1

Introduction

With the growing demand for cars fueled by technological advancements and economic growth, the used car market has expanded into a significant sector. This rise has created a vast marketplace not only for new vehicles but also for used cars.However, this market is being governed by a few people and forums who are manipulating the system of pricing. This document aims to predict the price of used cars by developing a model considering factors like brand, model mileage, year, mpg and engine size. It is designed to work for sellers, buyers and manufacturers in the used car market by enabling them get an idea of what they should actually pay or how much their cars would make when sold.The process of model building uses machine learning and data science techniques. Overview , the dataset consist of 100,000 UK Used Car Data set . In order to have the maximum accuracy, several regression methods were used such as linear regression,decision tree regression and random forest regression. Exploratory data analysis was performed on the dataset before training of models.

2

Data Loading

2.1 Data Overview

The dataset comprises scraped data of used car listings, amounting to 100,000 entries. The data has been divided into separate CSV files corresponding to different car manufacturers. This segmentation facilitates focused analysis and processing for each brand. The purpose of collecting this data was initially to create a tool for predicting the resale value of a car. The tool aims to help users determine how much to sell their car for, in comparison to other similar listings on the market. Over time, the dataset was extended to support a more comprehensive car value regression model.

Data Files

Here is a list of the CSV files included in the dataset:

- **audi.csv**: Contains data related to Audi cars.
- **bmw.csv**: Contains data related to BMW cars.
- **cclass.csv**: Contains data related to Mercedes C-Class cars.
- **focus.csv**: Contains data related to Ford Focus cars.
- **ford.csv**: Contains data related to Ford cars.
- **hyundi.csv**: Contains data related to Hyundai cars.

- **merc.csv**: Contains data related to Mercedes cars.
- **skoda.csv**: Contains data related to Skoda cars.
- **toyota.csv**: Contains data related to Toyota cars.
- **unclean cclass.csv**: Contains raw, unclean data for Mercedes C-Class cars.
- **unclean focus.csv**: Contains raw, unclean data for Ford Focus cars.
- **vauxhall.csv**: Contains data related to Vauxhall cars.
- **vw.csv**: Contains data related to Volkswagen cars.

Data Characteristics

- **Size:** The complete dataset covers 100,000 car listings, providing a comprehensive view of the used car market.
- **Segmentation:** The data is organized by manufacturer, allowing for targeted cleaning and analysis.
- **Purpose:** Originally created to assist in determining car resale values, it has been extended to support general car value prediction models.

This dataset provides a rich source of information for developing predictive models and gaining insights into market trends and pricing strategies.

1. Data Inspection

Objective: Understand the structure of each CSV file, including the number of rows, columns, and column names.

Description:

- **File Listing:** The process begins by listing all CSV files in a specified directory. This ensures that all relevant files are identified for further processing.
- **Data Structure Analysis:** For each file, the number of rows and columns are counted, and the column names are extracted. This information provides insight into the dataset's structure and helps identify any discrepancies or missing data.

2. Data Cleaning and Enhancement

Objective: Improve data quality by filling in missing values, dropping redundant columns, and renaming columns for consistency.

Description:

- **Missing Value Handling:**
  - **Filling Strategy:** Missing values in columns such as "mileage," "fuel type," and "engine size" are filled using corresponding backup columns ("mileage2," "fuel type2," "engine size2"). This approach leverages available data to improve dataset completeness.
  - **Dropping Columns:** After filling the missing values, the backup columns are removed from the dataset to eliminate redundancy.
- **Column Renaming:**
  - Ensure column names are standardized and intuitive. For instance, "fuel type" is renamed to "fuelType" to follow camel case convention.
- **Dropping Rows with Missing Values:** Any rows still containing missing values after the filling process are removed to ensure data integrity.

3. DATA MERGING

Objective: Consolidate all cleaned and enhanced CSV files into a single dataset with added brand information.

Description:

- **Brand Addition:**
  - **Strategy:** Extract the brand name from the filename and add it as a new column to each dataset. This information enhances the dataset by providing context about each car's manufacturer.

    Brand = `filename.split(".")[0].capitalize()`
- **File Saving:** The updated datasets are saved in a new directory, preserving the original data while making it more accessible and organized.

- **Data Concatenation:**
  - Combine all individual datasets into a single dataset.
  - The `concat` function is used to merge all data frames, ensuring a seamless integration of data from different files.

    Merged Data = `concat(all_dataframes, ignore_index=True)`

The result is a single CSV file that contains all the cleaned and enhanced car data, complete with brand names, ready for analysis and reporting.

2.2 Data Transformation and Cleaning Process

This step involves data manipulation to fill missing values, standardize columns, and remove inconsistencies. The final dataset is ready for analysis.

1. Loading and Initial Processing

- **Objective**: Load a CSV file containing car data and perform preliminary data cleaning.
- **Steps**:
  - **Load Data**: Read the CSV file into a DataFrame.
  - **Fill Missing Values**: Replace missing values in the 'tax' column with values from the 'tax(Â£)' column.

    ```
    merged_data_new['tax'] =
    merged_data_new['tax'].fillna(merged_data_new['tax(Â£)'])
    ```
  - **Drop Columns**: Remove the 'tax(Â£)' column once its values are merged.

    ```
    merged_data_new.drop(['tax(Â£)'], axis=1, inplace=True)
    ```
  - **Select Columns**: Choose relevant columns for further processing.

    ```
    merged_data = merged_data[columns_of_interest]
    ```
  - **Save Data**: Write the DataFrame to a CSV file.

    ```
    merged_data.to_csv("/path/to/file.csv", index=False)
    ```

2. Cleaning the 'brand' Column

- Standardize the 'brand' column by removing unwanted text.
- **Steps**:
  - **Remove Unwanted Text**: Replace "Unclean " from the 'brand' column.

    ```
    merged_data['brand'] =
    merged_data['brand'].str.replace('Unclean ', '')
    ```

  - **Save Changes**: Update the CSV file with cleaned 'brand' values.

