FODS PROJECT PHASE - II

SPARKLE VALUATE

DIAMOND PRICE PREDICTION

Senthamizh Vani A P - 211701049
Meenakshi R - 211701031

CONTENTS

ABSTRACT

The "Diamond Price Prediction" project aims to develop a data-driven model to accurately estimate the price of diamonds based on key features such as carat, cut, color, clarity, and other relevant attributes. Using advanced machine learning algorithms, the model analyzes patterns and trends in historical diamond pricing data to generate precise price predictions. This system can assist gemologists, jewelers, and buyers in making informed decisions, reducing the risk of overpricing or underpricing. By leveraging feature engineering and robust validation techniques, this project aspires to enhance transparency and efficiency in the diamond marketplace, bridging the gap between subjective evaluation and objective assessment.

UNDERSTANDING THE DATASET

Diamond Price Prediction dataset

- The aim of this analysis is to predict the price of diamonds based on their characteristics.
- The dataset used for this analysis is the Diamonds dataset from kaggle.
- The dataset contains 50000 observation and 10 variables.

UNDERSTANDING THE DATASET

Dataset Attributes

- **Carat:** The weight of the diamond, one of the most important factors influencing the price. Larger diamonds are generally more valuable.
- **Cut:** The quality of the diamond's cut, affecting its symmetry and brilliance. Cut categories often include Excellent, Good, Fair, etc.
- **Color:** A measure of how colorless a diamond is, usually graded on a scale from D (colorless) to Z (light yellow or brown).
- **Clarity:** The presence of internal or external flaws (inclusions or blemishes), with a scale ranging from Flawless (FL) to Included (I1, I2, I3).
- **Depth:** The total depth percentage of the diamond, calculated as the depth divided by the average diameter.
- **Price:** The target variable, usually in USD, representing the price of the diamond.

TYPES OF DATA

- **Categorical data**

Attributes such as cut and color represent categorical data because they denote specific categories. These attributes can be used to group and compare attributes of the diamond.

```
#loading the dataset
df = pd.read_csv('diamonds.csv')
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

TYPES OF DATA

- **Numerical data**

In diamond price prediction, numerical data plays a crucial role in determining the value of diamonds based on specific measurable attributes. These attributes are typically numeric and quantitative, which allow machine learning models to predict diamond prices based on patterns.

```
#values count of categorical variables
print(df.cut.value_counts(),'\n',df.color.value_counts(),'\n',df.clarity.value_counts())
```

```
cut
Ideal        19938
Premium      12806
Very Good    11204
Good          4557
Fair          1495
Name: count, dtype: int64
 color
G    10452
E     9085
F     8864
H     7711
D     6224
I     5058
J     2606
Name: count, dtype: int64
 clarity
SI1     12115
VS2     11404
SI2      8519
VS1      7579
VVS2     4694
VVS1     3369
IF       1632
I1        688
Name: count, dtype: int64
```

RELATIONSHIP BETWEEN THE FEATURES

1.**Carat Weight**:Heavier diamonds generally have higher prices, but the relationship is not linear.

2. **Cut Quality:**Higher quality cuts (Fair, Good, Very Good, Ideal, Premium) tend to increase the diamond's price.

3.**Color Grade**:Diamonds with less color (closer to D) are more valuable.

4.**Correlation Matrix:** A matrix showing the correlation coefficients between different features (e.g., carat, cut, color) to identify strong relationships.

.

5.**Feature vs. Target** Analysis:Analyzing how each feature (e.g., carat weight, cut quality) relates to the target variable (price) helps in understanding the impact of each feature on the price.

6.**Feature vs. Price Analysis:**Analyzing how each feature (e.g., carat weight, cut quality) relates to the target variable (price) helps in model building.

7.**Predictive Modeling**:Using machine learning models like Linear Regression, Random Forest, and Gradient Boosting to predict diamond prices with high accuracy.

DATASET DESCRIPTION

- The description of the dataset gives the count, mean, min value, standard deviation etc., for all the numerical values of the dataset. This helps to know the necessary values which can be used for the data cleaning process.

