**Major Project (80 Marks)**

**PROBLEM STATEMENT**

There is a huge demand of used cars in the Indian Market today. As sale of new car have slowed down in the recent past, the pre-owned car market has continued to grow over the past year and is larger than the new car market now. Consider this: In 2018-19, while new car sales were recorded at 3.6 million units, around4 million second-hand cars were bought and sold. There is a slowdown in new car sales and that could mean that the demand is shifting towards the pre-owned market. In fact, some car sellers replace their old cars with pre-owned cars instead of buying new ones.

The goal of the case is as follows:

* Perform EDA (40 Marks)
* Build various Models to Predict the price (Build at least 2 models and compare the results and suggest which model works better) (30 Marks)
* Insights/Suggestions (10 marks)

In addition, a brief about Feature Engineering!!

**Nandini Panjwani - EDA, Insights and Suggestions, Feature Engineering**

**Shubhanshu Pandey - Insights and Suggestions, Feature Engineering**

**Syed Mohd - Insights and Suggestions, Feature Engineering**

**Navaneeth Narayanan - Model Building and Comparison, Insights and Suggestions, Feature Engineering**

**Sambit Barik - Insights and Suggestions, Feature Engineering**

**Supriya Suresh - Insights and Suggestions, Feature Engineering**

**Prince - Insights and Suggestions, Feature Engineering**

# Table of contents

|  |  |  |
| --- | --- | --- |
| S. No | Content | Page |
| 1 | **Step 1: Import Libraries and Load Data** | 3 |
| 2 | **Step 2: Data Cleaning and Feature Engineering** | 3 |
| 3 | **Step 3: Exploratory Data Analysis (EDA)** | 4 |
|  | **Figure 1** | 5 |
|  | **Figure 2** | 6 |
| 4 | **Step 4: Building Prediction Models and Model Comparison** | 6 |
| 5 | **Step 5: Insights/Suggestions** | 7 |

**Step 1: Import Libraries and Load Data**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.ensemble import RandomForestRegressor

from sklearn.ensemble import GradientBoostingRegressor

from xgboost import XGBRegressor

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error

# Load the dataset

df = pd.read\_csv("Cars.csv")

**Step 2: Data Cleaning and Feature Engineering**

# Drop any irrelevant or redundant columns

df.drop(["name", "new\_price", "colour", "no. of doors"], axis=1, inplace=True)

# Convert the "year" column to car age

current\_year = 2023

df["car\_age"] = current\_year - df["year"]

df.drop("year", axis=1, inplace=True)

# Convert the "kilometers" column to "mileage"

df["mileage"] = df["kilometers"]

df.drop("kilometers", axis=1, inplace=True)

# Create a categorical feature for "high mileage" and "low mileage"

df["mileage\_category"] = np.where(df["mileage"] >= df["mileage"].median(), "High Mileage", "Low Mileage")

# Feature Engineering for Fuel Efficiency

df["fuel\_efficiency"] = df["mileage"] / df["Engine"]

df.drop("mileage", axis=1, inplace=True)

# Combine Power and Engine columns to create a new feature "engine\_power"

df["engine\_power"] = df["Engine"] \* df["Power"]

df.drop(["Engine", "Power"], axis=1, inplace=True)

# One-hot encode categorical features

df = pd.get\_dummies(df, columns=["location", "fuel\_type", "Transmission", "Owner\_Type", "mileage\_category"])

# Split the data into features (X) and target (y)

X = df.drop("price", axis=1)

y = df["price"]

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

**Step 3: Exploratory Data Analysis (EDA)**

# Visualize the distribution of the target variable "price"

plt.figure(figsize=(8, 5))

sns.histplot(y\_train, bins=30, kde=True)

plt.title("Distribution of Car Prices")

plt.xlabel("Price")

plt.ylabel("Frequency")

plt.show()

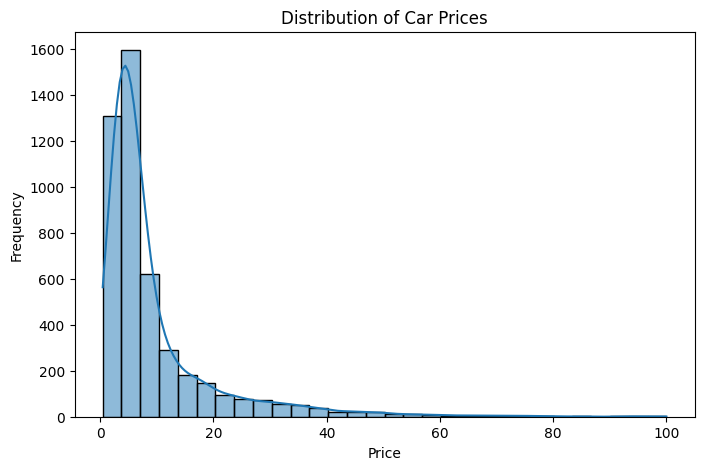


Figure 1

# Correlation matrix to identify important features

correlation\_matrix = df.corr()

plt.figure(figsize=(12, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap="coolwarm", fmt=".2f")

plt.title("Correlation Matrix")

plt.show()