    ```
    merged_data.to_csv("/path/to/file.csv", index=False)
    ```

3. Further Data Cleaning and Normalization

- Clean and normalize additional columns to prepare data for analysis.
- **Steps**:
  - **Standardize 'brand' Values**: Replace specific text values in the 'brand' column.
    * `data['brand'] = data['brand'].str.replace('cclass', 'Cclass')`
    * `data['brand'] = data['brand'].str.replace('focus', 'Focus')`
  - **Remove Invalid Data**:
    * **Filter Data**: Exclude rows where the 'year' is beyond the current year (2024). `data = data.query('year <= 2024')`
    * **Handle Missing Values**:
      - **'engineSize'**: Replace non-numeric and unknown values, then convert to numeric.

        ```
        data['engineSize'] =
        data['engineSize'].replace({'Unknown': '0'})
        data['engineSize'] =
        ```

```
data['engineSize'].str.extract('(\d+\.?\d*)')
```

* **'price'**: Remove non-numeric characters.

```
data['price'] =
data['price'].replace({'[Â£,]':
''},
regex=True).astype(float)
```

* **'mileage'**: Replace 'Unknown' with 0, remove commas, and convert to numeric.

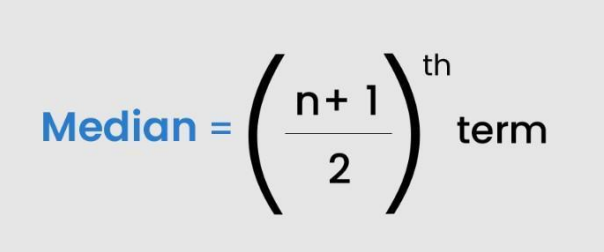
```
·data['mileage'] =
 data['mileage'].replace({'Unknown': '0', ',':
 ''}, regex=True).astype(float)
·data = data.query('(mileage != 0)')
```

- **Validate Fuel Types**: Keep only valid fuel types.

```
- data = data[data['fuelType'].isin(valid_fuel_types)]
```

- **Fill Missing Values**:

  Median Imputation: Replaces missing values with the median of the available values. It is more robust than mean imputation, The pros of median imputation are that it is simple to implement and does not require complex calculations. It is also robust to outliers and can preserve the overall distribution of the data.



  - **'tax'**: Use the median to fill missing values.

```
data['tax'] = data['tax'].fillna(data['tax'].median())
```

  - **'mpg'**: Use the median to fill missing values.

```
data['mpg'] = data['mpg'].fillna(data['mpg'].median())
```

  - **'engineSize'**: Use the median to fill missing values.

```
data['engineSize'] =
data['engineSize'].fillna(data
```

```
['engineSize'].median())
```

- **Filter Invalid 'engineSize' Values**: Exclude rows where 'engineSize' is zero, except for electric cars.

```
data = data.query((engineSize != 0)or(fuelType == "Electric"))
```

- **Calculate Car Age**:

```
data['carAge'] = current_year - data['year']
```

- **Convert Data Types**: Ensure 'year' column is in integer format.

```
data['year'] = data['year'].astype(int)
```

4. **Saving the Final Data** Save the cleaned DataFrame to a new CSV file. Write the final DataFrame to a CSV file. The cleaned and transformed dataset is now saved and ready for further analysis. This process ensures that the data is consistent, accurate, and ready for predictive modeling or other analytical tasks.

3

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is a fundamental step in data analysis that involves examining the dataset to uncover patterns, identify anomalies, and understand the underlying structure of the data. The goal is to summarize the main characteristics of the dataset before diving into more complex analyses. Exploratory Data Analysis (EDA) can be divided into three key parts: Univariate, Bivariate, and Multivariate Exploration. Each focuses on analyzing data at different levels of complexity and interaction.

3.1 Uni-variate Exploration

Uni-variate exploration is a critical first step in EDA that helps in understanding individual variables' distributions and characteristics. By examining numerical and categorical features separately, analysts can identify key patterns, trends, and potential issues that may need further investigation.

3.1.1 **Numerical Features** Numerical features are those that represent quantitative measurements and can be expressed with numbers. Examples include `price`, `mileage`, and `engineSize`. To analyze these features:

1. **Conversion to Numeric Values:**

    - **Purpose:** Ensure that the data is in a numeric format for accurate analysis. Often, datasets may contain numeric values stored as text, which needs

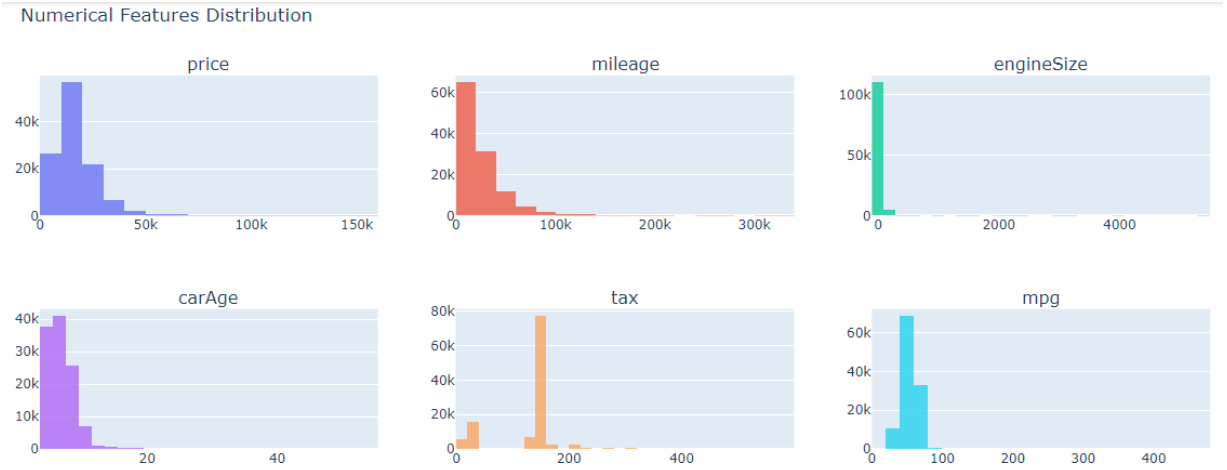
conversion to perform mathematical operations and visualizations. Convert columns that should be numeric into the appropriate type, handling non-numeric values appropriately (e.g., replacing them with NaN or zero).