```
#checking descriptive statistics
df.describe()
```

	carat	depth	table	price	x	y	z
count	50000.000000	50000.000000	50000.000000	50000.000000	50000.000000	50000.000000	50000.000000
mean	0.799444	61.753006	57.457830	3944.805440	5.734403	5.737956	3.541056
std	0.475173	1.431088	2.232092	3997.938105	1.123077	1.145579	0.707065
min	0.200000	43.000000	43.000000	326.000000	0.000000	0.000000	0.000000
25%	0.400000	61.000000	56.000000	951.000000	4.710000	4.720000	2.910000
50%	0.700000	61.800000	57.000000	2410.000000	5.700000	5.710000	3.530000
75%	1.040000	62.500000	59.000000	5351.000000	6.540000	6.540000	4.040000
max	5.010000	79.000000	95.000000	18823.000000	10.740000	58.900000	31.800000

DATA PREPROCESSING

Steps involved in Data Preprocessing:

- Data cleaning
- Identifying and removing outliers
- Encoding categorical variables

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 10 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   carat    50000 non-null  float64
 1   cut      50000 non-null  object 
 2   color    50000 non-null  object 
 3   clarity  50000 non-null  object 
 4   depth    50000 non-null  float64
 5   table    50000 non-null  float64
 6   price    50000 non-null  int64  
 7   x        50000 non-null  float64
 8   y        50000 non-null  float64
 9   z        50000 non-null  float64
dtypes: float64(6), int64(1), object(3)
memory usage: 3.8+ MB
```

```
df.shape
```

```
(50000, 10)
```

```
#checking for null values
df.info()
```

EXPLORATORY DATA ANALYSIS

- Exploratory Data Analysis (EDA) is the critical first step in understanding your data. By examining the data’s structure, trends, and relationships, we uncover hidden patterns, anomalies, and initial insights that guide further analysis. EDA allows us to ensure the data quality before building models or drawing conclusions.

–Histograms

–Correlation Matrix

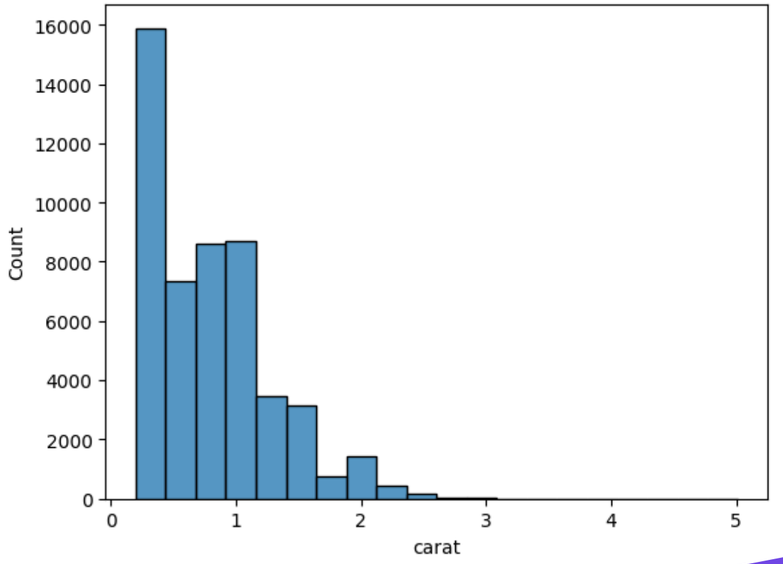
--Scatter plot

HISTOGRAM

- It helps visualize the distribution of diamond prices, providing insights into the data. It reveals whether the prices are skewed, which is common with a large number of lower-priced diamonds and fewer high-priced ones. This can inform decisions like applying transformations (e.g., logarithmic) to normalize the data for better model performance. A histogram also identifies outliers that may need removal or adjustment and aids in selecting meaningful price bins for classification tasks.

```
sns.histplot(df['carat'],bins=20)
```

