INTERNSHIP PROJECT REPORT

Submitted to

INLIGHNX GLOBAL PVT. LTD.

Bangalore, India

Date of Submission

15/01/26

Price Predicting model

Car Price Prediction Model

Model: Linear Regression, Lasso Regression

Prepared by

Nadar Deepthi Chenthil

AI & ML Intern

ITID5802

Internship Duration

15/11/25 – 15/01/26

Index

Link to GitHub with the .ipynb file containing the implementation code, all dataset .csv files and documentation:

https://github.com/Deepthi-Nadar/Project-using-ml/tree/main

1. Introduction

With the rapid growth of the automobile market, determining the correct resale price of used cars has become a challenging task. The price of a car depends on multiple factors such as brand value, age, fuel type, kilometers driven, and transmission type. Manual estimation often leads to inaccurate pricing.
This project aims to solve this problem using **Machine Learning techniques** to predict the **selling price of used cars** based on historical data. The system helps sellers and buyers make informed pricing decisions.

2. Problem Statement

To design and implement a machine learning model that accurately predicts the **selling price of a used car** using various car-related features. The model should minimize prediction error and provide reliable price estimates.

3. Objectives of the Project

- To analyze the factors affecting car prices
- To preprocess and clean the dataset for better accuracy
- To apply **Linear Regression** and **Lasso Regression** models
- To compare predicted prices with actual prices
- To evaluate model performance using suitable metrics

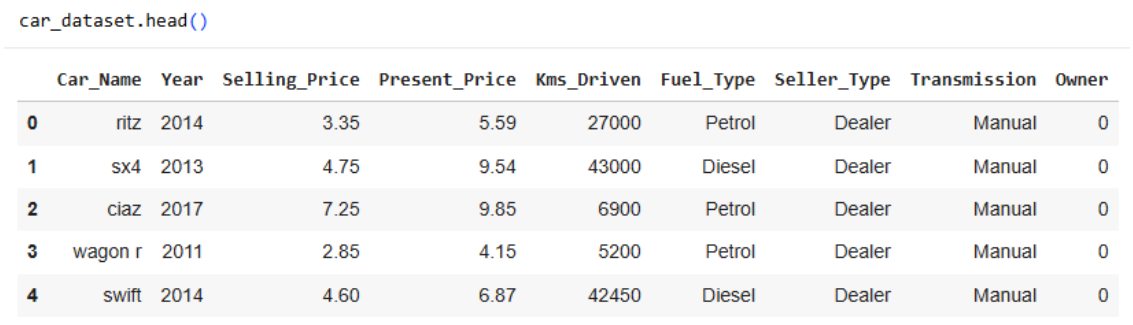
4. Dataset Description

The dataset `car data.csv` contains information about used cars and their selling prices.

Dataset Features

Feature	Description
Car_Name	Name/model of the car
Year	Year of manufacture
Selling_Price	Price at which the car was sold (Target variable)
Present_Price	Original showroom price
Kms_Driven	Distance driven by the car

Feature	Description
Fuel_Type	Type of fuel used
Seller_Type	Dealer or Individual
Transmission	Manual or Automatic
Owner	Number of previous owners

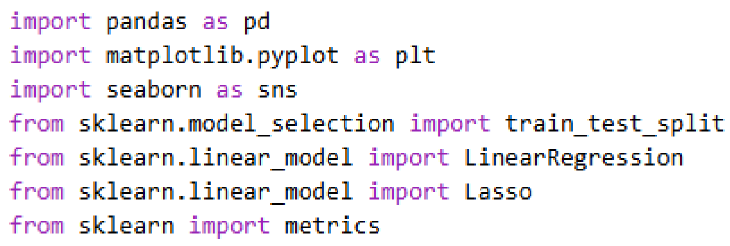


```
car_dataset.head()
```

	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

5. Tools and Technologies Used

- **Language:** Python
- **Libraries:** Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn
- **IDE:** Jupyter Notebook



```
import pandas as pd
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.linear_model import Lasso
from sklearn import metrics
```

6. System Methodology

6.1 Data Collection

The dataset was collected from a public source containing real-world used car information.

