Sparkling Insights:
Exploring the Diamond Data

Apostu Alexandru-Mihai (507)
Dragomir Elena Alexandra (507)

Contents

- Exploratory Data Analysis
- Plots
- Machine Learning

Introduction

Diamond Valuation

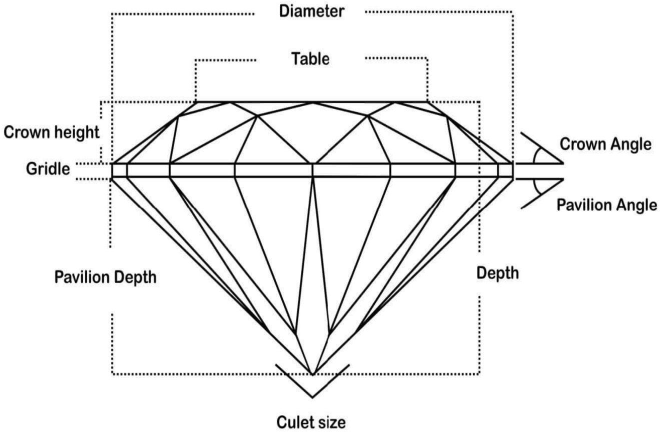
- The dataset offers a detailed look at how characteristics like carat, cut, and color influence diamond market values, essential for stakeholders from miners to consumers.

Analysis Objectives

- To unravel how various diamond attributes interplay to impact pricing, providing insights that guide smarter business and consumer decisions.

EDA

Dataset columns



Content

- **price** price in US dollars
- **carat** weight of the diamond
- **cut** quality of the cut
- **color** diamond colour, from J (worst) to D (best)
- **clarity** a measurement of how clear the diamond is (I1 -worst, SI2, SI1, VS2, VS1, VVS2, VVS1, IF - best)
- **x** length in mm
- **y** width in mm
- **z** depth in mm
- **depth** total depth percentage = z / mean(x, y)
- **table** width of top of diamond relative to widest point

Dataset Overview

```
[ ] df.head()
```

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

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 10 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   carat    53940 non-null  float64
 1   cut      53940 non-null  object 
 2   color    53940 non-null  object 
 3   clarity  53940 non-null  object 
 4   depth    53940 non-null  float64
 5   table    53940 non-null  float64
 6   price    53940 non-null  int64  
 7   x        53940 non-null  float64
 8   y        53940 non-null  float64
 9   z        53940 non-null  float64
dtypes: float64(6), int64(1), object(3)
memory usage: 4.1+ MB
```

- Short view of dataset
- Columns names
- Types of variables

Dataset Overview

```
[ ] unique_values = {col: df[col].unique() for col in df.select_dtypes(include='object').columns}
    unique_values

{'cut': array(['Ideal', 'Premium', 'Good', 'Very Good', 'Fair'], dtype=object),
 'color': array(['E', 'I', 'J', 'H', 'F', 'G', 'D'], dtype=object),
 'clarity': array(['SI2', 'SI1', 'VS1', 'VS2', 'VVS2', 'VVS1', 'I1', 'IF'],
       dtype=object)}
```

```
[ ] numerical_columns = ['carat', 'depth', 'table', 'price', 'x', 'y', 'z']
    ranges = {column: (df[column].min(), df[column].max()) for column in numerical_columns}

    print(ranges)

