**Assignment 2**

1. Handling Missing Values: Imputation

To address the issue encountered while handling the 'Mileage' column in question 'a', where the presence of units like 'kmpl' resulted in non-numeric values, a multi-step approach was adopted. Initially, question 'b' was addressed, which aimed to remove units from specific attributes such as 'Mileage', 'Engine', 'Power', and 'New\_Price'. This preprocessing step ensured that only numerical values remained in these columns, facilitating easier data manipulation. Subsequently, in question 'a', the focus shifted to handling missing values across all columns. Techniques such as imputation (replacing missing values with mean, median, or mode) or dropping missing values were employed, depending on the nature of the data and the impact on analysis. By first addressing question 'b' to preprocess the data and then proceeding with question 'a', a systematic approach was adopted to overcome challenges related to missing values and non-numeric entries, ensuring data integrity and facilitating subsequent analysis.

data\_set["Mileage"].fillna(data\_set["Mileage"].mean(), inplace=True)

data\_set["Engine"].fillna(data\_set["Engine"].mean(), inplace=True)

data\_set["Power"].fillna(data\_set["Power"].mean(), inplace=True)

data\_set["Seats"].fillna(data\_set["Seats"].mode()[0], inplace=True)

data\_set["New\_Price"].fillna(0, inplace=True)

data\_set.dropna(inplace=True)

**Justification:**

1. `dataset["Mileage"].fillna(dataset["Mileage"].mean(), inplace=True)`: This line fills missing values in the "Mileage" column with the mean value of the "Mileage" column. It ensures that any missing mileage values are replaced with a value that represents the average mileage across the dataset, providing a reasonable estimate for the missing data.

2. `dataset["Engine"].fillna(dataset["Engine"].mean(), inplace=True)`: This line fills missing values in the "Engine" column with the mean value of the "Engine" column. It replaces missing engine capacity values with the average engine capacity observed in the dataset, offering a sensible estimate for the missing data.

3. `dataset["Power"].fillna(dataset["Power"].mean(), inplace=True)`: This line fills missing values in the "Power" column with the mean value of the "Power" column. It ensures that any missing power ratings are replaced with the average power output across the dataset, providing a reasonable estimate for the missing data.

4. `dataset["Seats"].fillna(dataset["Seats"].mode()[0], inplace=True)`: This line fills missing values in the "Seats" column with the mode (most frequent value) of the "Seats" column. It replaces missing seating capacity values with the most common value observed in the dataset, offering a practical choice for the missing data.

5. `dataset["New\_Price"].fillna(0, inplace=True)`: This line fills missing values in the "New\_Price" column with 0. It treats missing new price values as a separate category and ensures that they do not affect calculations or analyses involving price data. However, it's essential to verify the validity of this approach in the context of the dataset and the specific analysis being performed.

Missing values in the columns "Mileage," "Engine," "Power," "Seats," and "New\_Price" are imputed in this code. It is standard practice for numerical data to substitute missing values for the means of the corresponding column (e.g., "Mileage," "Engine," and "Power") in the numerical columns. We use the mode, or most frequent value, for the "Seats" column to fill in the blanks. In the case of "New\_Price," it is acceptable to presume that missing values indicate a new price of 0. Dropped are rows that have missing values in other columns.

A graph with a bar and a number of columns

Description automatically generated with medium confidence

1. Removing Units and Keeping Numerical Values

data\_set['Mileage'] = data\_set['Mileage'].str.replace(' km/kg', '').str.replace(' kmpl', '').astype(float)

data\_set['Engine'] = data\_set['Engine'].str.replace(' CC', '').astype(int)

data\_set['Power'] = data\_set['Power'].str.replace('null', '0.0').str.replace(' bhp', '').astype(float)

data\_set['New\_Price'] = data\_set['New\_Price'].str.replace(' Lakh', '').astype(float)

"I have successfully removed units from attributes such as 'Mileage', 'Engine', 'Power', and 'New\_Price' in the dataset, converting them into numerical values. This preprocessing step ensures that the dataset contains only numeric data in relevant columns, facilitating subsequent analyses and computations. By utilizing pandas' `str.replace()` and `astype()` functions, I efficiently transformed string representations of numerical values into actual numerical values. This achievement improves the dataset's usability for mathematical operations and statistical analyses, enhancing overall data quality and analysis outcomes."

