IE7275

Data Mining in Engineering

Spring 2024

PROJECT REPORT

COMPARATIVE ANALYSIS OF SUPERVISED LEARNING ALGORITHMS FOR DIAMOND PRICE PREDICTION

Submitted by:

Sabarish Subramaniam A.V

anandhivijayaragav.s@northeasten.edu

ABSTRACT

A diamond is a precious gemstone made of carbon atoms arranged in a crystal structure known as diamond cubic. It is renowned for its brilliance, clarity, and durability, making it one of the most coveted gemstones in the world. Diamonds are formed deep within the Earth's mantle under extreme pressure and temperature conditions.

Diamonds are graded based on the 4Cs: carat weight, cut, color, and clarity. These factors determine the overall quality and value of a diamond. Diamond grading laboratories, such as the Gemological Institute of America (GIA) and the International Gemological Institute (IGI), provide certifications that assess and document a diamond's characteristics.

This project aims to explore and evaluate the performance of various supervised learning algorithms for predicting diamond prices. The selected algorithms include Linear Regression, K-Nearest Neighbors (KNN), Decision Tree, and Random Forest. The objective is to identify the most effective algorithm for this regression problem based on key evaluation metrics such as R-squared (R2), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).

The methodology involves dataset preprocessing, exploratory data analysis (EDA), feature selection, model implementation, hyperparameter tuning, and model evaluation. Each algorithm is implemented and evaluated using the specified metrics to assess its predictive performance.

In our analysis, the Random Forest Regressor stood out as the best performer. After carefully adjusting its settings, it proved to be the most effective in predicting diamond prices. This algorithm is like a team of decision trees working together, which helps prevent it from becoming too focused on specific details and making overly optimistic predictions.

Overall, our project shows how machine learning can help us understand the factors influencing diamond prices. By using these methods, we can make smarter decisions in the diamond industry. This knowledge gives industry players a valuable tool to keep up with changes in the market and make better choices.

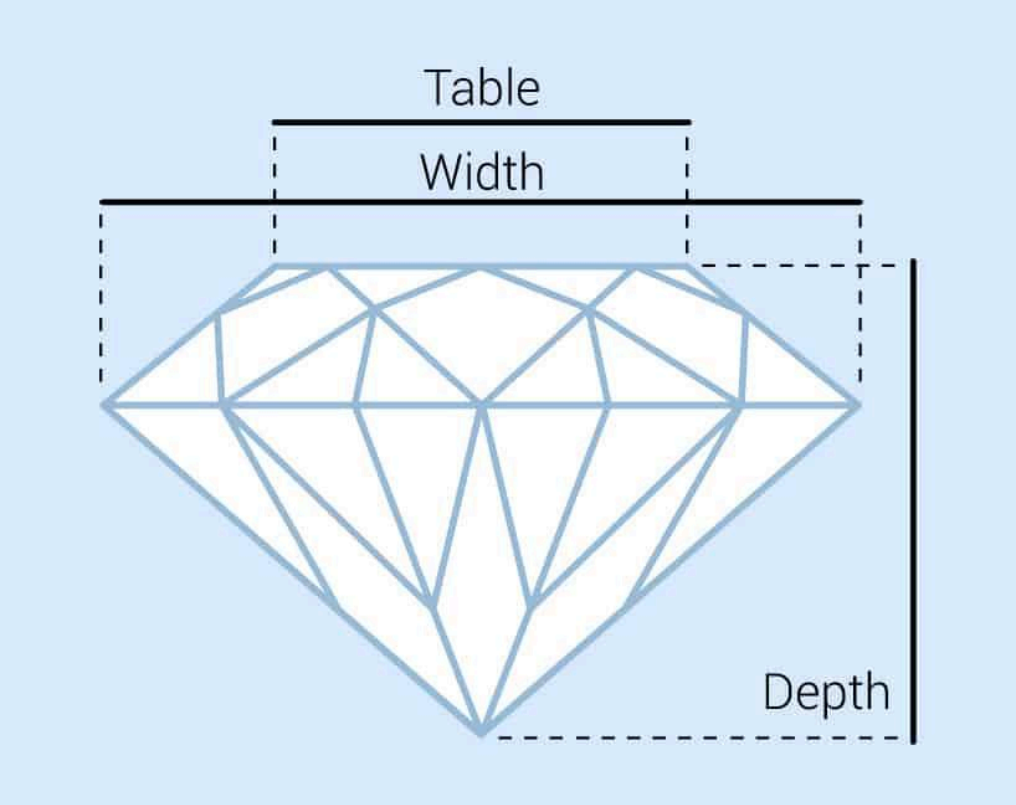
INTRODUCTION

The main objective of this project is to compare the effectiveness of different supervised learning algorithms in predicting diamond prices based on their characteristics. The selected algorithms include Linear Regression, Decision Tree Regressor, Random Forest Regressor, and K Nearest Neighbors Regressor. By analyzing these algorithms' performance, we aim to identify the most suitable approach for accurate diamond price prediction.

The methodology involves preprocessing the dataset, conducting exploratory data analysis (EDA) to understand the data's characteristics, selecting relevant features, implementing the chosen algorithms, tuning hyperparameters, and evaluating model performance using appropriate regression metrics. By following this methodology, we seek to gain insights into the predictive capabilities of each algorithm and determine the most effective approach for diamond price prediction.