{'carat': (0.2, 5.01), 'depth': (43.0, 79.0), 'table': (43.0, 95.0), 'price': (326, 18823), 'x': (0.0, 10.74), 'y': (0.0, 58.9), 'z': (0.0, 31.8)}
```

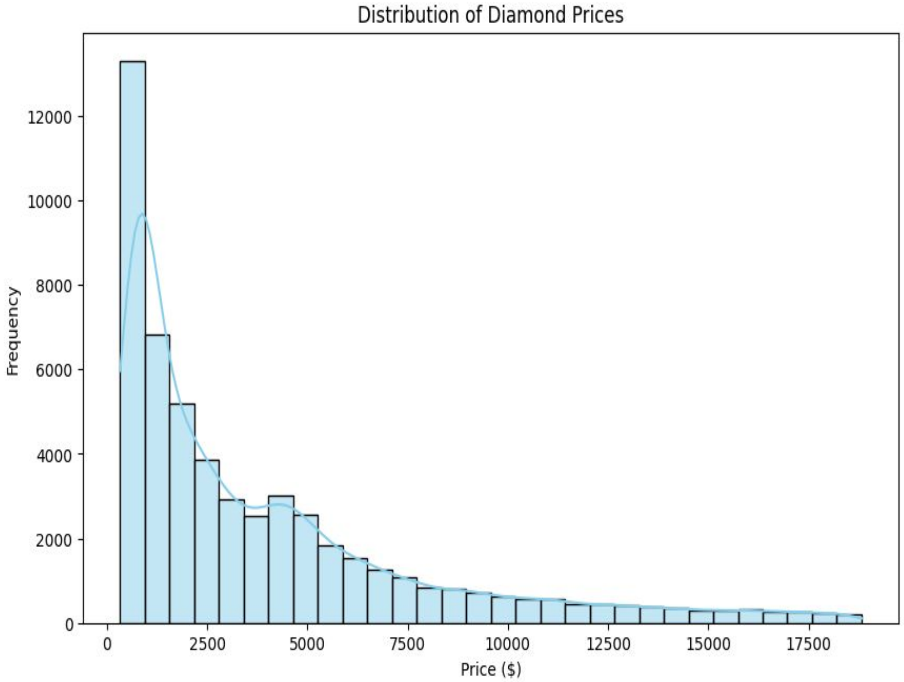
```
[ ] df.describe()
```

	carat	depth	table	price	x	y	z
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000
mean	0.797940	61.749405	57.457184	3932.799722	5.731157	5.734526	3.538734
std	0.474011	1.432621	2.234491	3989.439738	1.121761	1.142135	0.705699
min	0.200000	43.000000	43.000000	326.000000	0.000000	0.000000	0.000000
25%	0.400000	61.000000	56.000000	950.000000	4.710000	4.720000	2.910000
50%	0.700000	61.800000	57.000000	2401.000000	5.700000	5.710000	3.530000
75%	1.040000	62.500000	59.000000	5324.250000	6.540000	6.540000	4.040000