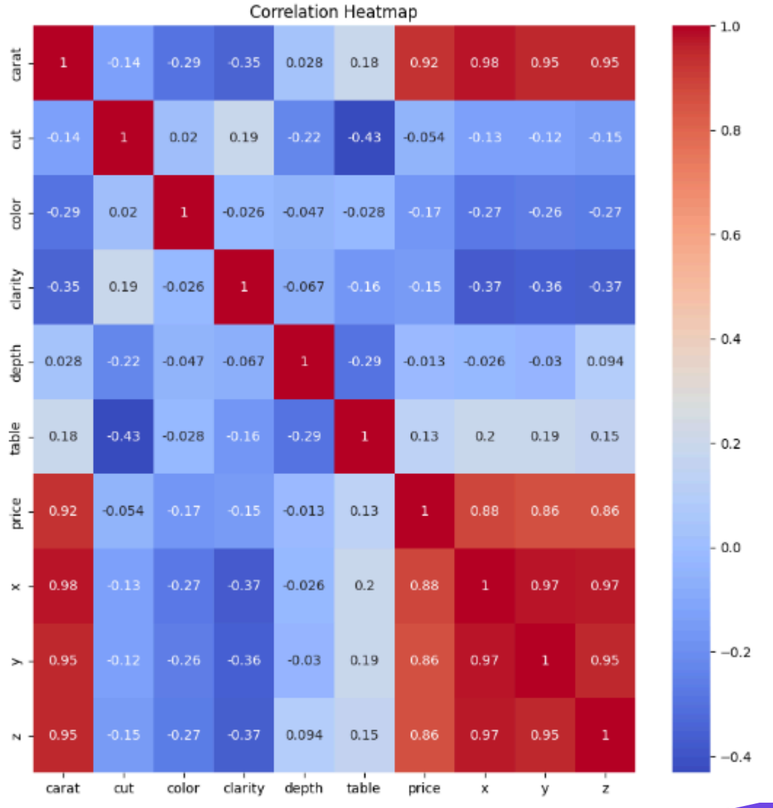
```
<Axes: xlabel='carat', ylabel='Count'>
```