6.2 Data Preprocessing

- Checked for missing and null values

```
car_dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Car_Name       301 non-null    object 
 1   Year           301 non-null    int64  
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Kms_Driven     301 non-null    int64  
 5   Fuel_Type      301 non-null    object 
 6   Seller_Type    301 non-null    object 
 7   Transmission   301 non-null    object 
 8   Owner          301 non-null    int64  
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

```
car_dataset.isnull().sum()
```

	0
Car_Name	0
Year	0
Selling_Price	0
Present_Price	0
Kms_Driven	0
Fuel_Type	0
Seller_Type	0
Transmission	0
Owner	0

dtype: int64

- Converted categorical features into numerical values using **Label Encoding**

```
# encoding (Fuel_Type) Column
car_dataset.replace({'Fuel_Type':{'Petrol':0,'Diesel':1,'CNG':2}},inplace=True)

# encoding (Seller_Type) Column
car_dataset.replace({'Seller_Type':{'Dealer':0,'Individual':1}},inplace=True)

# encoding (Transmission) Column
car_dataset.replace({'Transmission':{'Manual':0,'Automatic':1}},inplace=True)
```

```
car_dataset.head()
```

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

- Selected relevant features for model training

```
X = car_dataset.drop(['Car_Name','Selling_Price'],axis=1)
Y = car_dataset['Selling_Price']
```

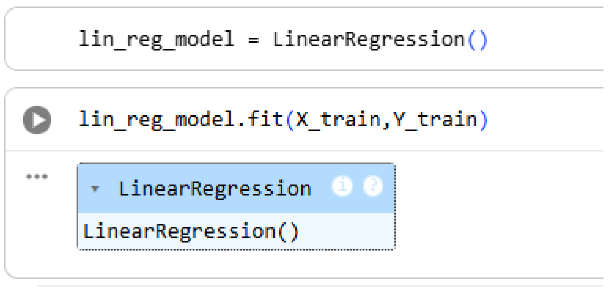
6.3 Feature Engineering

- Extracted meaningful features such as car age from the year
- Transformed categorical data for better model compatibility

7. Model Implementation

7.1 Linear Regression

Linear Regression models the relationship between independent variables and the selling price. It is simple and effective for baseline predictions.



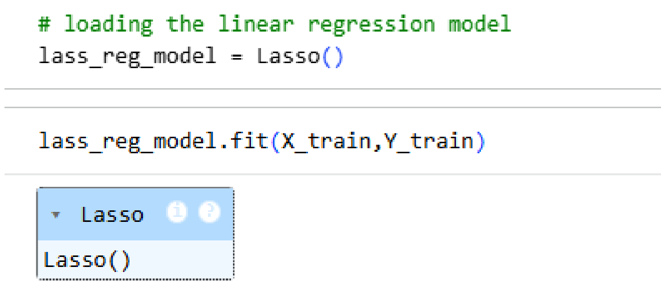
```
lin_reg_model = LinearRegression()
```

```
lin_reg_model.fit(X_train,Y_train)
```

```
LinearRegression
LinearRegression()
```

7.2 Lasso Regression

Lasso Regression adds regularization to Linear Regression, helping reduce overfitting and improve prediction stability by penalizing large coefficients.



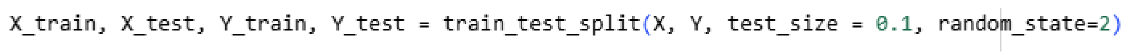
```
# loading the linear regression model
lass_reg_model = Lasso()
```

```
lass_reg_model.fit(X_train,Y_train)
```

```
Lasso
Lasso()
```

8. Model Training and Testing

- The dataset was split into **training (90%)** and **testing (10%)** sets
- Models were trained using the training dataset
- Predictions were made on the testing dataset



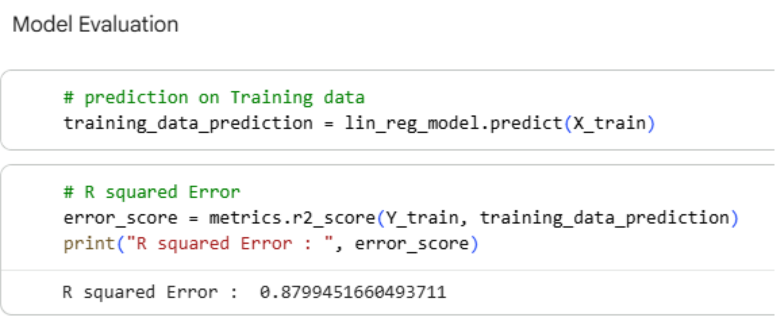
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.1, random_state=2)
```