max	5.010000	79.000000	95.000000	18823.000000	10.740000	58.900000	31.800000

- Unique values for categorical columns
- Range for numerical columns
- Description of the dataset

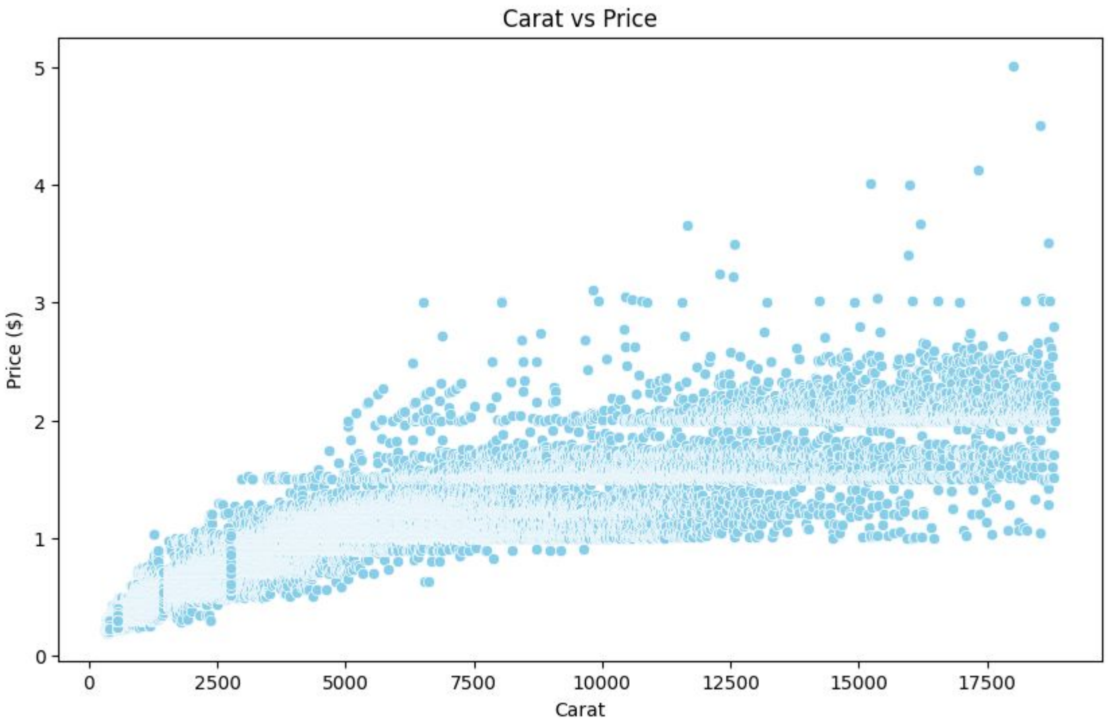
Plots

Distribution of Diamond Prices



- Most diamonds are priced below $5,000.
- There is a significant peak around the $1,000 mark.
- The distribution is right-skewed, with fewer diamonds at higher prices.
- The overlaid curve highlights the concentration of lower-priced diamonds.

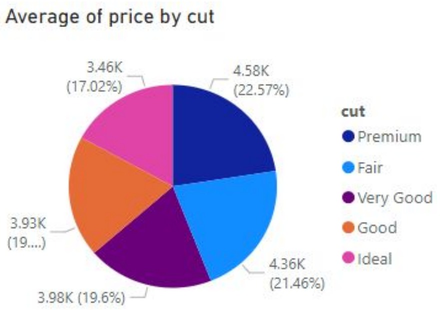
Scatter Plot - Carat vs. Price



- The price is increasing along with the diamonds' weight

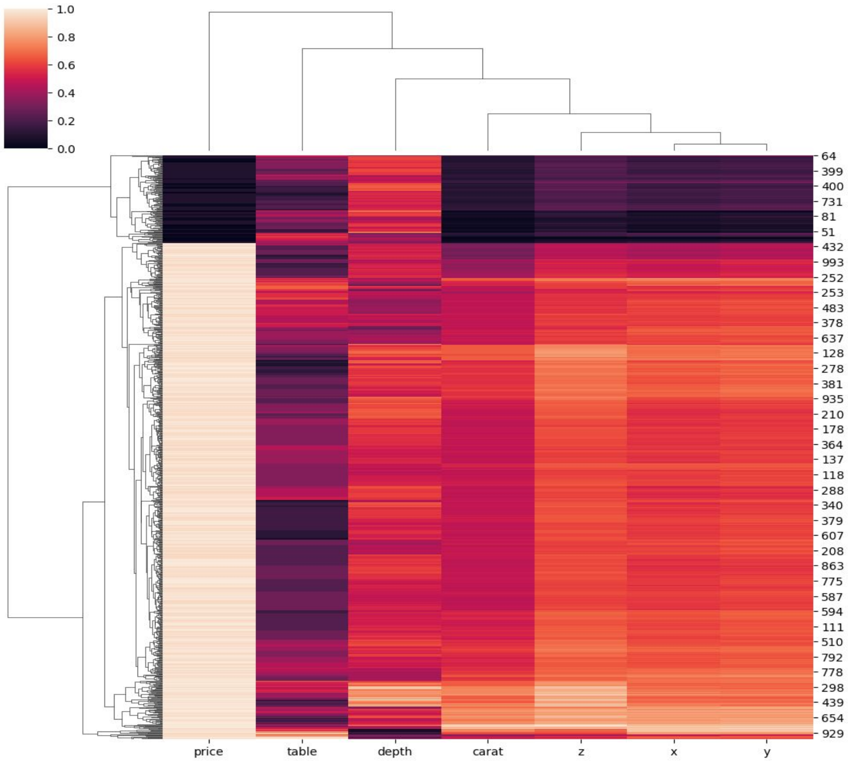