CORRELATION MATRIX

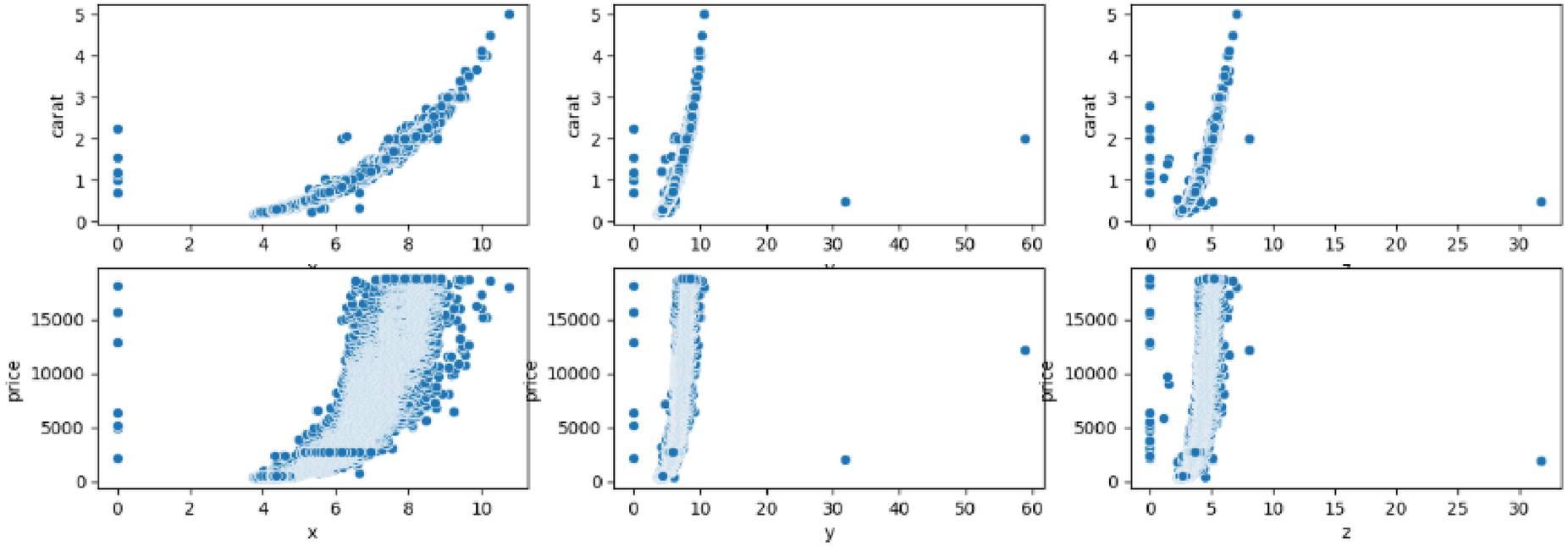
- It shows the correlation coefficients between variables in the dataset. It helps to identify the variables which are more related to each other.

```
#plotting the correlation heatmap
plt.figure(figsize=(10,10))
sns.heatmap(df.corr(),annot=True,cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



SCATTER PLOTS

- Scatter plots visualize relationships between two numerical variables, revealing trends and potential linear or non-linear associations.



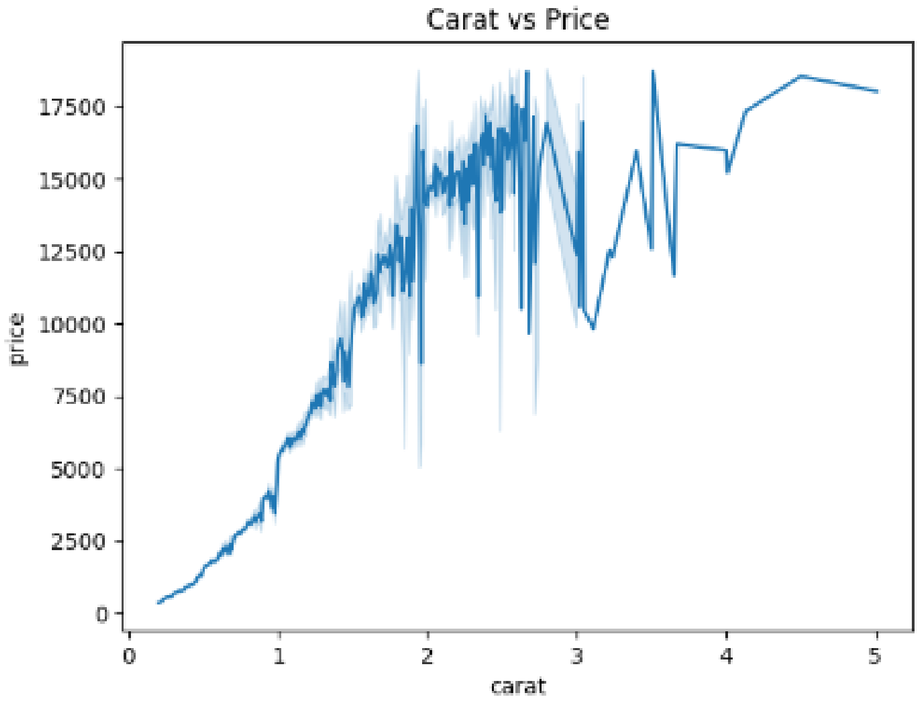
VISUALISATION

Visualization techniques transform data into graphical representations to highlight trends, patterns, and relationships, aiding in better data understanding. Common methods include scatter plots, histograms, box plots, and heatmaps. These visualizations reveal feature importance, data distribution, and correlations.

- Line chart
- Pivot table
- Box plot
- Pie chart
- Bar chart
- Residual plot

LINE CHART

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



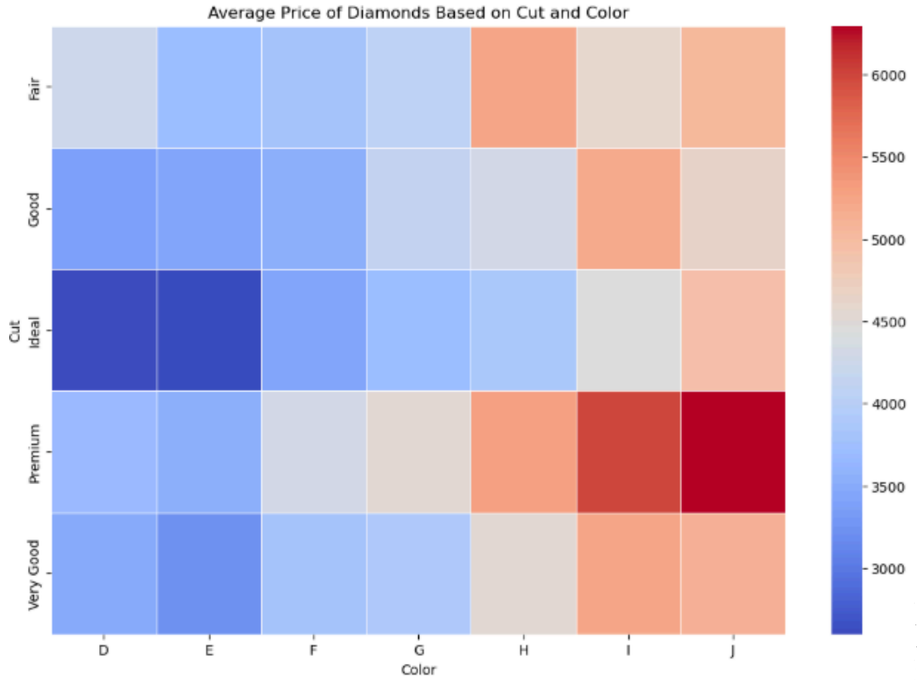
Ploting the relationship between Price and Carat