2. **Handling Missing Values:**

- **Purpose:** Missing data can distort analysis and lead to misleading conclusions. It is crucial to address these gaps.Fill missing values with a specified value, such as zero, to maintain the integrity of the analysis. This approach assumes that missing values are not inherently meaningful but are necessary for completeness.

3. **Histogram Visualization:**

- **Purpose:** Histograms provide a visual representation of the distribution of numerical features. They show how values are spread across different intervals or bins.



- **Interpretation:**
  - **Price Distribution:** Typically, the price of cars is right-skewed, meaning most cars are priced lower, with fewer cars having higher prices.
  - **Mileage Distribution:** Mileage often shows a right-skewed pattern where the majority of cars have lower mileage, and fewer have high mileage.
  - **Car Age Distribution:** This distribution usually peaks at lower ages, indicating that most cars are relatively new, with a tail extending towards

older cars.

- **Engine Size Distribution:** Engine sizes may have peaks at common values, with a gradual decrease in frequency for larger engines. This reflects common preferences or regulatory constraints.
- **Tax Distribution:** Taxes might cluster around specific values, indicating common tax brackets.
- **MPG Distribution:** Fuel efficiency may be concentrated within certain ranges, showing typical values for most cars.

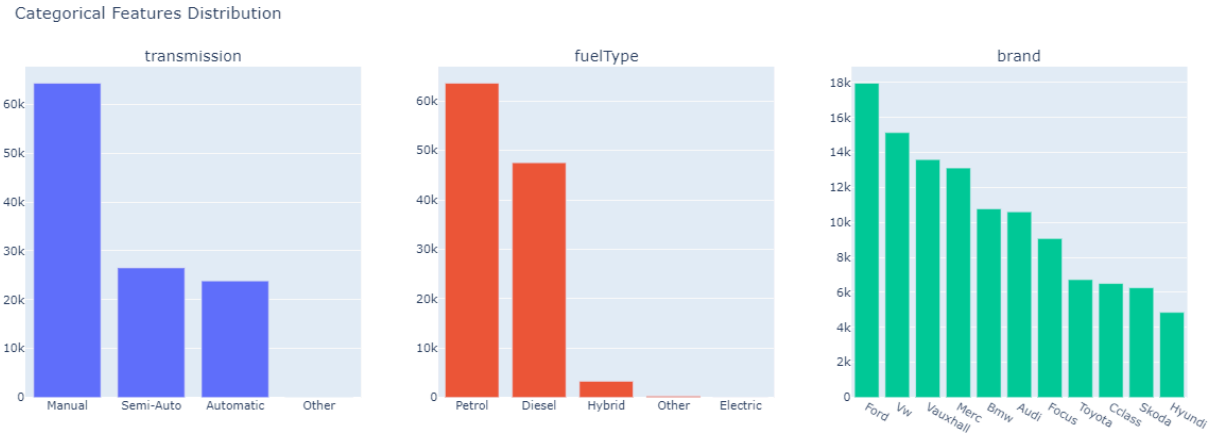
3.1.2 **Categorical Features** Categorical features represent qualitative attributes that can be divided into distinct categories. Examples include `transmission`, `fuelType`, and `brand`. To analyze these features:

1. **Frequency Count:**

   - **Purpose:** Determine how often each category occurs in the dataset. This helps in understanding the distribution and popularity of different categories.
   - **Process:** Count the number of occurrences for each unique category within a feature.

2. **Bar Chart Visualization:**

   - **Purpose:** Bar charts provide a visual representation of the frequency of each category. They help in comparing the relative sizes of different categories.



   - **Interpretation:**

- **Transmission Distribution:** Shows the popularity of different types of transmissions (e.g., automatic vs. manual).
- **Fuel Type Distribution:** Indicates the distribution of various fuel types used in cars (e.g., petrol, diesel).
- **Brand Distribution:** Highlights the frequency of different car brands, showing which brands are most common in the dataset.

3.2 Bi-variate Exploration

Bivariate exploration provides insights into how different variables interact with each other, revealing potential correlations and patterns. By examining numerical and categorical features in relation to the target variable, analysts can identify relationships that may inform further analysis or model development. Bivariate Exploration examines the relationships between pairs of variables in a dataset. This type of analysis helps identify patterns, correlations, and potential causal relationships between features. It is a crucial step in understanding how different variables interact with each other.

3.2.1 **Numerical Features vs. Target Variable** When exploring numerical features against a target variable, such as `price`, scatter plots are often used. These plots help visualize potential correlations or patterns between pairs of numerical variables.

1. **Conversion to Numeric Values:**
    - **Purpose:** Ensure that numerical data is correctly formatted to facilitate accurate plotting and analysis.
    - **Process:** Convert relevant columns to numeric data types, addressing any non-numeric entries by coercing them to a standard placeholder (e.g., `NaN` or zero).
2. **Handling Missing Values:**
    - **Purpose:** Ensure completeness in the dataset to prevent skewed visualizations.
    - **Process:** Fill missing values with zero or another appropriate value to maintain dataset integrity.

3. **Scatter Plot Visualization:**

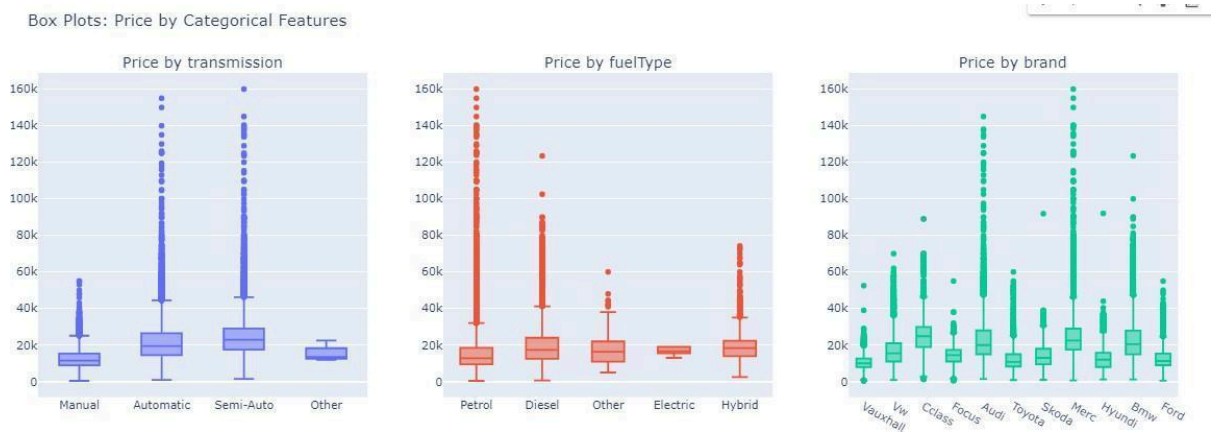
- **Purpose:** Scatter plots depict the relationship between two numerical variables, helping identify trends, clusters, and outliers.