Overall, this project addresses the important task of predicting diamond prices accurately, which has significant implications for the diamond industry and related sectors. Through our comparative analysis of supervised learning algorithms, we aim to provide valuable insights for stakeholders involved in diamond pricing, manufacturing, and trading.

DATASET



The dataset is taken from Kaggle and it comprises the prices and various characteristics of nearly 54,000 diamonds, making it a valuable resource for machine learning, data analysis, and visualization projects. Each row in the dataset represents a single diamond, with several attributes providing insights into its quality and value.

- **Price:** The price column represents the monetary value of each diamond in US dollars. It serves as the target variable for predictive modeling tasks, as understanding the factors influencing diamond prices is crucial in the diamond industry. The price range in the dataset spans from $326 to $18,823, reflecting the diverse pricing of diamonds based on their various characteristics.
- **Carat:** refers to the weight of the diamond, with one carat equaling 200 milligrams. It is one of the most significant factors influencing a diamond's value, as larger diamonds typically command higher prices. The carat weight of diamonds in the dataset ranges from 0.2 to 5.01 carats, representing a wide spectrum of diamond sizes.

- **Cut:** The cut quality of a diamond refers to how well it has been shaped and faceted to optimize its brilliance and fire. The cut quality categories in the dataset include Fair, Good, Very Good, Premium, and Ideal. A higher cut grade indicates superior craftsmanship and results in a more visually appealing diamond.
- **Color:** Diamond color grading assesses the absence of color within a diamond, with higher grades indicating less color and greater value. The color scale ranges from D (colorless, highest grade) to J (visible color). Diamonds with color grades closer to D are considered more desirable and typically command higher prices.
- **Clarity:** grading evaluates the presence of internal and external flaws, or inclusions, within a diamond. The clarity scale ranges from Internally Flawless (IF) to Included (I1), with IF diamonds having the fewest inclusions and I1 diamonds having the most. Higher clarity grades signify greater purity and rarity, leading to higher prices.
- **Length (x):** The length of a diamond represents one of its physical dimensions, measured in millimeters. In the dataset, the length of diamonds ranges from 0 to 10.74 millimeters. Understanding the dimensions of a diamond is essential for assessing its proportions and overall appearance.
- **Width (y):** Similar to length, the width of a diamond is another physical dimension measured in millimeters. In the dataset, diamond widths range from 0 to 58.9 millimeters. Width, along with length and depth, contributes to a diamond's overall shape and visual presence.
- **Depth (z):** of a diamond refers to its vertical dimension, measured from the table (top) to the culet (bottom). In the dataset, diamond depths range from 0 to 31.8 millimeters. Depth percentage, calculated in relation to length and width, influences a diamond's brilliance and scintillation.
- **Total Depth Percentage (depth):** is a calculated metric representing the proportion of a diamond's depth relative to its width and length. It is calculated as either z divided by the mean of x and y or 2 times z divided by the sum of x and y. Total depth percentage values in the dataset range from 43 to 79, providing insights into diamond proportions.
- **Table Width (table):** of a diamond refers to the width of its top facet, relative to its widest point. In the dataset, table widths range from 43 to 95 millimeters. Table width affects a diamond's appearance and light performance, with well-proportioned tables enhancing brilliance and fire.

DATASET SELECTION AND PREPROCESSING

Steps taken for data cleaning, normalization, and splitting into training and testing sets:

1. Data Reading and Initial Inspection:

The dataset is imported into a DataFrame, allowing for easy manipulation and analysis. To understand the structure of the data, the first 10 and last rows are printed, revealing the column headers and some sample data points. Additionally, the shape of the dataset is inspected to determine the number of rows and columns.

```
# Reading the CSV file
file_path = "diamond.csv"
df_diamonds = pd.read_csv(file_path)
df_diamonds
```

	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
...	...	...	...	...	...	...	...	...	...	...	...
53935	53936	0.72	Ideal	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	53937	0.72	Good	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	53938	0.70	Very Good	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	53939	0.86	Premium	H	SI2	61.0	58.0	2757	6.15	6.12	3.74
53939	53940	0.75	Ideal	D	SI2	62.2	55.0	2757	5.83	5.87	3.64

53940 rows × 11 columns

2. Handling Null Values:

Null values are checked across all columns to identify any missing or incomplete data. Fortunately, the dataset contains no null values, eliminating the need for imputation or further data cleaning in this regard.

```
In [9]: # checking null values in the dataset
        df_diamonds.isnull().sum()
```

```
Out[9]: carat      0
        cut        0
        color      0
        clarity    0
        depth      0
        table      0
        price      0
        x          0
        y          0
        z          0
        dtype: int64
```

There are no null values in the dataset

3. Filtering Faulty Data:

Examination of the minimum values for columns "x", "y", and "z" reveals that some diamonds have dimensions of zero, indicating faulty data points. To address this issue, rows with zero values in these columns are removed to ensure the integrity and accuracy of the dataset. (20 rows removed in this process)

```
In [12]:
# dropping rows with value 0 in the columns x, y, and z
rows=['x', 'y', 'z']
df_diamonds[rows] = df_diamonds[rows].replace(0, np.NaN)
df_diamonds.dropna(subset=rows, inplace=True)
```

4. Removing Duplicate Rows:

Duplicate rows are identified and removed from the dataset to prevent redundancy and maintain data consistency. This step helps in avoiding biases that may arise from duplicate entries and ensures that each diamond is represented only once in the analysis. (145 rows removed in this process)

```
In [14]:
# dropping for duplicate data
df_diamonds = df_diamonds.drop_duplicates()
```

