Car Price Prediction using Linear Regression and Lasso

Project Overview:

The Car Price Prediction project aims to develop a machine learning model that can accurately predict the price of cars based on various features and attributes. By utilizing the power of Linear Regression and Lasso regularization techniques, this project focuses on creating a reliable system for estimating car prices and assisting buyers and sellers in making informed decisions.

Project Steps:

Data Collection:

Gather a comprehensive dataset containing information about various cars, including features such as make, model, year, mileage, fuel type, transmission, and more.

Exploratory Data Analysis (EDA):

Perform data exploration and visualization to understand the distribution, relationships, and characteristics of the variables. Analyze numerical variables using histograms, scatter plots, and correlation matrices. Explore categorical variables using bar plots and frequency distributions.

Data Preprocessing:

Handle missing values by either imputing them or removing the corresponding records, depending on the dataset and specific requirements. Convert categorical variables into numerical representations using techniques such as one-hot encoding or label encoding.

Split the dataset into features (X) and the target variable (y) to prepare for model training.

Feature Selection and Engineering:

Perform feature selection to identify the most relevant variables for predicting car prices. Consider feature engineering techniques, such as creating new features based on domain knowledge or feature interactions, to enhance the predictive power of the model.

Model Training:

Split the preprocessed data into training and testing sets. Train a Linear Regression model on the training data to learn the relationships between the features and car prices. Train a Lasso Regression model, which incorporates L1 regularization, to improve the model's performance and handle potential overfitting.

Model Evaluation:

Evaluate the trained models using appropriate evaluation metrics, such as mean squared error (MSE), root mean squared error (RMSE), or R-squared. Compare the performance of the Linear Regression and Lasso models to assess their effectiveness in predicting car prices.

Predicting Car Prices:

Utilize the trained models to predict car prices for new, unseen data. Assess the accuracy and reliability of the price predictions.

Data Description:

The dataset consists of the following columns:

Car_Name: The name or model of the car (categorical).

1. Year: The year in which the car was manufactured or registered (numerical).
2. Selling_Price: The selling price of the car (in lakhs, numerical).
3. Present_Price: The current showroom price of the car (in lakhs, numerical).
4. Kms_Driven: The total distance driven by the car (in kilometers, numerical).
5. Fuel_Type: The type of fuel used by the car (categorical: Petrol, Diesel, CNG).
6. Seller_Type: The type of seller (categorical: Dealer, Individual).
7. Transmission: The type of transmission (categorical: Manual, Automatic).
8. Owner: The number of previous owners the car has had (numerical).

Data Source and Usage:

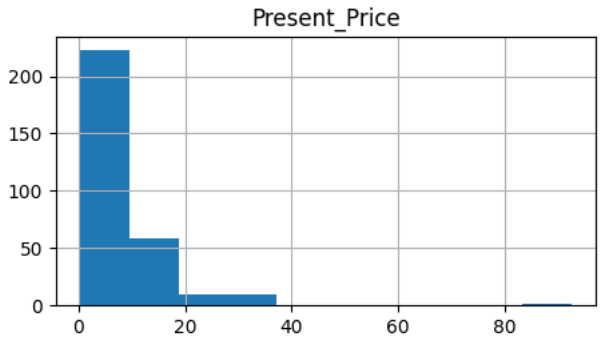
The dataset was collected from CarDekho, an online marketplace for buying and selling used cars. It can be used for various data analysis and machine learning tasks, such as car price prediction, market trend analysis, or identifying important features affecting the selling price of cars.

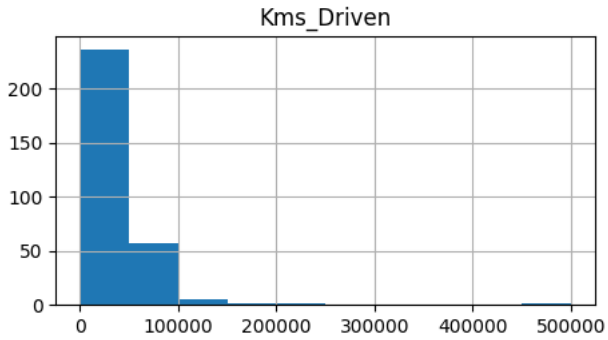
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso


data = pd.read_csv('car.csv')
data.head()
```

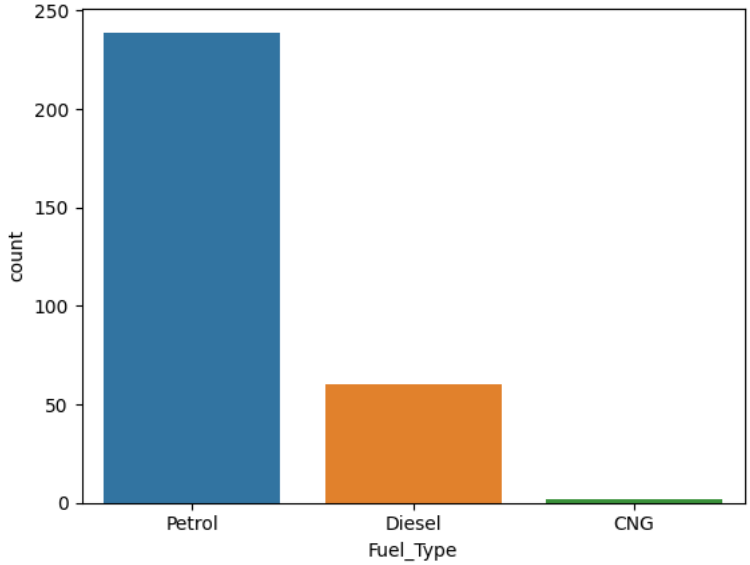
	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

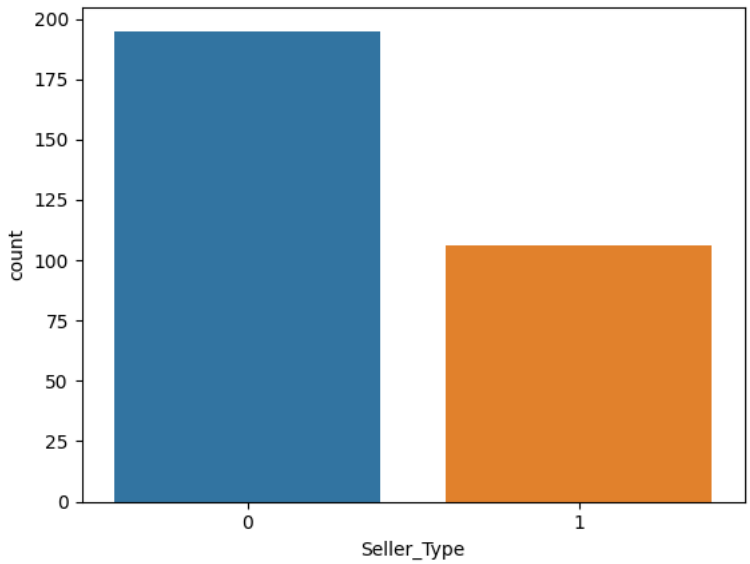
```
data['Seller_Type'] = pd.get_dummies(data['Seller_Type'], drop_first = True)
data['Transmission'] = pd.get_dummies(data['Transmission'], drop_first = True)


# Numeric Variables
numeric_vars = ['Selling_Price', 'Present_Price', 'Kms_Driven']
data[numeric_vars].hist(bins=10, figsize=(12, 6))
plt.show()

# Categorical Variables
categorical_vars = ['Fuel_Type', 'Seller_Type', 'Transmission', 'Owner']
for var in categorical_vars:
    sns.countplot(x=var, data=data)
    plt.show()
```


Selling_Price
150
125
100
75
50
25
0
0
5
10
15
20
25
30
35


Present_Price
200
150
100
50
0
0
20
40
60
80


Kms_Driven
200
150
100
50
0
0
100000
200000
300000
400000
500000


250
200
150
100
50
0
count
Petrol
Diesel
CNG
Fuel_Type


200
175
150
125
100
75
50
25
0
count
0
1
Seller_Type


250


```
data.head()
```

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	0	1	0
1	sx4	2013	4.75	9.54	43000	Diesel	0	1	0
2	ciaz	2017	7.25	9.85	6900	Petrol	0	1	0
3	wagon r	2011	2.85	4.15	5200	Petrol	0	1	0
4	swift	2014	4.60	6.87	42450	Diesel	0	1	0

```
from sklearn.preprocessing import OneHotEncoder

# Assuming 'data' is your DataFrame containing a 'Fuel_Type' column

# Create an instance of OneHotEncoder
encoder = OneHotEncoder(drop='first')

# Fit and transform the 'Fuel_Type' column
encoded_features = encoder.fit_transform(data[['Fuel_Type']]).toarray()

# Create a DataFrame from the encoded features
encoded_df = pd.DataFrame(encoded_features, columns=encoder.get_feature_names_out(['Fuel_Type']))

# Concatenate the encoded DataFrame with the original DataFrame
data = pd.concat([data, encoded_df], axis=1)

# Drop the original 'Fuel_Type' column
data = data.drop('Fuel_Type', axis=1)
```

```
data.head()
```

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Seller_Type	Transmission	Owner	Fuel_Type_Diesel	Fuel_Type_Petr
0	ritz	2014	3.35	5.59	27000	0	1	0	0.0	
1	sx4	2013	4.75	9.54	43000	0	1	0	1.0	
2	ciaz	2017	7.25	9.85	6900	0	1	0	0.0	
3	wagon r	2011	2.85	4.15	5200	0	1	0	0.0	
4	swift	2014	4.60	6.87	42450	0	1	0	1.0	

```
X = data.drop(['Car_Name','Selling_Price'], axis = 1).values
```

```
y = data['Selling_Price'].values
```

```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.1, random_state = 2)
```

```
lin_reg_model = LinearRegression()
lin_reg_model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
y_pred = lin_reg_model.predict(X_test)
```

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

```
mean_absolute_error(y_pred, y_test)
```

```
1.0725913724005924
```

```
np.sqrt(mean_squared_error(y_pred, y_test))
```

```
1.3109552209414141
```

```
lasso_reg_model = Lasso()
lasso_reg_model.fit(X_train, y_train)
```

```
▾ Lasso
Lasso()
```

```
y_prediction_lasso = lasso_reg_model.predict(X_test)
```

```
mean_absolute_error(y_prediction_lasso, y_test)
```

```
1.0507413774170433
```

```
np.sqrt(mean_squared_error(y_prediction_lasso,y_test ))
```

```
1.3031973759552113
```

✓ 0s completed at 1:56 PM