```
sns.lineplot(x='carat',y='price',data=df)
plt.title('Carat vs Price')
plt.show()
```

PIVOT TABLE

- A pivot table is a data summarization tool that aggregates data based on one or more categorical variables. It’s commonly used for grouping, aggregation, and multi-dimensional analysis.

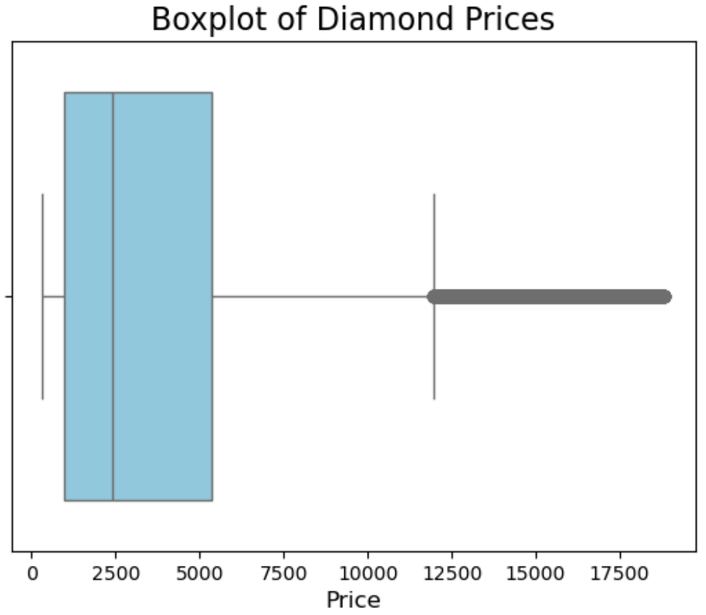
```
pivot_table = df.pivot_table(values='price',
                             index='cut',
                             columns='color',
                             aggfunc='mean',
                             fill_value=0)  # Replace NaN with 0