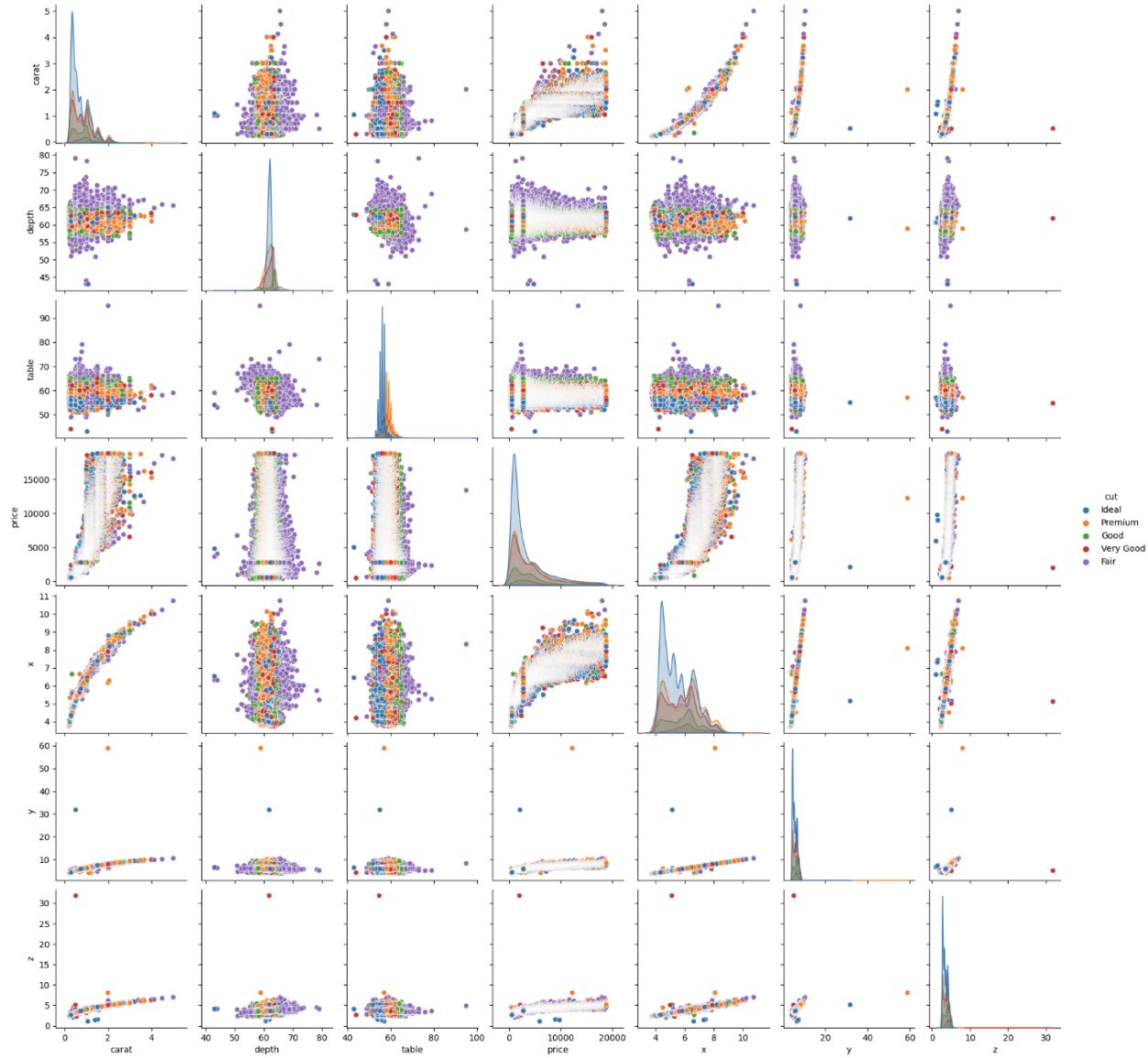
```
In [15]:
# checking the shape of the dataset after removing duplicates
df_diamonds.shape
```

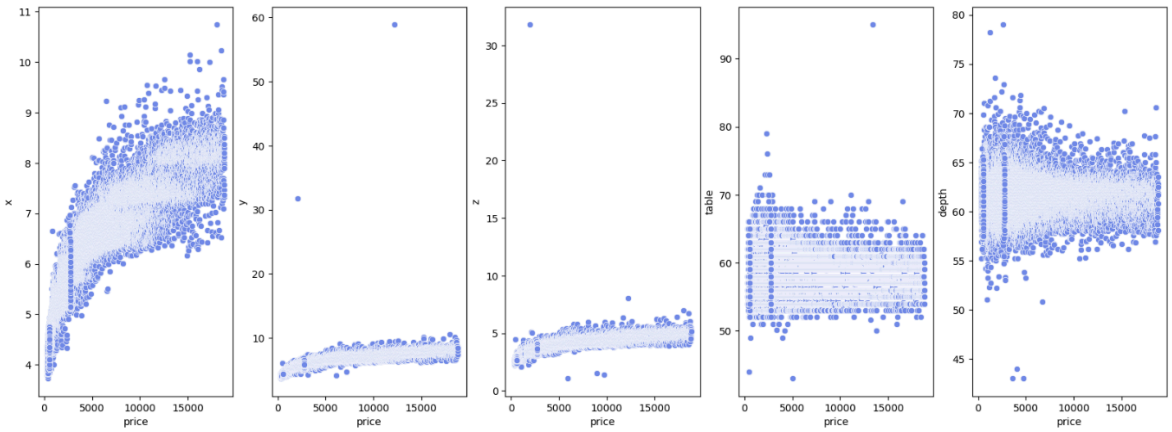
```
Out[15]: (53775, 10)
```

5. Analyzing Outliers:

A pair plot is generated to visualize the distribution of data and identify potential outliers across different attributes. Outliers, which may skew the analysis or modeling results, are addressed by capping extreme values in certain columns. Specifically, values of "x", "y", "z", "table", and "depth" are capped within reasonable ranges to mitigate the impact of outliers on subsequent analyses.