- **Interpretation:**
  - **Mileage vs. Price:** This plot shows how mileage relates to car price. Often, higher mileage may correlate with lower prices, indicating depre- ciation.
  - **Engine Size vs. Price:** Larger engine sizes might be associated with higher prices, but other factors can influence this relationship.
  - **Car Age vs. Price:** Generally, older cars are priced lower, highlighting depreciation over time.
  - **Tax vs. Price:** Tax may have a complex relationship with price, potentially influenced by various factors like car value and regional regulations.
  - **MPG vs. Price:** Fuel efficiency (miles per gallon) can impact car prices, with efficient cars potentially commanding higher prices.

3.2.2 **Categorical Features vs. Target Variable** For categorical features, box plots are used to compare distributions of the target variable (e.g., `price`) across different categories.

Box Plot Visualization:

- **Purpose:** Box plots display the distribution of a numerical variable across different categories, highlighting the median, quartiles, and potential outliers.



- **Interpretation:**
  - **Price by Transmission:** Different transmission types (e.g., automatic, manual) may show varying price ranges, reflecting consumer preferences and market trends.
  - **Price by Fuel Type:** Fuel types (e.g., petrol, diesel, electric) can significantly influence car prices due to factors like fuel economy, environmental concerns, and government incentives.
  - **Price by Brand:** Car brand is a major determinant of price, with luxury brands typically showing higher median prices compared to economy brands.

- **Components of a Box Plot:**
  - **Median (Line):** The central line in the box represents the median price for each category.
  - **Quartiles (Box):** The box spans from the first quartile (Q1) to the third quartile (Q3), indicating the interquartile range (IQR).

- **Whiskers:** Extend to show the range of the data, typically calculated as 1.5 times the IQR.
- **Outliers (Points):** Individual points outside the whiskers represent outliers, indicating prices that deviate significantly from the typical range.

3.3 Multivariate Exploration

Multivariate Exploration examines the relationships among three or more variables simultaneously. This analysis helps uncover complex patterns, interactions, and dependencies that might not be visible when analyzing variables in isolation or pairs. It is a critical step in understanding the holistic nature of the dataset and identifying potential factors influencing the target variable.

3.3.1 **Correlation Analysis** Correlation analysis measures the strength and direction of relationships between pairs of variables.

1. **Selection of Numeric Columns:**
    - Focus on columns that can participate in meaningful mathematical operations, ensuring relevance in correlation analysis.
2. **Computation of Correlation Matrix:**
    - A correlation matrix summarizes the pairwise correlations between multiple variables in a tabular format.
3. **Interpretation of Correlation Coefficients:**
    - **Range:** Correlation coefficients range from -1 to 1.
        - **Positive Correlation (+1):** Indicates a perfect positive relationship; as one variable increases, the other increases.
        - **Negative Correlation (-1):** Indicates a perfect negative relationship; as one variable increases, the other decreases.
        - **No Correlation (0):** No linear relationship between the variables.
    - **Strength of Correlation:**

- **Strong (0.7 to 1 or -0.7 to -1):** Indicates a strong relationship.
- **Moderate (0.3 to 0.7 or -0.3 to -0.7):** Indicates a moderate relationship.
- **Weak (0 to 0.3 or 0 to -0.3):** Indicates a weak relationship.

Theoretical Formulas and Concepts

- **Pearson's Correlation Coefficient:** The Pearson correlation coefficient is a standardized measure of the degree to which two variables move in relation to each other. It's important to note that it only measures linear relationships and may not accurately describe more complex relationships.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Explanation: Measures the linear relationship between two variables x and y.

Components

- x_i and y_i:

  These are the individual data points for variables x and y.

- $\bar{x}$ and $\bar{y}$:
  - These are the means (averages) of the variables x and y, respectively.
  - They are calculated as:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

$$\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$$

- $\sum(x_i - \bar{x})(y_i - \bar{y})$:
  - This is the sum of the products of the deviations of each x_i and y_i from their respective means.
  - It represents the covariance between x and y.
- $\sum(x_i - \bar{x})^2$ and $\sum(y_i - \bar{y})^2$:

- These are the sums of squares of the deviations of x and y from their respective means.
- They represent the variance of x and y.

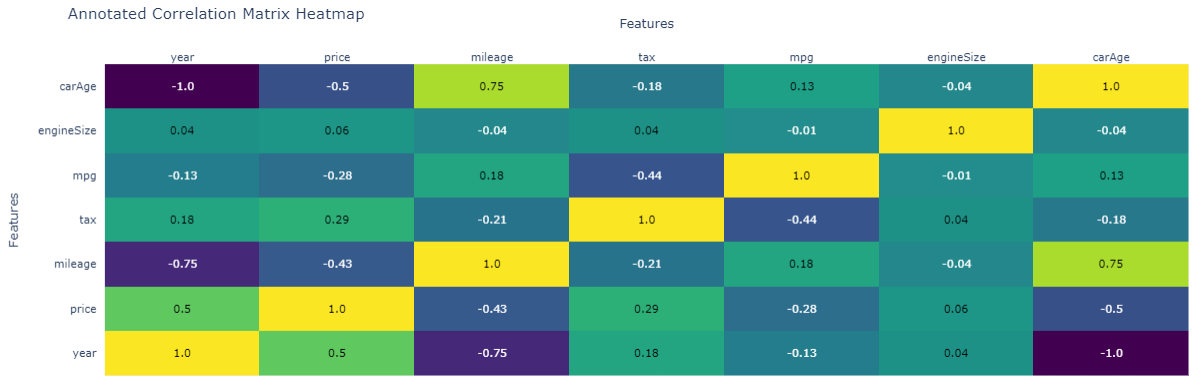
- $\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}$:
  - This is the square root of the product of the variances of x and y.
  - It normalizes the covariance, ensuring the correlation coefficient ranges between -1 and 1.