plt.figure(figsize=(12, 8))
sns.heatmap(pivot_table, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
# Display the pivot table
print(pivot_table)
```



BOX PLOT

- A boxplot is a powerful visualization tool used to summarize the distribution of a dataset and identify outliers.

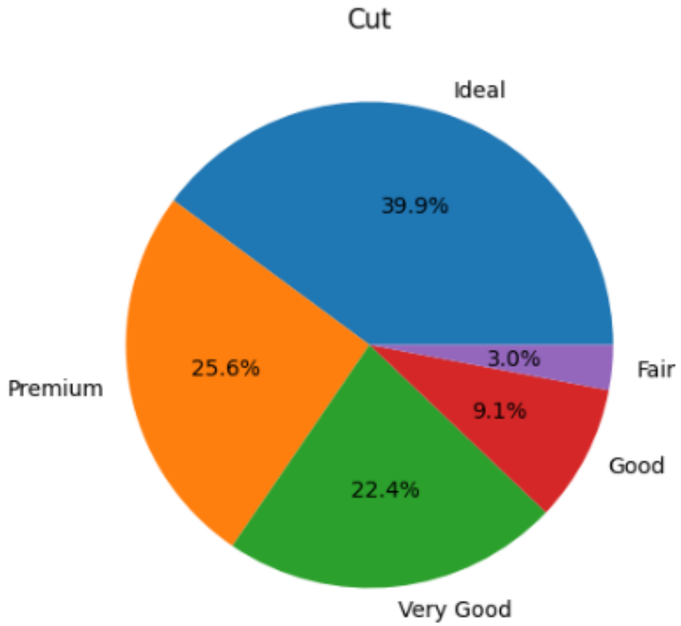
```
sns.boxplot(x=df['price'], color='skyblue')
plt.title('Boxplot of Diamond Prices', fontsize=16)
plt.xlabel('Price', fontsize=12)
plt.show()
```



PIE CHART

- This visual helps to understand how cut quality impacts pricing.

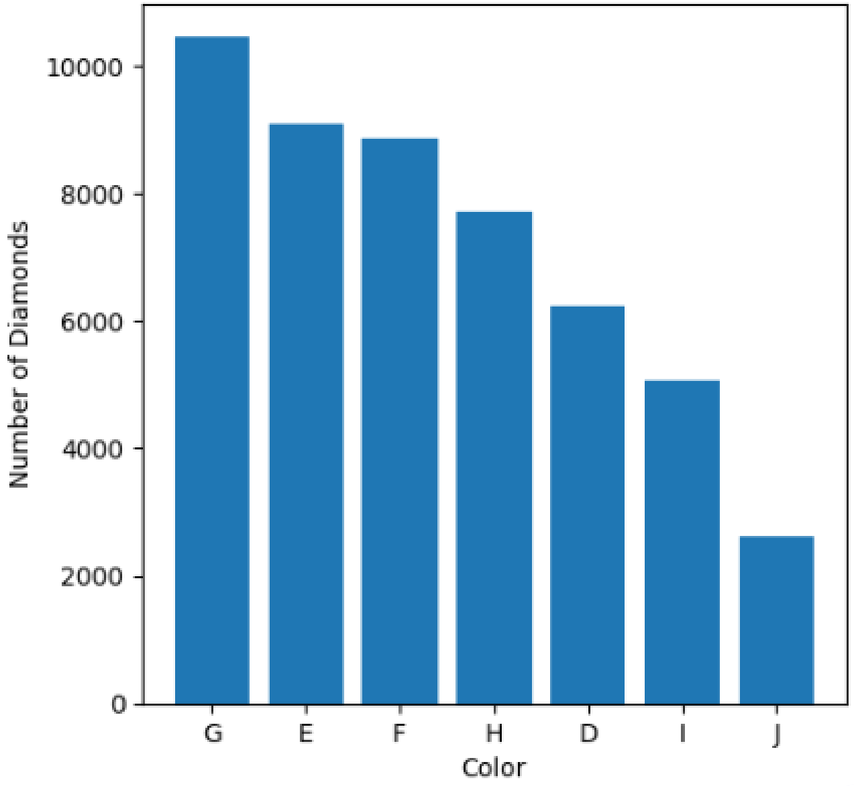
```
plt.figure(figsize=(5,5))
plt.pie(df['cut'].value_counts(),labels=['Ideal','Premium','Very Good','Good','Fair'],autopct='%1.1f%%')
plt.title('Cut')
plt.show()
```



BAR GRAPH

- This graph shows relationship between number of diamonds vs color

```
plt.figure(figsize=(5,5))
plt.bar(df['color'].value_counts().index,df['color'].value_counts())
plt.ylabel("Number of Diamonds")
plt.xlabel("Color")
plt.show()
```

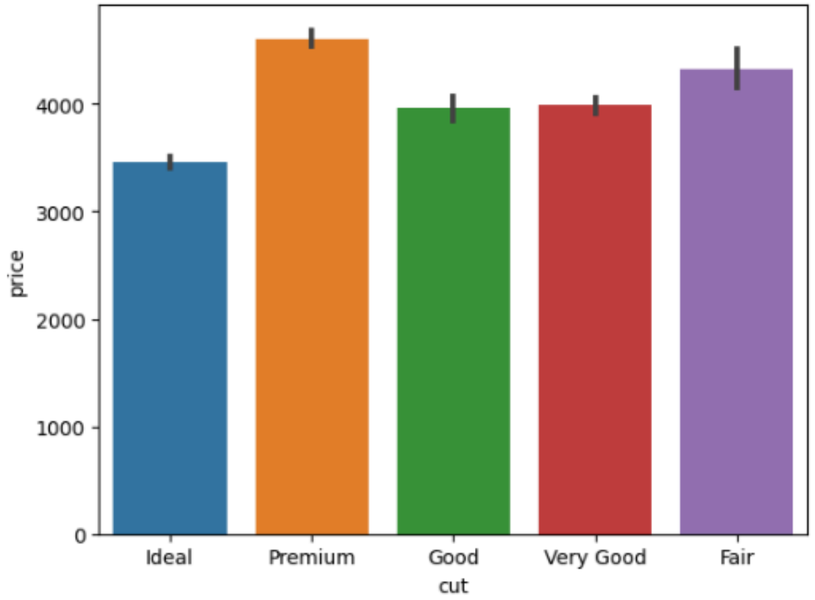


- Show the average or median diamond prices for each category of cut (e.g., Fair, Good, Very Good, Premium, Ideal).
- This helps identify how cut quality impacts pricing.

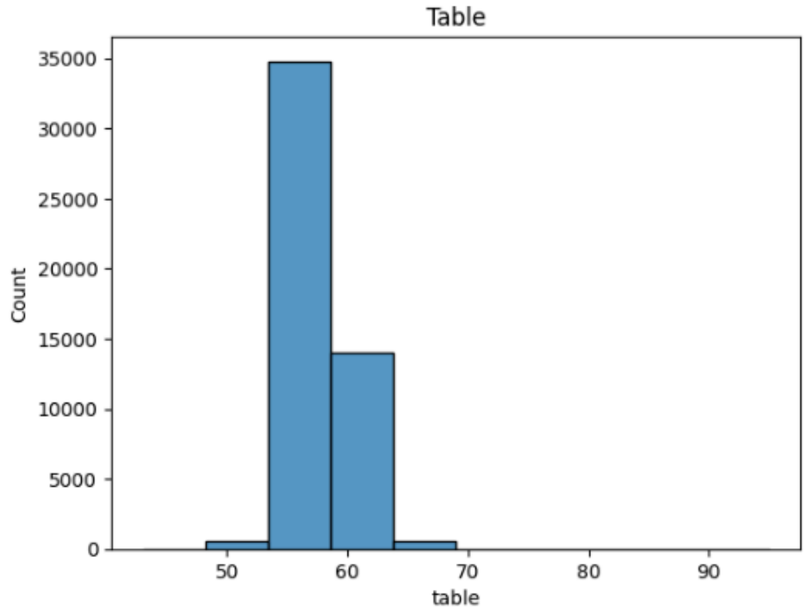
Comparing Diamond's features with Price