Box Plot - Cut vs. Price

- Premium cuts have the highest median prices, also leading in average price at $4.58K, while Ideal cuts offer a broad price range.
- Note the variance in price within each cut category, particularly the longer tails in Ideal and Premium cuts, which indicate the presence of high-priced outliers.



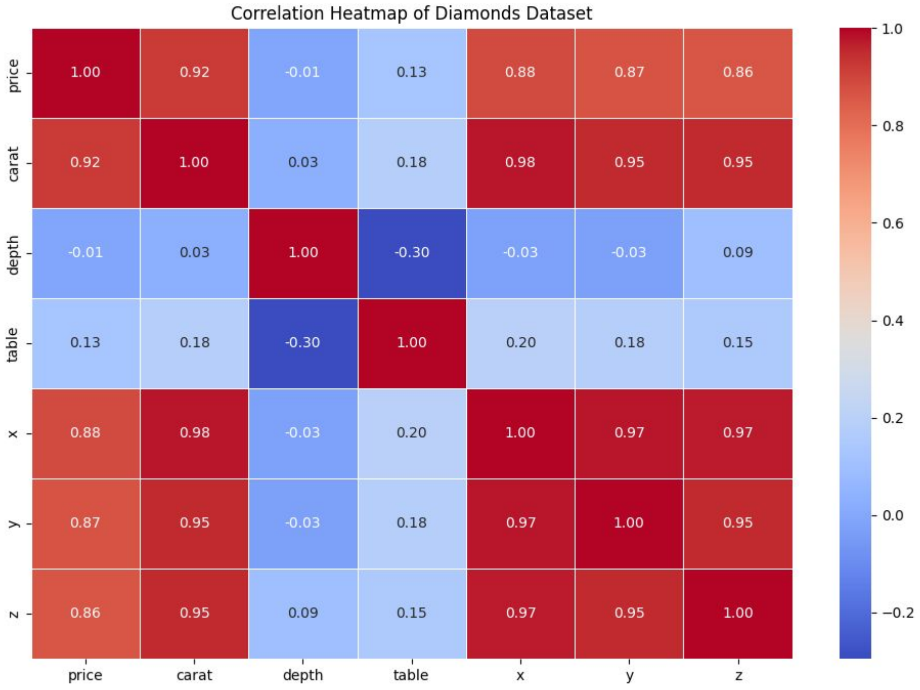
Cut quality does not always directly correlate to higher prices, it seems like market demand and diamond availability also play critical roles

Hierarchical Clustermap



- This hierarchy allows ordering the data in clusters.
- It arranges the data using a dissimilarity matrix (also called distance matrix), which gives information on how far are two features