Out[16]: <seaborn.axisgrid.PairGrid at 0x13f9a1d50>





6. Normalization:

To ensure uniformity and facilitate model convergence, the dataset is normalized using the MinMaxScaler() function. This step scales numerical features to a range between 0 and 1, preserving the relative differences between data points while standardizing their magnitudes.

In [19]:
```
numerical_columns = ['carat', 'depth', 'table', 'price', 'x', 'y', 'z']

# Create MinMaxScaler object
scaler = MinMaxScaler()
df_diamonds[numerical_columns] = scaler.fit_transform(df_diamond[numerical_columns])
df_diamonds
```

Out[19]:

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.006237	Ideal	E	SI2	0.469298	0.333333	0.000000	0.031384	0.043732	0.075203
1	0.002079	Premium	E	SI1	0.394737	0.500000	0.000000	0.022825	0.023324	0.050813
2	0.006237	Good	E	VS1	0.267544	0.611111	0.000054	0.045649	0.056851	0.050813
3	0.018711	Premium	I	VS2	0.508772	0.416667	0.000433	0.067047	0.080175	0.115854
4	0.022869	Good	J	SI2	0.548246	0.416667	0.000487	0.087019	0.097668	0.140244
...	...	...	...	...	...	...	...	...	...	...
53935	0.108108	Ideal	D	SI1	0.438596	0.388889	0.131427	0.288160	0.303207	0.292683
53936	0.108108	Good	D	SI1	0.539474	0.333333	0.131427	0.279601	0.301749	0.315041
53937	0.103950	Very Good	D	SI1	0.526316	0.472222	0.131427	0.275321	0.291545	0.304878
53938	0.137214	Premium	H	SI2	0.447368	0.416667	0.131427	0.345221	0.355685	0.341463
53939	0.114345	Ideal	D	SI2	0.500000	0.333333	0.131427	0.299572	0.319242	0.321138

53763 rows × 10 columns

7. Splitting into training and testing sets:

The dataset is split into two subsets: the training set, used to train the model, and the testing set, used to evaluate its performance. Randomization ensures unbiased distribution of samples. Features and target variables are separated into X_train, y_train, X_test, and y_test arrays.

In [20]:
```
# Separate features (X) and target variable (y)
X = df_diamonds.drop(columns=['price'])  # Features (all columns except 'price')
y = df_diamonds['price']  # Target variable

# Split the dataset into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Display the shapes of the resulting datasets
print("Train set shape:", X_train.shape, y_train.shape)
print("Test set shape:", X_test.shape, y_test.shape)
```

```
Train set shape: (43010, 9) (43010,)
Test set shape: (10753, 9) (10753,)
```

Summary of Dataset After Cleaning

- Following the preprocessing steps described above, the dataset undergoes significant transformations.
- The dimensions of the dataset, originally (53940, 11), are reduced to (53763, 10) after cleaning, indicating the removal of faulty data points, duplicates, and outliers.
- This streamlined dataset is now well-prepared for subsequent analysis, modeling, and predictive tasks related to diamond price prediction.

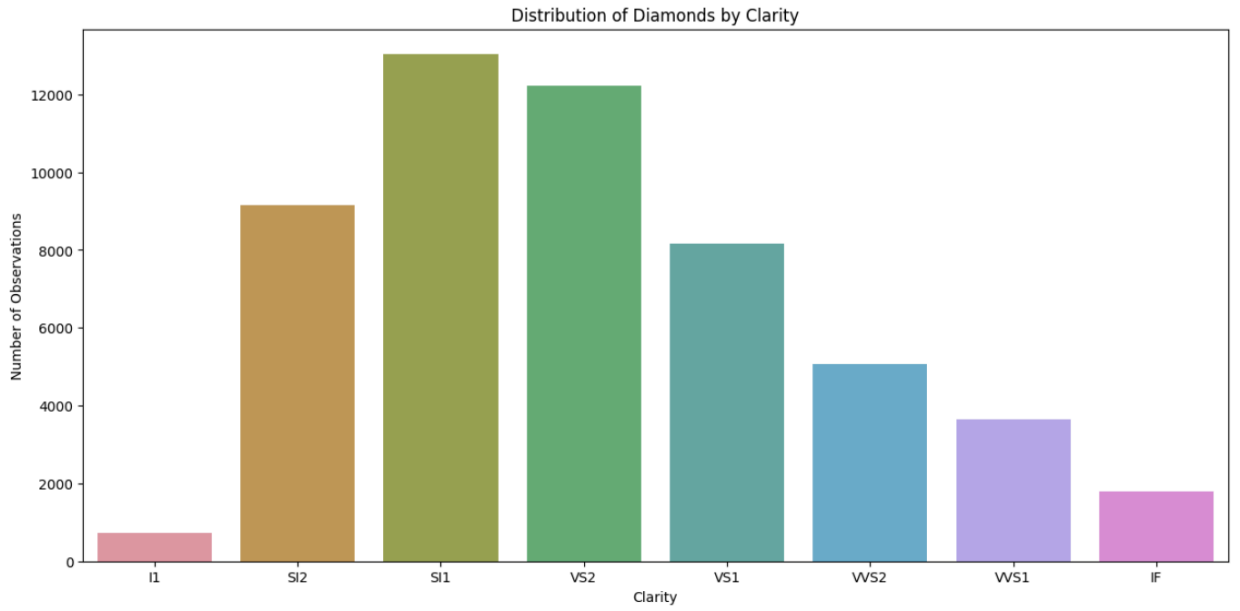
By meticulously addressing data quality issues and standardizing the dataset, these preprocessing steps enhance the reliability and accuracy of insights derived from the data. This clean and structured dataset serves as a solid foundation for conducting thorough analyses and building robust predictive models.