```
sns.barplot(x='cut',y='price',data=df)
```

```
<Axes: xlabel='cut', ylabel='price'>
```



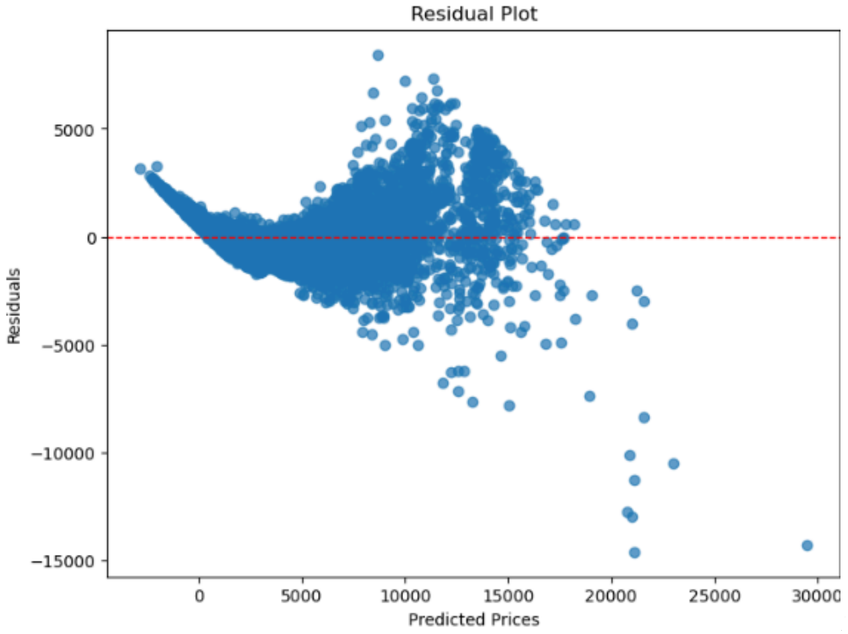

```
sns.histplot(df['table'],bins=10)
plt.title('Table')
plt.show()
```



RESIDUAL PLOT

- Residual plots display the differences between predicted and actual values, helping to identify patterns or biases in predictions.