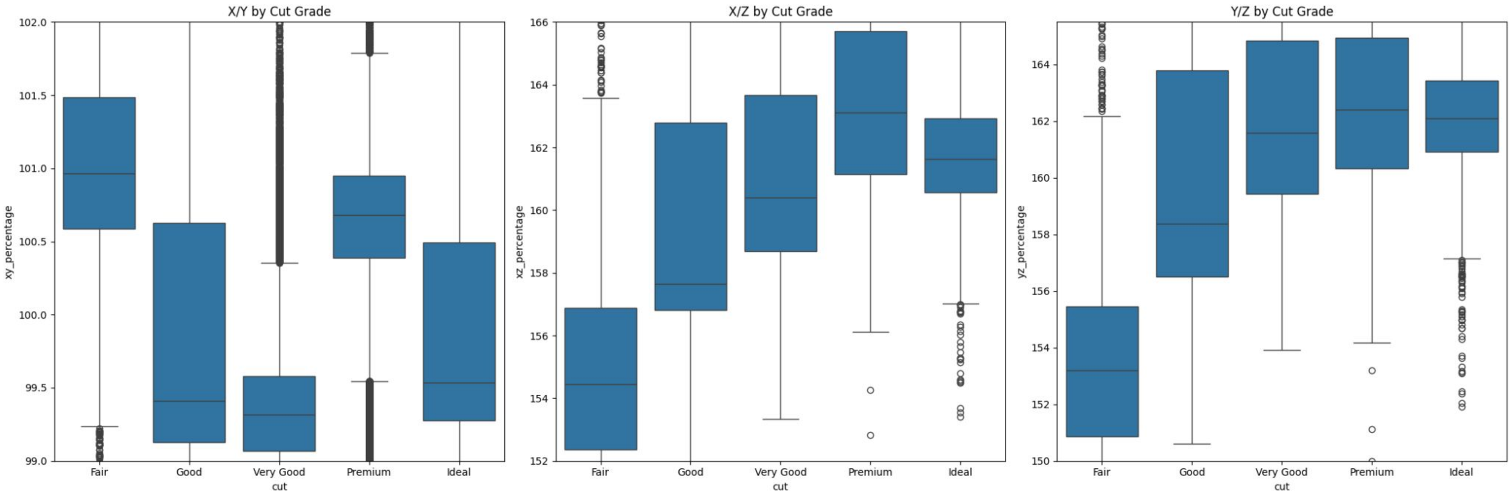
Correlations Heatmap



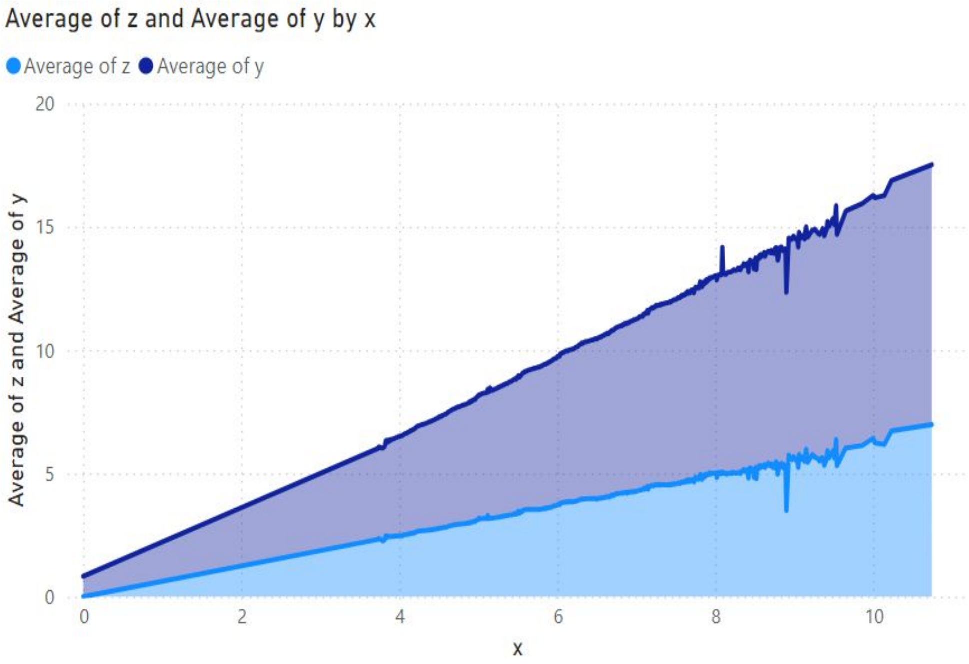
- The color scale indicates correlation strength, ranging from -1 (blue) to 1 (red).
- The variables price, carat, z, x, and y show significant correlations with each other

Diamond Size Ratio by Cut

- Premium and especially Ideal cuts typically exhibit tighter, more consistent dimension ratios, reflecting superior cut quality and precision.



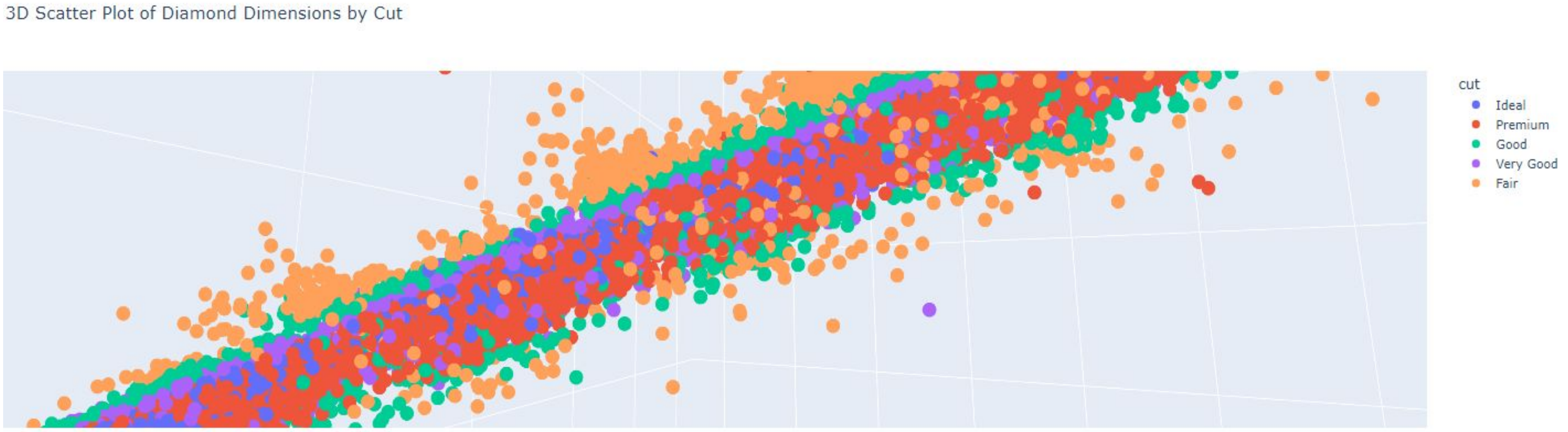
Area chart for diamond size



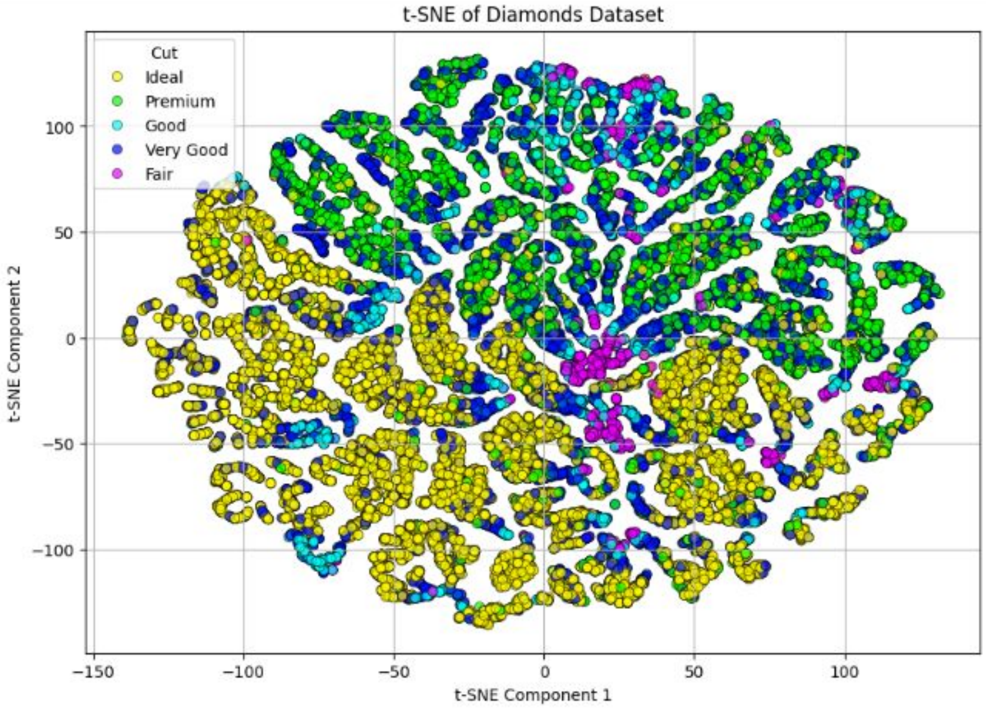
- Observe the constant ratio evolution of each parameter for the diamond size

3D Scatter Plot - Diamond Dimensions by Cut