Interpretation of r

- $r = 1$: Perfect positive linear relationship. As x increases, y increases proportionally.
- $r = -1$: Perfect negative linear relationship. As x increases, y decreases proportionally.
- $r = 0$: No linear relationship between x and y.

3.3.2 **Visualization: Annotated Correlation Matrix Heatmap** A heatmap visu- ally represents the correlation matrix, using color intensity to indicate the strength of relationships.

1. **Annotated Heatmap:**
   - **Purpose:** Provides a visual summary of correlations, making it easy to identify strong, moderate, or weak relationships.
2. **Interpretation:**
   - **Color Scale:** Varies between -1 (strong negative correlation) and 1 (strong positive correlation). Neutral colors (e.g., in the middle of the scale) represent weak or no correlation.
   - **Annotations:** Display numerical correlation values for precise interpretation.
   - **Axes:** Feature names on both axes allow quick identification of variable pairs.



- **Car Age and Year:** A strong negative correlation is indicated (-1.0), suggesting that as the car's age increases, the model year decreases. This is expected because the car age is typically calculated as the difference between the current year and the model year.
- **Car Age and Mileage:** A positive correlation (0.75) suggests that older cars tend to have higher mileage. This aligns with the general expectation that cars accumulate more mileage as they age.
- **Price and Year:** A positive correlation (0.5) between price and year suggests that newer cars tend to be more expensive, reflecting depreciation over time.
- **Price and Mileage:** A negative correlation (-0.43) suggests that cars with higher mileage tend to be less expensive, possibly due to wear and tear.
- **Price and Engine Size:** A small positive correlation (0.06) suggests a weak relationship between engine size and price. This could mean that while larger engines might be more expensive, other factors influence the price more strongly.
- **Tax and MPG (Miles Per Gallon):** A negative correlation (-0.44) between tax and mpg suggests that cars with better fuel efficiency might be in lower tax brackets, possibly due to environmental incentives.

- **Practical Implications:**
  - Understanding these relationships helps in predictive modeling, where features like car age and mileage could be significant predictors of a car's price.

- It also aids in feature selection, allowing you to choose features that are not highly correlated, reducing multicollinearity in models.

Theoretical Documentation: Correlation Network Graph

The purpose of a correlation network graph is to visually represent the relationships between different variables in a dataset. Each variable is depicted as a node, and edges (lines) between nodes represent the correlation between these variables. The strength and direction of the correlation are indicated by the thickness and color of the edges, respectively. This graph helps in identifying patterns, clusters, and significant relationships within the data.

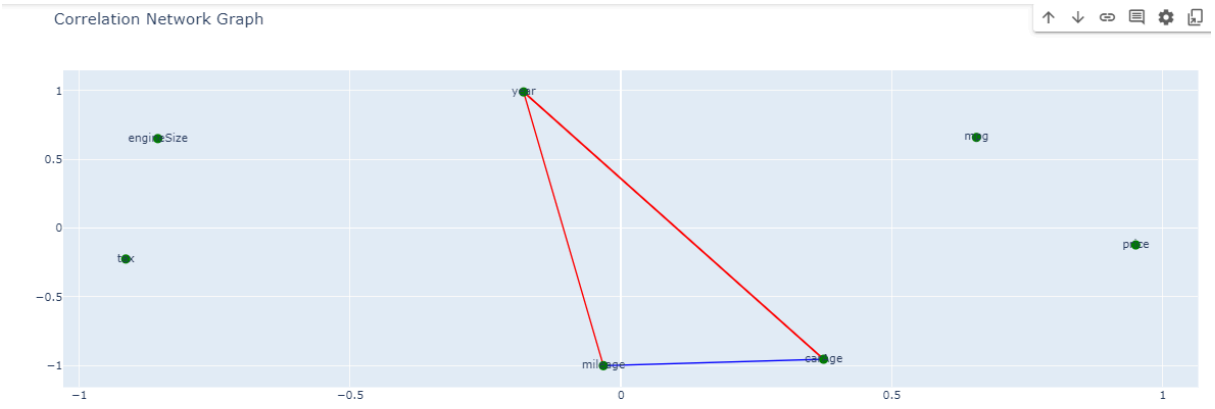
Key Concepts:

1. **Correlation Coefficient** :
    - The correlation coefficient is a measure that indicates the extent to which two variables are linearly related. It ranges from -1 to 1.
    - A positive r indicates a positive correlation, while a negative r indicates a negative correlation.
    - The closer r is to 1 or -1, the stronger the correlation.
2. **Nodes**: Each node in the network graph represents a variable from the dataset. Nodes are positioned in the graph based on a layout algorithm that considers the forces between connected nodes (i.e., nodes with strong correlations).
3. **Edges**:
    - Edges represent the correlation between two variables (nodes).
    - The thickness of an edge reflects the strength of the correlation: thicker edges correspond to stronger correlations.
    - The color of the edge indicates the direction of the correlation: typically, blue edges represent positive correlations, while red edges represent negative correlations.

4. **Threshold for Correlation**: In many cases, a threshold is set to determine which correlations are significant enough to be represented by an edge in the graph. For instance, only correlations with an absolute value greater than 0.5 might be shown, focusing on stronger relationships.

Graph Interpretation:

- **Positive Correlation**: When an edge is blue and thick, it indicates that the two connected variables have a strong positive linear relationship. This means that as one variable increases, the other tends to increase as well.
- **Negative Correlation**: When an edge is red and thick, it indicates a strong negative linear relationship between the two connected variables. This means that as one variable increases, the other tends to decrease.
- **Isolated Nodes**: If a node has no edges connecting it to other nodes, it indicates that the variable has weak or no significant linear correlation with the other variables in the dataset based on the chosen threshold.
- **Clusters**: Groups of nodes that are closely connected by edges may indicate a cluster of variables that are highly correlated with each other. This can suggest that these variables are related in some meaningful way.