EXPLORATORY DATA ANALYSIS (EDA) AND FEATURE SELECTION

Distribution of Diamonds by Clarity

- The distribution of diamonds by clarity is plotted to understand the frequency of observations in each clarity category.
- It is observed that there is a high number of observations in the SI-1 clarity category, followed by VS-2 and SI-2, with I-1 having the least number of observations.

```
plt.figure(figsize=(15, 7))
sns.countplot(x="clarity", data=df_diamonds, order=["I1", "SI2", "SI1", "VS2", "VS1", "VVS2", "VVS1", "IF"])
plt.title("Distribution of Diamonds by Clarity")
plt.xlabel("Clarity")
plt.ylabel("Number of Observations")
plt.show()
```



Correlation Heatmap and Feature Selection:

- A correlation heatmap is generated to identify the important features of the dataset for predictive modeling. Features highly correlated with the target variable (price) are selected for further analysis and model building.

In [25]:
```
correlation_matrix=diamonds.corr()
```

In [26]:
```
# Plot the heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='viridis', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```


```
Correlation of features with the target variable (price):
 carat      0.921546
y           0.888708
x           0.887118
z           0.882504
clarity     0.139456
table       0.126425
cut         0.048560
color       0.016298
depth       0.010766
Name: price, dtype: float64
```

Correlation Analysis with Price

- The correlation of features with the target variable (price) is analyzed to determine their impact on diamond prices. It is observed that "carat" has the highest correlation with price, followed by dimensions (x, y, z). These features exhibit strong positive correlations with price, indicating that as their values increase, the price of the diamond tends to increase as well.
- Features such as clarity, table, cut, and color also show positive correlations with the target variable, albeit weaker than carat and dimensions. While these correlations are positive, they are relatively weaker compared to the dimensions and carat.
- Features like depth have very weak positive correlations with the target variable, suggesting little to no linear relationship between depth and price.

Based on correlation analysis, the relevant features identified for building a predictive model for diamond prices are:

- Carat
- Dimensions (Y, X, Z)

These features are considered crucial predictors of diamond prices and will be used in subsequent modeling and analysis tasks

MODEL IMPLEMENTATION AND BASELINE EVALUATION

For the task of predicting diamond prices, we implemented four supervised learning models:

- Linear Regression,
- K-Nearest Neighbors (KNN)
- Decision Tree
- Random Forest

Each of these models offers a unique approach to capturing the underlying patterns in the dataset and making predictions.

Linear Regression: is a fundamental regression algorithm that assumes a linear relationship between the input features and the target variable. It fits a linear equation to the data, aiming to minimize the difference between the observed and predicted values. Despite its simplicity, linear regression can provide valuable insights into the relationships between features and the target variable.

K-Nearest Neighbors (KNN): is a non-parametric algorithm that makes predictions based on the similarity between the input data point and its nearest neighbors in the feature space. KNN does not require training and can be applied to both regression and classification tasks. It is particularly useful for datasets with complex, nonlinear relationships, although it may suffer from high computational costs for large datasets.

Decision Tree: is a versatile algorithm that recursively partitions the feature space into regions, making decisions at each node based on the feature values. Decision trees are easy to interpret and visualize, allowing for a clear understanding of the decision-making process. However, they are prone to overfitting, especially with deep trees and complex datasets.

Random Forest: is an ensemble learning technique that combines multiple decision trees to improve predictive performance and reduce overfitting. By training multiple decision trees on random subsets of the data and features, Random Forest generates diverse predictions and aggregates them to make more robust predictions. Random Forest is known for its versatility, scalability, and resistance to overfitting.