- We can observe there is a visible dependency that clusters the dimensions into lines, between the variables x, y, z as they evolve in 3D space, as noted in the boxplots we discussed earlier.



Representation using t-SNE



- Represent the diamonds using 2 components resulted from t-SNE from the features for dimensions, 'depth', 'table', 'x', 'y', 'z' and color them by ‘cut’.

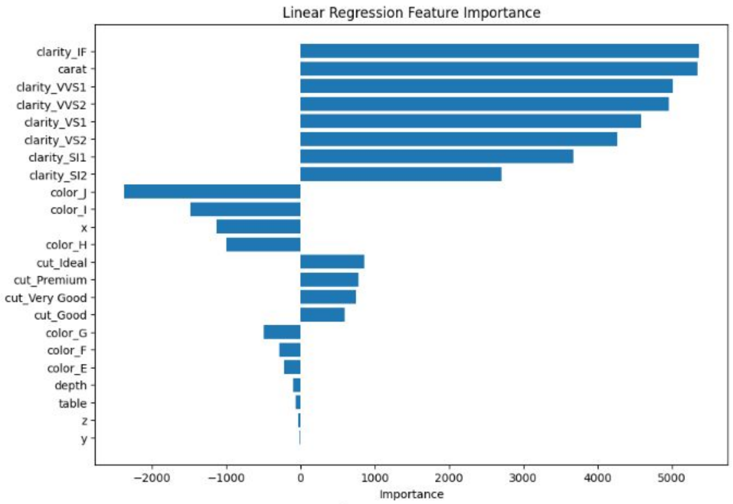
Machine Learning

- Linear Regression
- Decision Tree
- Random Forest
- Support Vector Regression
- Gradient Boosting
- Clustering

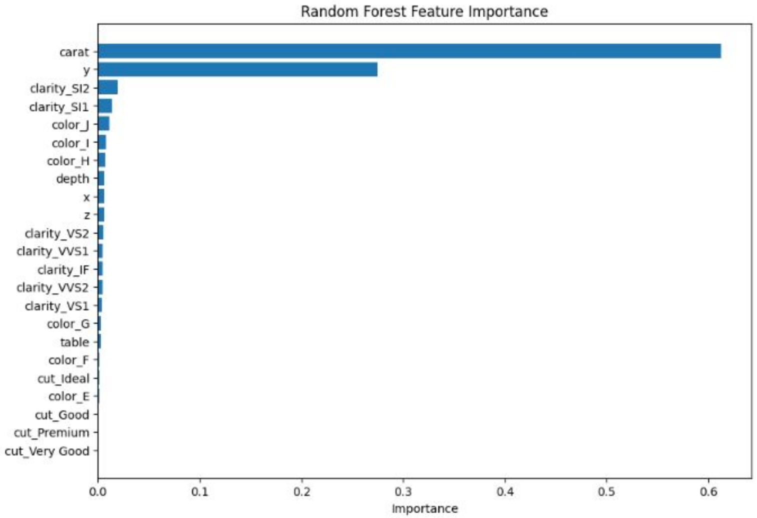
Results

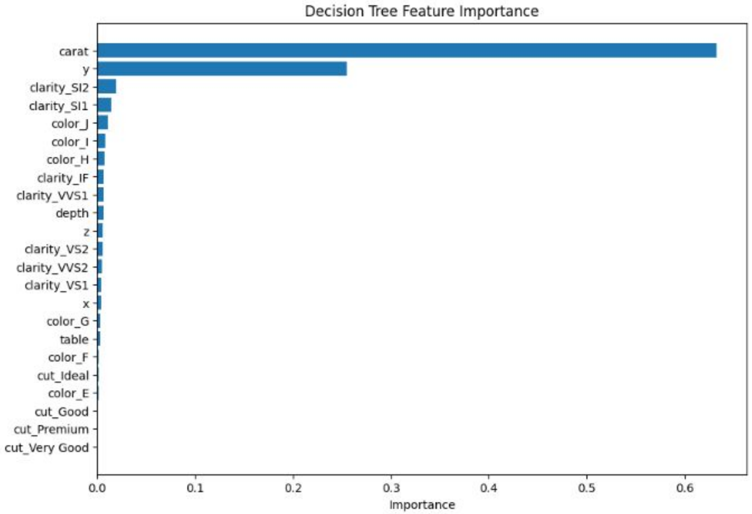