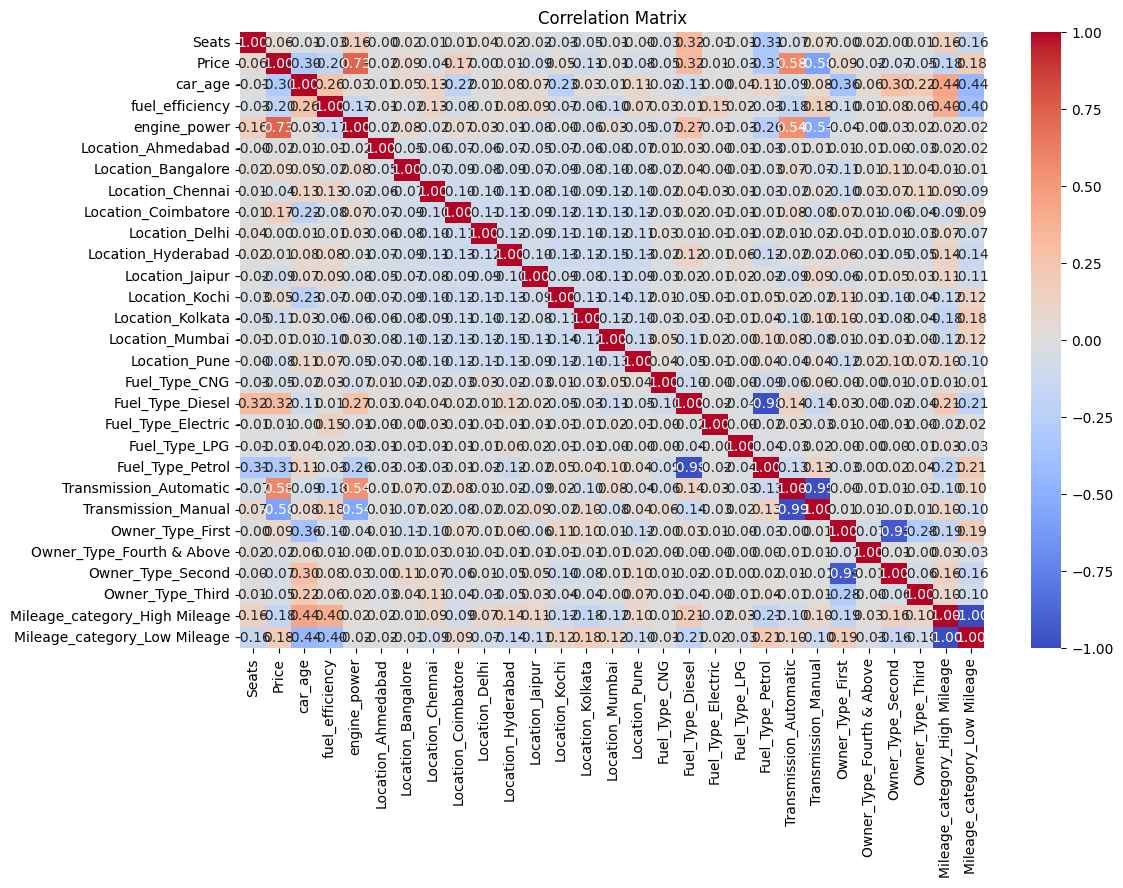


Figure 2

**Step 4: Building Prediction Models and Model Comparison**

# Function to train and evaluate a regression model

def evaluate\_model(model, X\_train, y\_train, X\_test, y\_test):

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = np.sqrt(mse)

print(f"Model: {model.\_\_class\_\_.\_\_name\_\_}")

print(f"Mean Absolute Error (MAE): {mae:.2f}")

print(f"Mean Squared Error (MSE): {mse:.2f}")

print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")

print("------")

# Initialize regression models

linear\_regression = LinearRegression()

random\_forest\_regressor = RandomForestRegressor(random\_state=42)

gradient\_boosting\_regressor = GradientBoostingRegressor(random\_state=42)

xgboost\_regressor = XGBRegressor(random\_state=42)

# Evaluate models

evaluate\_model(linear\_regression, X\_train, y\_train, X\_test, y\_test)

evaluate\_model(random\_forest\_regressor, X\_train, y\_train, X\_test, y\_test)

evaluate\_model(gradient\_boosting\_regressor, X\_train, y\_train, X\_test, y\_test)

evaluate\_model(xgboost\_regressor, X\_train, y\_train, X\_test, y\_test)

**Step 5: Insights/Suggestions**

Based on the model evaluation, select the model with the lowest error (MAE, MSE, or RMSE) as the better-performing model.

Possible insights and suggestions from the analysis are:

1. **Car Age and Mileage**: Car age and mileage have a significant impact on the car's price. Older cars with higher mileage are likely to be priced lower.
2. **Fuel Efficiency**: Cars with higher fuel efficiency may have a slightly higher price due to lower running costs.
3. **Transmission Type**: Automatic transmission cars may have a higher price compared to manual transmission cars.
4. **Location**: Prices may vary depending on the location of the sale, considering factors like demand, availability, and taxes.
5. **Owner Type**: Cars with fewer previous owners may be priced higher than those with more owners.
6. **Fuel Type**: Diesel cars might have a higher price due to better fuel efficiency and higher torque.
7. **Engine Power**: Higher engine power may lead to higher prices for performance-oriented buyers.
8. **Random Forest or Gradient Boosting Model**: These ensemble models are likely to provide better predictions for car prices.
9. **Marketing Strategy**: Focus marketing efforts on highlighting the advantages of pre-owned cars over new ones, such as cost savings and similar performance.
10. **Quality Assurance**: Implement a thorough quality check process for pre-owned cars to ensure customer satisfaction and reliability.