```python
# Train a Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Calculate residuals
residuals = y_test - y_pred

# Plotting residuals manually
plt.figure(figsize=(8, 6))
plt.scatter(y_pred, residuals, alpha=0.7)
plt.axhline(y=0, color='r', linestyle='--', linewidth=1)
plt.xlabel("Predicted Prices")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()
```



MODEL BUILDING

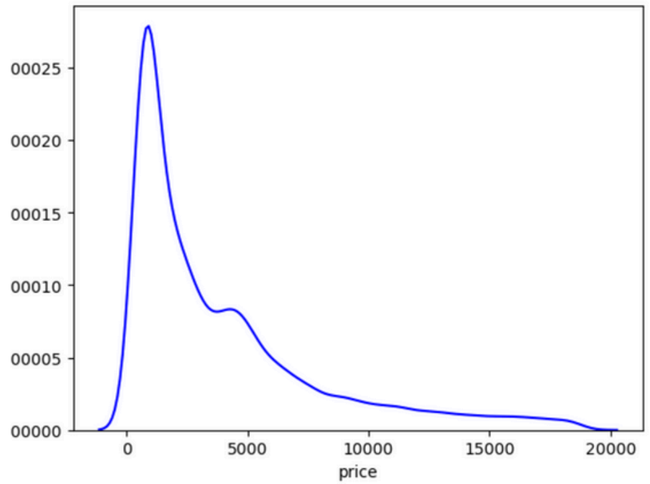
The model uses data science techniques to predict diamond prices based on features like carat, cut, color, and clarity. Steps include data preprocessing (encoding, scaling), exploratory analysis, and applying regression models (e.g., Linear Regression, Random Forests, XGBoost). Feature engineering and hyperparameter tuning ensure accurate, reliable predictions for market use

```
from sklearn.metrics import mean_squared_error,mean_absolute_error
```

Decision Tree Regressor

```
#distribution plot for actual and predicted values
ax = sns.distplot(y_test,hist=False,color='r',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()
```

Decision Tree Regressor



MODEL BUILDING
DECISION TREE REGRESSOR

```
DecisionTreeRegressor
DecisionTreeRegressor()
```

```
#training the model
dt.fit(x_train,y_train)
#train accuracy
dt.score(x_train,y_train)
```

```
0.9999995617234543
```

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor()
rf
```

```
RandomForestRegressor
RandomForestRegressor()
```

```
#training the model
rf.fit(x_train,y_train)
#train accuracy
rf.score(x_train,y_train)
```

```
0.9970489911076822
```

The Decision Tree Regressor predicts diamond prices by creating a tree-based model that splits data on feature thresholds, capturing nonlinear relationships and providing interpretable, efficient predictions for complex datasets.

MODEL EVALUATION

- Model evaluation is the process of assessing how well a trained machine learning model performs on unseen data. It uses metrics like R-squared for regression tasks, or accuracy, precision, recall, and F1-score for classification tasks. Visual tools such as learning curves and residual plots help analyze performance trends, detect overfitting or underfitting, and identify systematic errors. Effective evaluation ensures the model is robust, generalizes well, and meets the problem's requirement.
- Evaluation Metrics

```
print('Random Forest Regressor RMSE:',np.sqrt(mean_squared_error(y_test,rf_pred)))
print('Random Forest Regressor Accuracy:',rf.score(x_test,y_test))
print('Random Forest Regressor MAE:',mean_absolute_error(y_test,rf_pred))
```

```
Random Forest Regressor RMSE: 620.3188867364595
Random Forest Regressor Accuracy: 0.9761202379789445
Random Forest Regressor MAE: 306.1187898892857
```

MODEL EVALUATION

Linear Regression

Linear regression is a statistical method that models the relationship between one dependent variable (price) and one or more independent variables (carat, cut, etc.) using a straight line.

```
# Train linear regression model
linear_model = LinearRegression()
linear_model.fit(X_train, y_train)

# Predict and evaluate
y_pred = linear_model.predict(X_test)
print("Linear Regression MAE:", mean_absolute_error(y_test, y_pred))
print("Linear Regression R²:", r2_score(y_test, y_pred))
```

```
Linear Regression MAE: 979.6395593278766
Linear Regression R²: 0.8566213071802513
```

CONCLUSION

Both the models have almost the same accuracy. However, the Random Forest Regressor model is slightly better than the Decision Tree Regressor model. There is something interesting about the data. The price of the diamonds with J color and I1 clarity is higher than the price of the diamonds with D color and IF clarity which couldn't be explained by the models. This could be because of the other factors that affect the price of the diamond..This insight indicates that while the models perform well with the available data, the complexity of diamond pricing may involve additional features or external factors that are not accounted for in the current analysis. Further exploration and inclusion of more comprehensive data could help refine the models and explain these unexpected pricing patterns more accurately.

THANK YOU!