1. Categorical Variable Transformation: One-Hot Encoding for "Fuel\_Type" and "Transmission"

data\_set = pd.get\_dummies(data\_set, columns=['Fuel\_Type', 'Transmission'], drop\_first=True)

The task of converting categorical variables, specifically "Fuel\_Type" and "Transmission," into numerical one-hot encoded values was successfully achieved through the utilization of the `pd.get\_dummies()` function in pandas. By specifying the target columns and setting `drop\_first=True`, redundant dummy variables were created, effectively transforming the categorical data into numerical format suitable for machine learning algorithms. This preprocessing step enhances the dataset's compatibility with various analytical techniques, enabling more robust and accurate predictive modeling and analysis.

A graph with a red and blue bar

Description automatically generated

d. Adding new features  
  
Feature 1: Kilometers Per Year

data\_set['Kilometers\_Driven\_Per\_Year'] = data\_set['Kilometers\_Driven'] / (2024 - data\_set['Year'])

data\_set.head()

In this code, By dividing the total kilometers driven by the number of years since the car's manufacturing year (2024 - Year), we can determine the average kilometers driven annually for each automobile by creating a new column called "Kilometers\_Per\_Year" in this code.With the help of this new tool, you can now find out how many kilometers each car is used for each year.

Feature 2:Current Age

import datetime

current\_year = datetime.datetime.now().year

data\_set['Current\_Age'] = current\_year - data\_set['Year']

data\_set.head()

Next in this code, we make a new column called "Current\_Age" and use the current year minus the "Year" column to determine each car's age.This will update your dataset with a "Current\_Age" column that shows each car's current age.

e. Data Manipulation Operations on the Dataset

selected\_columns = data\_set[['Name', 'Location', 'Year', 'Kilometers\_Driven', 'Owner\_Type', 'Mileage', 'Engine', 'Power', 'Seats', 'New\_Price', 'Price', 'Fuel\_Type\_Electric', 'Fuel\_Type\_Petrol', 'Transmission\_Manual', 'Kilometers\_Driven\_Per\_Year', 'Current\_Age']]

filtered\_data = selected\_columns[selected\_columns['Year'] >= 2018]

renamed\_columns = filtered\_data.rename(columns={'Kilometers\_Driven': 'KMs\_Driven', 'Owner\_Type': 'Owner'})

renamed\_columns['Price\_per\_KM'] = renamed\_columns['Price'] / renamed\_columns['KMs\_Driven']

sorted\_data = renamed\_columns.sort\_values(by='New\_Price', ascending=False)

summary = sorted\_data.groupby('Name').agg({'New\_Price': 'mean', 'Mileage': 'mean', 'Engine': 'mean', 'Power': 'mean', 'Seats': 'mean'})

summary

In this analysis, I successfully executed a sequence of data manipulation operations on the dataset. Beginning with the selection of pertinent columns such as 'Name', 'Location', 'Year', and others, I systematically applied filtering to extract records corresponding to the years 2018 and beyond. Renaming operations enhanced the dataset's clarity by making column names more consistent, exemplified by the transformation of 'Kilometers\_Driven' to 'KMs\_Driven' and 'Owner\_Type' to 'Owner'. The introduction of a new column, 'Price\_per\_KM', through mutation, added valuable insights by calculating the price per kilometer driven. Sorting the dataset by 'New\_Price' in descending order further organized the data. Finally, summarization with group by operations allowed the aggregation of mean values for key attributes like 'New\_Price', 'Mileage', 'Engine', 'Power', and 'Seats' based on the 'Name' column. This comprehensive data manipulation approach not only refines the dataset for subsequent analyses but also provides a nuanced understanding of average attributes across different car models.