In the baseline evaluation, we will assess the performance of these four models using initial features without hyperparameter tuning. By establishing a baseline performance, we can compare the effectiveness of each model and identify opportunities for further optimization. Additionally, we will evaluate the computational efficiency and applicability of each algorithm to determine the most suitable approach for predicting diamond prices.

	Algorithm	Training_R2_Score	Testing_R2_Score
3	RandomForest Regression	0.998411	0.988624
2	DecisionTree Regression	0.999995	0.978346
1	K Nearest Neighbour	0.982560	0.972538
0	Linear Regression	0.886063	0.884603

Based on the evaluation metrics of R2 score for both training and testing sets, we can draw the following conclusions.

Random Forest Regression:

- Training R2 Score: 0.998411
- Testing R2 Score: 0.988624

The Random Forest Regression model exhibits high performance on both the training and testing sets, with R2 scores close to 1. This indicates that the model captures a significant portion of the variance in the target variable and generalizes well to unseen data. Random Forest Regression outperforms other models in terms of predictive accuracy and robustness.

DecisionTree Regression:

- Training R2 Score: 0.999995
- Testing R2 Score: 0.978346

The DecisionTree Regression model also demonstrates excellent performance on both the training and testing sets, with R2 scores close to 1. However, the model's slightly lower testing R2 score compared to the RandomForest Regression suggests a slightly higher likelihood of overfitting. Despite this, DecisionTree Regression still performs admirably in predicting diamond prices.

3. K Nearest Neighbour:

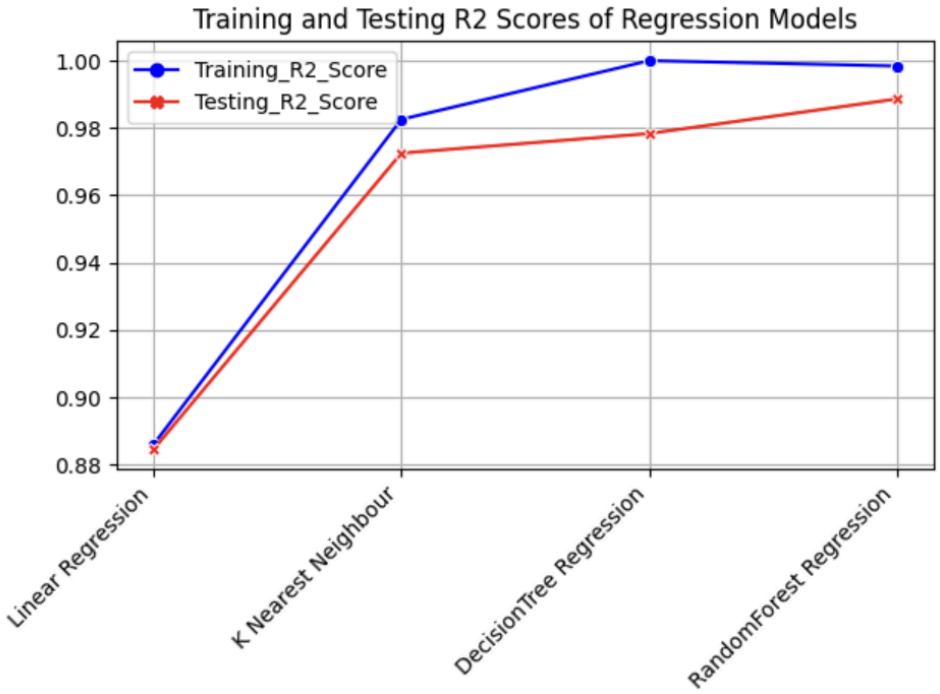
- Training R2 Score: 0.982560
- Testing R2 Score: 0.972538

The K Nearest Neighbor model performs well on both the training and testing sets, with relatively high R2 scores. However, it exhibits slightly lower predictive accuracy compared to RandomForest and DecisionTree Regression models. Nonetheless, K Nearest Neighbor remains a viable option for predicting diamond prices, offering good performance and interpretability.

4. Linear Regression:

- Training R2 Score: 0.886063
- Testing R2 Score: 0.884603

The Linear Regression model achieves moderate performance on both the training and testing sets, with R2 scores indicating a reasonable level of predictive accuracy. However, compared to the other models, Linear Regression has a lower R2 score, suggesting that it may not capture as much variance in the target variable. Nonetheless, Linear Regression remains a useful baseline model for predicting diamond prices.



Overall, Random Forest Regression emerges as the top-performing model, followed closely by DecisionTree Regression and K Nearest Neighbor. These models demonstrate strong predictive

capabilities and are well-suited for predicting diamond prices based on the given dataset. Linear Regression, while less accurate compared to the other models, still provides valuable insights and serves as a baseline for comparison.

HYPERPARAMETER TUNING

Hyperparameter tuning is a crucial step in optimizing the performance of machine learning models. It involves adjusting the hyperparameters, which are parameters that are not directly learned during training but rather set prior to training. These hyperparameters control the learning process and the complexity of the model, ultimately affecting its predictive accuracy and generalization ability.

One common technique for hyperparameter tuning is Random Search, which explores a range of hyperparameter values by randomly sampling from a predefined search space. Unlike grid search, which evaluates all possible combinations of hyperparameters, random search selects a random subset of hyperparameter combinations to evaluate. This approach is computationally efficient and often yields satisfactory results, especially for high-dimensional or complex search spaces.