Insights from the Graph:

1. **Car Age and Mileage**: There is a strong positive correlation (indicated by a blue line) between `carAge` and `mileage`. This suggests that as the age of the

car increases, the mileage tends to increase as well. This relationship makes intuitive sense, as older cars have generally been driven more, accumulating higher mileage.

2. **Car Age and Year**: There is a strong negative correlation (indicated by a red line) between `carAge` and `year`. This is expected because newer cars have a lower `carAge`. The inverse relationship between the year of manufacture and the age of the car is a direct consequence of how carAge is calculated.

3. **Year and Mileage Negative Correlation**: The red line suggests that as the year increases (i.e., the car is newer), the mileage tends to decrease. This makes intuitive sense because newer cars generally have lower mileage compared to older cars, assuming they have been driven for fewer years.

4. **Potential Impact on Price Prediction**: This negative correlation between `year` and `mileage` might be important when considering predicting price, as newer cars with lower mileage might retain more value. This insight could be valuable for feature engineering in predictive models where car price is the target variable.

5. **Other Features**: The positions of other features like `engineSize`, `mpg`, `tax`, and `price` indicate they might not have strong correlations with each other or the features connected by the lines. These features are isolated with no significant edges connecting them, suggesting they might be independent or weakly correlated with the other variables.

Scatter Matrix Analysis

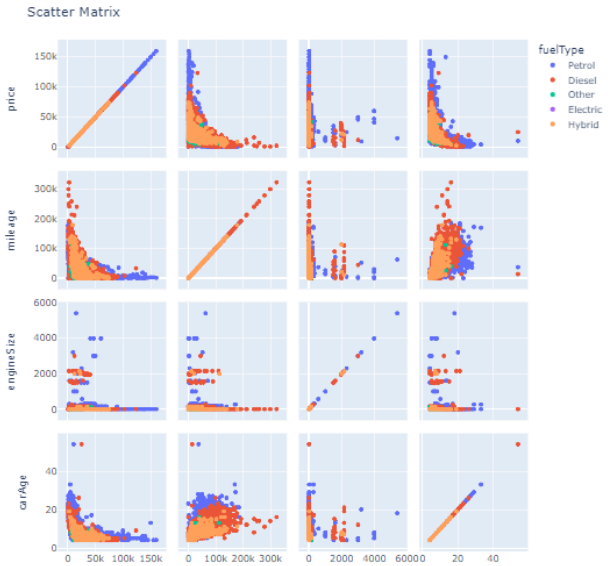
A scatter matrix, also known as a pair plot, is a grid of scatter plots that shows the pairwise relationships between different variables in a dataset. It provides a comprehensive view of the interactions between variables and helps in identifying trends, correlations, and outliers.

Key Concepts:

1. **Scatter Plot**:

- A scatter plot is a graphical representation of the relationship between two variables. Each point on the scatter plot represents an observation from the dataset.
- The position of the point is determined by the values of the two variables for that observation.

- **Dimensions**: In the scatter matrix, each dimension (or variable) is plotted against every other dimension. This results in a grid of scatter plots where the same variable is plotted along both the x-axis and y-axis on the diagonal plots.
- **Color Coding**: The scatter plots are color-coded based on a categorical variable, in this case, `fuelType`. This allows us to see how different categories (e.g., petrol, diesel) distribute across the numerical variables.
- **Diagonal Plots**: The diagonal of the scatter matrix typically shows a histogram or density plot of each variable. These plots help in understanding the distribution of individual variables.



Insights from the Scatter Matrix:

1. **Price vs. Mileage and Car Age**: There is a clear trend of decreasing price with increasing mileage and car age. This suggests that older cars with higher mileage

are generally cheaper. This trend is intuitive, as cars tend to depreciate over time and with use.

2. **Mileage vs. Car Age**: The scatter plot of mileage vs. car age shows a strong positive linear relationship, confirming that older cars tend to have higher mileage. This relationship is expected, as older cars have been on the road longer and are likely to have been driven more.

3. **Engine Size**: The plots involving `engineSize` show no strong linear relationships with the other variables. The distribution of engine sizes appears relatively consistent across different mileage and car age ranges.

4. **Fuel Type Distribution**: The color coding reveals that the distribution of `fuelType` is consistent across the variables. Petrol and diesel are the most common fuel types, and there is no significant difference in their distribution across price, mileage, engine size, and car age.

4

Model Training

1. Data Preprocessing

Data preprocessing is a critical step to ensure that the machine learning models work effectively. The dataset consists of both numerical and categorical variables that require handling before feeding them into models.

1.1 **Categorical Data Encoding** In this dataset, categorical variables like `brand`, `model`, `fuelType`, and `transmission` were converted to numerical labels. This is essential because most machine learning models can only handle numerical inputs.

The encoding used was **ordinal encoding**, where each unique category is assigned a unique integer. For instance:

If `brand` contains categories like "BMW", "Audi", and "Ford", they will be converted to integers such as 0, 1, 2 respectively. This is represented as:

- **brand_label:** Numeric representation of the car brand.
- **model_label:** Numeric representation of the car model.
- **fuelType_label:** Numeric representation of the fuel type (e.g., Petrol, Diesel).
- **transmission_label:** Numeric representation of the transmission type (e.g., Manual, Automatic).

1.2 **Handling Outliers** Outliers can distort the training process, particularly in regres- sion tasks. To remove outliers, the Interquartile Range (IQR) method was used. This method helps identify values that fall far outside the typical range of the data.

Interquartile Range (IQR): The range between the 75th percentile (Q3) and the 25th percentile (Q1) of the data:

$$\text{IQR} = Q_3 - Q_1$$

Lower Bound:

$$\text{Lower Bound} = Q_1 - 1.5 \times \text{IQR}$$

Upper Bound:

$$\text{Upper Bound} = Q_3 + 1.5 \times \text{IQR}$$

Any data points falling outside these bounds were considered outliers and removed. This was applied to all numerical variables like *mileage*, *engineSize*, *carAge*, *tax*, *mpg*, and the target variable *price*.

4.1 Feature Selection and Data Splitting

After prepossessing, the relevant features (predictor variables) were selected for training the models. These features included both numerical and encoded categorical variables.

