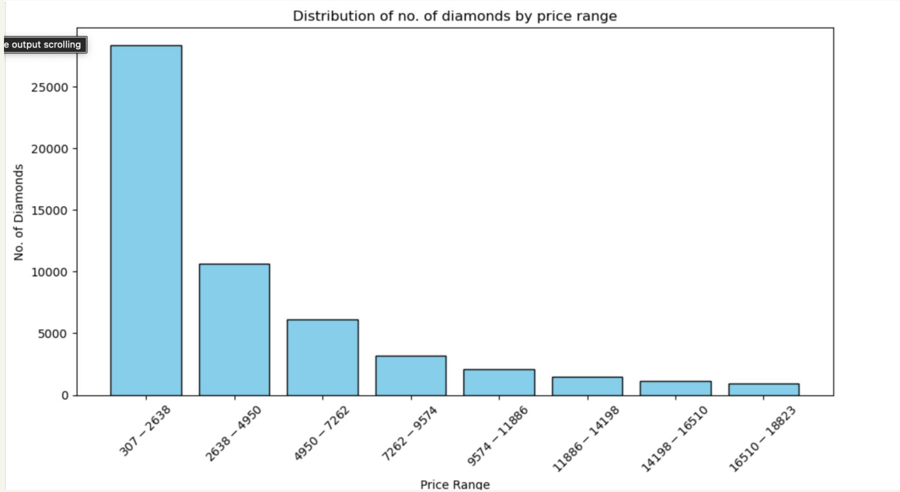
Visualization the data





- First Pie chart is for where it predicts the distribution of the price of the diamonds are based on the variables

- 97.3% of the diamond depth is about 54-65mm, 1.1% of the diamond depth is about 50-5mm7, and 1.6% of the diamond depth is about 64-71mm



- Distribution of no. of diamonds by price range

Machine Learning Models

```python
def linear_model(ycolumn):

    X = diamond_df.drop(columns=[ycolumn,"Unnamed: 0"])
    y = diamond_df[ycolumn]

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

    lin_reg = LinearRegression()
    lin_reg.fit(X_train, y_train)

    y_pred = lin_reg.predict(X_test)

    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)

    return mse, r2
```

Linear Regression model

```python
def decision_tree(ycolumn):
    X = diamond_df.drop(columns=[ycolumn,"Unnamed: 0"])
    y = diamond_df[ycolumn]

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
    model =tree.DecisionTreeClassifier(max_depth=9, random_state=42, min_samples_leaf=3)
    model = model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    return mse, r2
```

Decision tree model

Results

Linear Regression model

```
MSE: 2141792.6468459307, R2: 0.8626700641808112
```

Decision Tree model

```
MSE: 2495368.551415153, R2: 0.8399990757668726
```