We used for training features from the columns ‘carat’, ‘cut’, ‘color’, ‘clarity’, ‘depth’, ‘table’, ‘x’, ‘y’, ‘z’ in order to predict the ‘price’ column. Bellow we have the metrics for each training

	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	R-squared (R2)	Median Absolute Error
Linear Regression	737.15	1288705.48	1135.21	0.92	526.00
Decision Tree	383.26	716342.19	846.37	0.95	129.00
Random Forest	296.68	408995.77	639.53	0.97	101.49
Support Vector Regression	787.31	2208847.56	1486.22	0.86	357.75
Gradient Boosting	437.14	715907.75	846.11	0.95	192.12

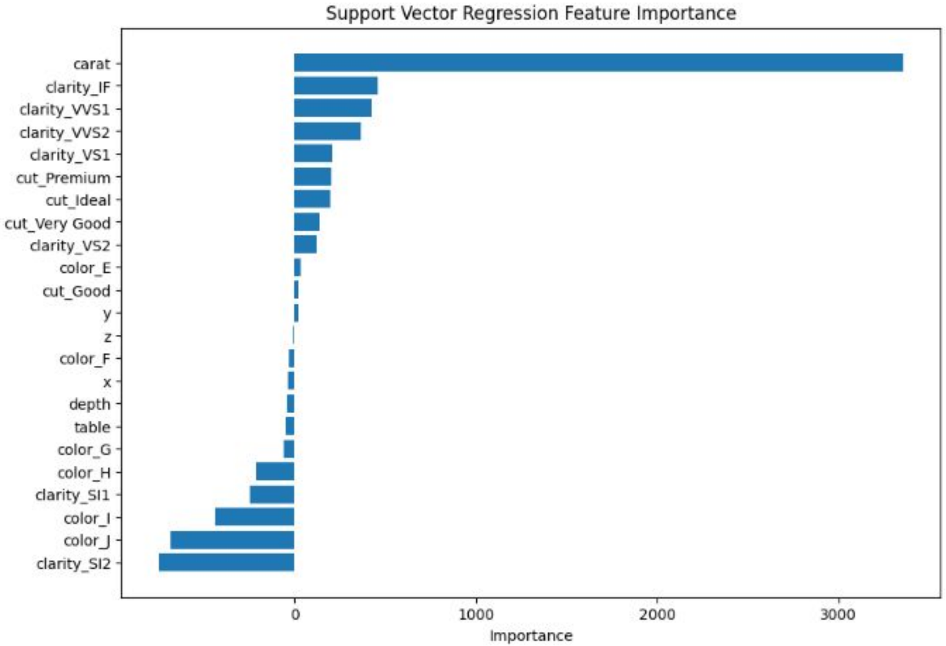


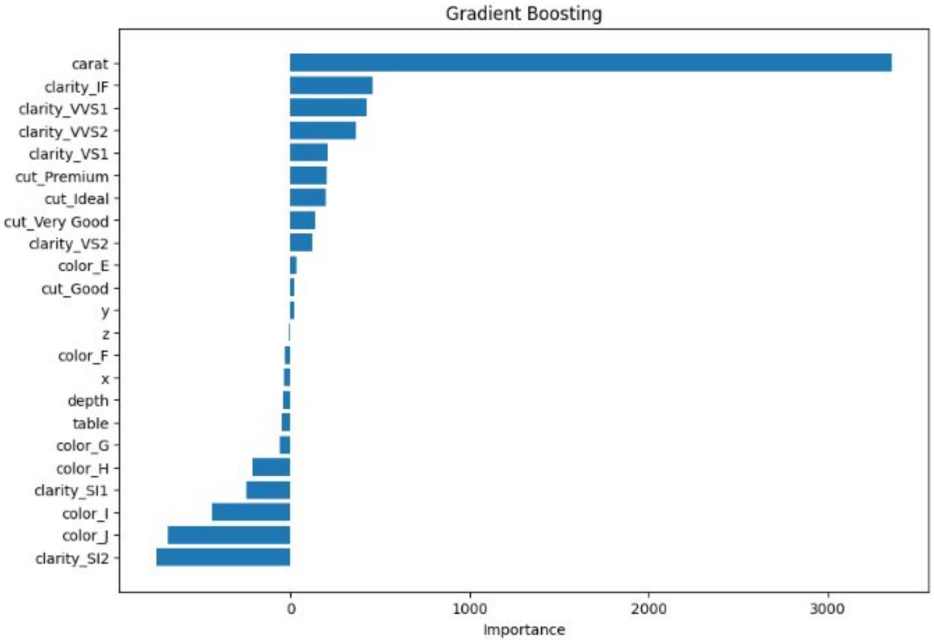
Feature importance





Feature importance





Top important features for each model

Top 5 most important features

	Linear Regression	Decision Tree	Random Forest	Support Vector Regression	Gradient Boosting
0	clarity_IF	carat	carat	carat	y
1	carat	y	y	clarity_IF	carat
2	clarity_VVS1	clarity_SI2	clarity_SI2	clarity_VVS1	z
3	clarity_VVS2	clarity_SI1	clarity_SI1	clarity_VVS2	clarity_SI2
4	clarity_VS1	color_J	color_J	clarity_VS1	x

Top 5 least important features

	Linear Regression	Decision Tree	Random Forest	Support Vector Regression	Gradient Boosting
0	y	cut_Very Good	cut_Very Good	clarity_SI2	cut_Very Good
1	z	cut_Premium	cut_Premium	color_J	cut_Premium
2	table	cut_Good	cut_Good	color_I	cut_Good
3	depth	color_E	color_E	clarity_SI1	table
4	color_E	cut_Ideal	cut_Ideal	color_H	color_E

Thank you