4.1.1 Features

Numerical Features:

- *mileage:* Distance the car has traveled.
- *engineSize:* Size of the engine in liters.
- *carAge:* Age of the car (calculated from the car's year).
- *tax:* Annual car tax.
- *mpg:* Miles per gallon (fuel efficiency).

Categorical Features (encoded as labels):

- *brand_label:* Encoded representation of the car brand.

- *model_label:* Encoded representation of the car model.
- *fuelType_label:* Encoded representation of the fuel type.
- *transmission_label:* Encoded representation of the transmission type.

The target variable to be predicted was *price*.

4.1.2 Data Splitting

The dataset was split into two parts:

- **Training Set (80% of the data):** Used to train the machine learning models.
- **Test Set (20% of the data):** Used to evaluate the model's performance on unseen data.

This ensures that the model generalizes well to new data and avoids over fitting.

4.1.3 Feature Scaling

To ensure that all features contribute equally during model training, standardization was applied. The `StandardScaler` method was used to scale the features by centering them around zero with unit variance.

The scaling formula is:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma}$$

Where:

- X is the original feature value.
- μ is the mean of the feature.
- σ is the standard deviation of the feature.

By transforming the data this way, features with different scales (e.g., mileage in thousands vs. engine size in liters) are normalized, preventing features with larger scales from dominating the learning process.

4.2 Model Selection and Hyperparameter Tuning

Several machine learning models were used to predict car prices, each with its unique ap- proach to learning from the data. The models were evaluated based on key performance metrics, and hyperparameter tuning was employed to optimize their performance.

4.2.1 Models Used

Linear Regression

Linear Regression models the relationship between the dependent variable price and the independent variables using the linear equation:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Where:

- y is the predicted car price,
- β_0 is the intercept,
- β_1 through β_n are the coefficients of the respective features x_1 through x_n.

Decision Tree

A Decision Tree partitions the data based on feature values to make predictions. It recursively splits the data into branches, making decisions at each node based on a threshold.

Hyperparameters Tuned:

- Maximum depth of the tree (`max_depth`).
- Minimum samples required to split a node (`min_samples_split`).

Random Forest

Random Forest is an ensemble method that builds multiple decision trees and combines their predictions to make a final prediction. The final prediction is obtained by averaging the predictions of all the individual trees:

$$\hat{y} = \frac{1}{T} \sum_{t=1}^{T} y_t$$

Where:

- $\hat{y}$ is the final predicted car price,
- T is the total number of trees,
- y_t is the prediction from the t-th tree.

Hyperparameters Tuned:

- Number of trees in the forest (`n_estimators`).
- Maximum depth of the trees (`max_depth`).
- Minimum samples required to split a node (`min_samples_split`).

Gradient Boosting

Gradient Boosting builds trees sequentially, where each new tree corrects the errors of the previous trees. The final prediction is a weighted sum of the predictions from all trees:

$$\hat{y} = \sum_{t=1}^{T} \alpha_t y_t$$

Where:

- $\hat{y}$ is the final predicted car price,
- α_t is the weight associated with the t-th tree (determined by the learning rate),
- y_t is the prediction from the t-th tree.

Hyperparameters Tuned:

- Learning rate (`learning_rate`): Controls how much each new tree contributes to the overall model.
- Number of trees (`n_estimators`).
- Maximum depth of the trees (`max_depth`).

Neural Network (MLPRegressor)

A neural network model with one or more hidden layers that learns complex relationships between features and the target. The model computes the following for each layer:

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}$$

$$\mathbf{a}^{(l)} = \sigma(\mathbf{z}^{(l)})$$

Where:

- $\mathbf{z}^{(l)}$ is the weighted input for layer l,
- $\mathbf{W}^{(l)}$ is the weight matrix for layer l,
- $\mathbf{a}^{(l-1)}$ is the activation from the previous layer,
- $\mathbf{b}^{(l)}$ is the bias vector for layer l,
- $\sigma(\cdot)$ is the activation function.

Hyperparameters Tuned:

- Number of hidden layers and neurons (`hidden_layer_sizes`).
- Activation functions (`activation`): Functions that introduce non-linearity into the model.
- Solver (`solver`): Algorithm used for weight optimization.
- Learning rate (`learning_rate`): Controls the step size during optimization.

4.3 Model Evaluation

Each model was evaluated using cross-validation and the following key performance metrics:

Cross-validation is used to ensure that the model's performance is robust and not specific to a particular data split. By splitting the data into multiple folds, training on a subset, and validating on the remaining fold, this method provides a better understanding of how the model will perform on unseen data.

In this project, 5-fold cross-validation was employed, which helps in mitigating overfitting and provides a more reliable estimate of model performance across different hyperparameters. Cross-validation results were evaluated using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared.

4.3.1 Performance Metrics

Mean Absolute Error (MAE):

MAE measures the average magnitude of the errors in predictions, without considering their direction. It is given by:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Where:

- n is the number of data points,
- y_i is the actual value for the i-th data point,
- $\hat{y}_i$ is the predicted value for the i-th data point.

Mean Squared Error (MSE):

MSE penalizes larger errors by squaring the residuals. It is defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where:

- n is the number of data points,
- y_i is the actual value for the i-th data point,
- $\hat{y}_i$ is the predicted value for the i-th data point.

Root Mean Squared Error (RMSE):

RMSE provides the square root of the MSE, giving a result in the same units as the target variable:

$$\text{RMSE} = \sqrt{\text{MSE}}$$

R-squared (R^2):

R-squared represents the proportion of variance in the target variable explained by the features. It is given by:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where:

- y_i is the actual value for the i-th data point,
- $\hat{y}_i$ is the predicted value for the i-th data point,
- $\bar{y}$ is the mean of the actual values,
- n is the total number of data points.

An R^2 value close to 1 indicates that the model explains a large portion of the variance in the target variable.

4.4 Model Results

Model Results and Analysis

Linear Regression

- **Mean Absolute Error (MAE):** 3760.45
- **Mean Squared Error (MSE):** 24,464,170.50
- **Root Mean Squared Error (RMSE):** 4946.13
- **R-squared:** 0.56

Analysis: Linear Regression is a simple and interpretable model but may not capture the complexities of non-linear relationships in the data. The R-squared value indicates that the model explains 56% of the variance in the target variable, which is moderate but leaves room for improvement. The relatively high RMSE suggests that the model might not be well-suited for this dataset compared to more sophisticated models.