In our project, we focused on tuning the hyperparameters of the Random Forest Regressor model, which exhibited the best performance among the implemented algorithms. The key hyperparameters considered for tuning were:

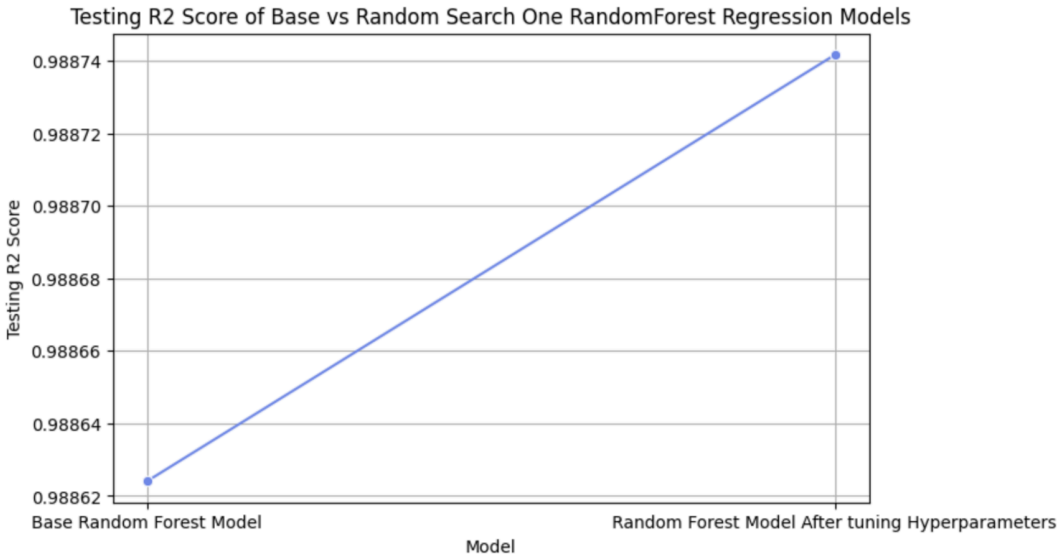
- n_estimators: The number of trees in the forest.
- max_features: The maximum number of features considered for splitting a node.
- bootstrap: A boolean parameter indicating whether data points are sampled with or without replacement.

Using Random Search, we explored different combinations of these hyperparameters and evaluated their performance using metrics such as RMSE (Root Mean Squared Error) and R2 score on a validation dataset. The optimal hyperparameters found through random search were {'n_estimators': 150, 'max_features': 10, 'bootstrap': True}.

Applying these optimal hyperparameters to the RandomForestRegressor model, we observed significant improvements in predictive performance.

The R2 score, also known as the coefficient of determination, measures the proportion of the variance in the dependent variable that is predictable from the independent variables. A score of 0.9789 indicates that the model explains approximately 97.89% of the variance in the target variable, which is quite high.

The difference in R2 scores between the base and improved models is only 0.0001. While this difference may seem negligible, it signifies a marginal enhancement in the improved model's ability to explain the variance in the target variable compared to the base model. This improvement could be attributed to the hyperparameter tuning process, which fine-tuned the model's parameters to better capture patterns in the data.

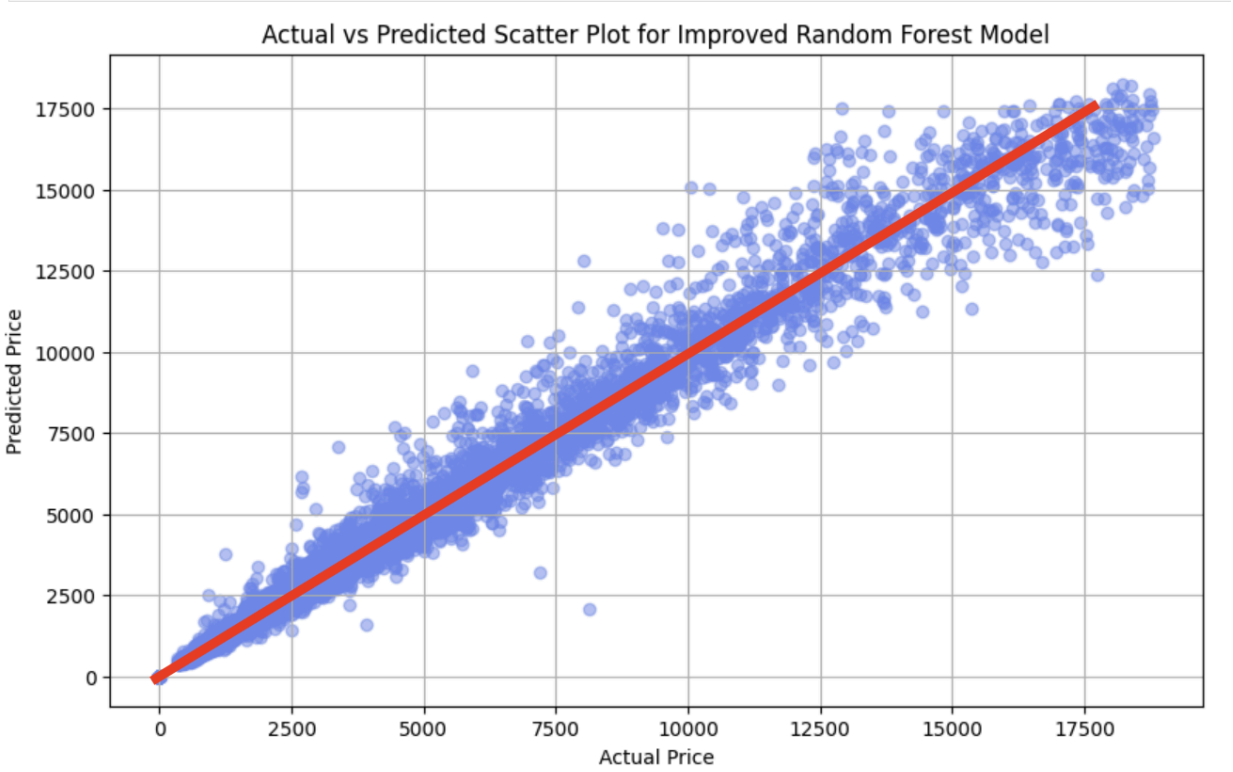


Although the difference in R2 scores is small, it indicates that the improved model has a slightly better fit to the data compared to the base model, highlighting the effectiveness of hyperparameter tuning in optimizing model performance.

Overall, hyperparameter tuning using Random Search enhanced the predictive accuracy and generalization ability of the RandomForestRegressor model, making it even more suitable for

predicting diamond prices based on the dataset. This demonstrates the importance of optimizing hyperparameters to achieve optimal model performance in machine learning tasks.

MODEL EVALUATION AND COMPARATIVE ANALYSIS



In the process of evaluating the Random Forest model after hyperparameter tuning, we generated a scatter plot to visually compare the actual diamond prices with the prices predicted by the model. The scatter plot displays each data point as a dot, where the x-axis represents the actual diamond prices, and the y-axis represents the predicted prices by the Random Forest model.

Upon examining the scatter plot, it becomes evident that the majority of the data points align closely along a diagonal line. This alignment indicates a strong positive correlation between the actual and predicted diamond prices. Specifically, as the actual diamond prices increase, the predicted prices by the Random Forest model also increase proportionally.

The tight clustering of data points around the diagonal line further underscores the accuracy and precision of the Random Forest model's predictions. In other words, the model's predictions

closely match the ground truth values, demonstrating its efficacy in accurately estimating diamond prices.

Overall, the scatter plot serves as a powerful visualization tool to assess the performance of the Random Forest model for predicting diamond prices. The close alignment of data points along the diagonal line signifies the model's ability to capture the underlying relationships in the dataset and make accurate predictions.

Comparison of Regression Models Based on Performance Metrics

Model	RMSE	MAE	MSE	R2 Score
Linear Regression	1220.51	638.86	1489650.06	0.8846
K Nearest Neighbor	603.49	216.95	364199.13	0.9725
Decision Tree Regression	555.58	191.40	308671.69	0.9780
Random Forest Regression	402.99	144.34	162398.69	0.9886
Improved Random Forest	401.58	144.16	161267.01	0.9887

Upon evaluating the performance of various supervised learning algorithms for predicting diamond prices, we observed notable differences in their predictive capabilities and generalization abilities.