Decision Tree

- **Best Parameters:** `{'max_depth': 20, 'min_samples_split': 10}`
- **Mean Absolute Error (MAE):** 1405.93
- **Mean Squared Error (MSE):** 4,879,623.68

- **Root Mean Squared Error (RMSE):** 2208.99

- **R-squared:** 0.91

Analysis: The Decision Tree model performs well with an R-squared of 0.91, explain- ing 91% of the variance in the target variable. The tuning of hyperparameters such as `max_depth` and `min_samples_split` helps to balance bias and variance. The model's error metrics are significantly lower compared to Linear Regression, making it a good candidate for further exploration.

Random Forest

- **Best Parameters:**

  `{'max_depth':20, 'min_samples_split':5, 'n_estimators':200}`

- **Mean Absolute Error (MAE):** 1,243.13
- **Mean Squared Error (MSE):** 3,694,212.64
- **Root Mean Squared Error (RMSE):** 1,922.03
- **R-squared:** 0.93

Analysis: Random Forest outperforms Decision Tree by reducing the model's variance while maintaining strong performance. The cross-validation process helps to optimize the number of estimators and depth of trees. With an R-squared of 0.93, Random Forest demonstrates strong predictive power, while the low MAE and RMSE suggest that it handles errors more effectively.

Gradient Boosting

- **Best Parameters:**

  `{'learning_rate': 0.2, 'max_depth': 7, 'n_estimators': 200}`

- **Mean Absolute Error (MAE):** 1239.43
- **Mean Squared Error (MSE):** 3,392,095.97
- **Root Mean Squared Error (RMSE):** 1841.76
- **R-squared:** 0.94

Analysis: Gradient Boosting shows the best performance among the models, achieving an R-squared of 0.94. The fine-tuning of hyperparameters, particularly `learning_rate` and `max_depth`, allows the model to fit the data more closely without overfitting. The RMSE of 1841.76 indicates strong predictive accuracy. Its ability to capture complex relationships in the data makes it a strong candidate for deployment.

Neural Network

- **Best Parameters:**

  ```
  {'activation':'relu',
  'hidden_layer_sizes':(50,),
  'learning_rate':'constant', 'solver':'adam'}
  ```

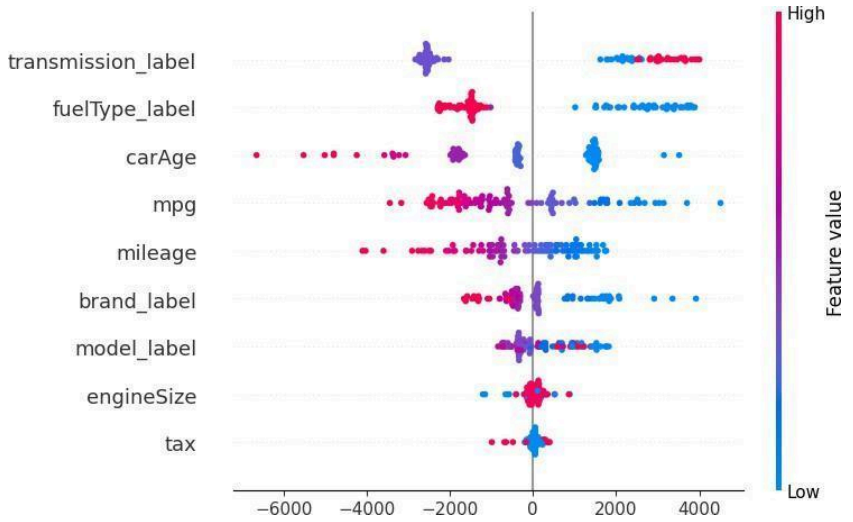
- **Mean Absolute Error (MAE):** 2827.40
- **Mean Squared Error (MSE):** 14,544,824.80
- **Root Mean Squared Error (RMSE):** 3813.77
- **R-squared:** 0.74

Analysis: The Neural Network model, despite its complexity, underperforms compared to Decision Tree, Random Forest, and Gradient Boosting models. While it achieves an R-squared of 0.74, the higher MAE and RMSE suggest that it struggles with the prediction task, possibly due to overfitting or an inadequate number of training iterations. Further tuning of the neural network structure might improve performance.

4.4.1 SHAP Analysis for Model Interpretation

The SHAP summary plot visually illustrates the impact of each feature on the target variable. Each point on the plot represents a Shapley value for a feature and an instance. The color gradient (from blue to red) indicates the feature's value (low to high).

Results and Insights The SHAP summary plot highlights the most influential features in predicting car prices, and the direction of their impact. Here's a breakdown of the key findings:



- **Transmission Type (transmission_label):** This feature emerges as the most influ- ential in the model's predictions. The plot shows that the type of transmission strongly impacts the car price, with higher SHAP values indicating a positive contribution.
- **Fuel Type (fuelType_label):** Fuel type also plays a critical role, with different fuel types affecting car prices significantly. The plot reflects varying contributions, with both positive and negative SHAP values depending on the fuel type.
- **Car Age (carAge):** As expected, the age of the car has a considerable negative impact on the price. Older cars reduce the predicted value, while newer cars contribute positively.
- **Mileage:** The car's mileage is inversely related to its price, with higher mileage pushing the prediction towards a lower price, as shown by the negative SHAP values.
- **Engine Size (engineSize):** Larger engines positively influence car prices. Cars with bigger engine sizes show positive SHAP values, meaning that this feature increases the predicted price.

SHAP Summary Plot

- The SHAP summary plot illustrates the contributions of the most important features in the model. The top features that contribute to the car price predictions include transmission type, fuel type, car age, and mileage.

The blue-to-red gradient indicates low to high feature values. For example, lower values of car age (indicating newer cars) and higher values of transmission and fuel type tend to push predictions higher. In contrast, higher mileage generally lowers the predicted price.

5

Conclusions

The **Gradient Boosting** model outperformed all other models, achieving the highest R-squared value of 0.94, indicating that it explained 94% of the variance in car prices. The low RMSE of 1841.76 also reflects its superior predictive accuracy.