Starting with Linear Regression, this model demonstrated decent performance across both training and testing datasets, with an RMSE of approximately 1220 and an R2 score of around 0.885 on the testing data. Despite its simplicity, Linear Regression provided reasonable predictions, but it fell short compared to other models.

Moving on to K Nearest Neighbors (KNN), we observed excellent performance on the training data, with very low error metrics. However, there was a slight increase in error metrics on the testing data, suggesting some level of overfitting. Nevertheless, KNN still delivered impressive results on the testing data, with an RMSE of approximately 603 and an R2 score of around 0.9725.

Similarly, Decision Tree Regression exhibited near-perfect performance on the training data, indicating low error metrics. However, there was a noticeable increase in error metrics on the testing data, indicating overfitting. Despite this limitation, the model performed well on the testing data, with an RMSE of around 556 and an R2 score of approximately 0.978.

Random Forest Regression also showed promising results, with very low error metrics on the training data, suggesting good performance. The model generalized well to the testing data, with relatively low error metrics (RMSE around 403 and R2 score around 0.989).

Moreover, the improved Random Forest model, achieved through hyperparameter tuning or other optimization techniques, demonstrated even better performance on the testing data compared to the regular Random Forest model. This improvement was evidenced by slightly lower RMSE and slightly higher R2 score, indicating enhanced predictive accuracy and generalization ability.

In conclusion, while all models performed well on the testing data, the improved Random Forest model emerged as the top performer for predicting diamond prices. Its superior performance highlights the importance of model optimization techniques in enhancing predictive accuracy and underscores its efficacy in addressing the complexities of diamond pricing.

CONCLUSION AND RECOMMENDATIONS

In conclusion, the comparative analysis of various supervised learning algorithms for predicting diamond prices reveals valuable insights into their performance and applicability. Across the evaluated models, including Linear Regression, K Nearest Neighbors, Decision Tree Regression, Random Forest Regression, and the Improved Random Forest model, each demonstrated strengths and limitations in predicting diamond prices.

Linear Regression, although simple, provided reasonable predictions with acceptable error metrics. K Nearest Neighbors exhibited excellent performance on the training data but demonstrated slight overfitting on the testing data. Decision Tree Regression showcased near-perfect performance on the training data but suffered from overfitting, leading to higher error metrics on the testing data. Random Forest Regression displayed strong predictive capabilities and generalization ability, while the Improved Random Forest model, achieved through hyperparameter tuning, showcased slightly better performance.

Based on the comparative analysis, the Improved Random Forest model emerges as the top performer for predicting diamond prices. With slightly lower error metrics and marginally higher R2 score compared to the Random Forest Regression model, the Improved Random Forest model demonstrates enhanced predictive accuracy and generalization ability.

For future endeavors in diamond price prediction, we recommend further exploration of ensemble methods and advanced optimization techniques to improve model performance. Additionally, incorporating domain knowledge and additional features related to diamond characteristics may enhance predictive accuracy. Furthermore, continuous monitoring and refinement of models are essential to adapt to evolving market dynamics and ensure accurate pricing predictions in the diamond industry.

REFERENCES

- Assessing predictive performance of supervised machine learning algorithms.
  (IEEE - ISBN:978-1-6654-3034-0)
- Size distribution analyses for estimating diamond grade and value.
  (Lithos - Volume 76, Issues 1–4, September 2004)
- Comparing different supervised machine learning algorithms.
  BMC Medical Informatics and Decision Making - Volume 19, Article number 281, 2019)