9. Performance Evaluation

The models were evaluated using:

- **Mean Absolute Error (MAE)**
- **Mean Squared Error (MSE)**
- **R² Score**

1. Linear Regression

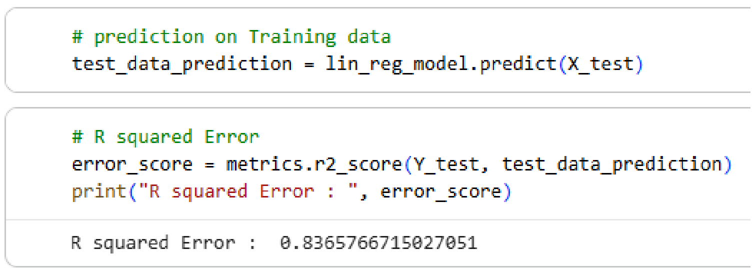


Model Evaluation

```
# prediction on Training data
training_data_prediction = lin_reg_model.predict(X_train)
```

```
# R squared Error
error_score = metrics.r2_score(Y_train, training_data_prediction)
print("R squared Error : ", error_score)
```

```
R squared Error :  0.8799451660493711
```

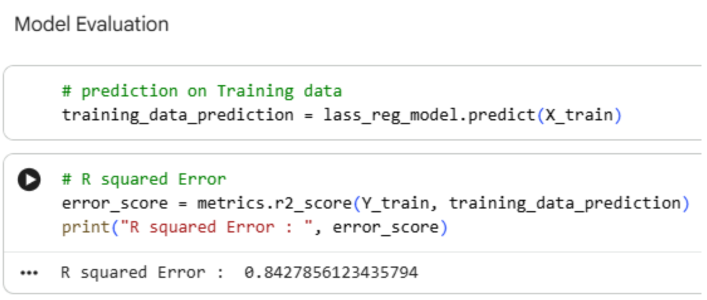


```
# prediction on Training data
test_data_prediction = lin_reg_model.predict(X_test)
```

```
# R squared Error
error_score = metrics.r2_score(Y_test, test_data_prediction)
print("R squared Error : ", error_score)
```

```
R squared Error :  0.8365766715027051
```

2. Lasso Regression



Model Evaluation

```
# prediction on Training data
training_data_prediction = lass_reg_model.predict(X_train)
```

```
# R squared Error
error_score = metrics.r2_score(Y_train, training_data_prediction)
print("R squared Error : ", error_score)
```

```
R squared Error :  0.8427856123435794
```

```
# prediction on Training data
test_data_prediction = lass_reg_model.predict(X_test)
```

```
# R squared Error
error_score = metrics.r2_score(Y_test, test_data_prediction)
print("R squared Error : ", error_score)
```

```
R squared Error :  0.8709167941173195
```

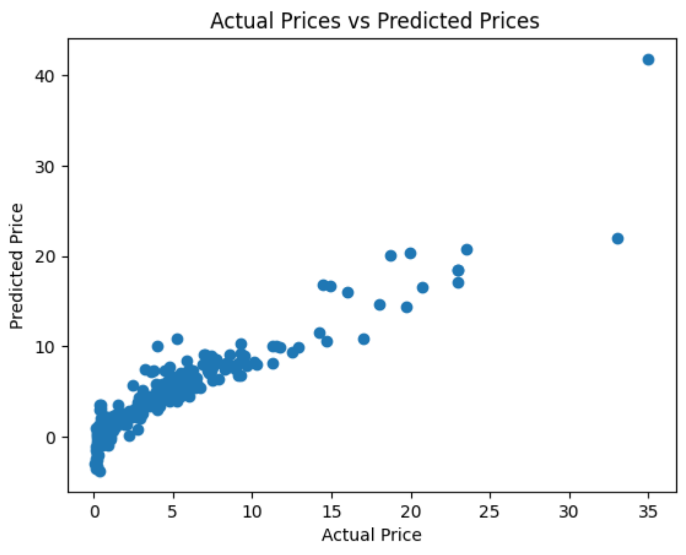
These metrics help measure prediction accuracy and error rate.

10. Results and Discussion

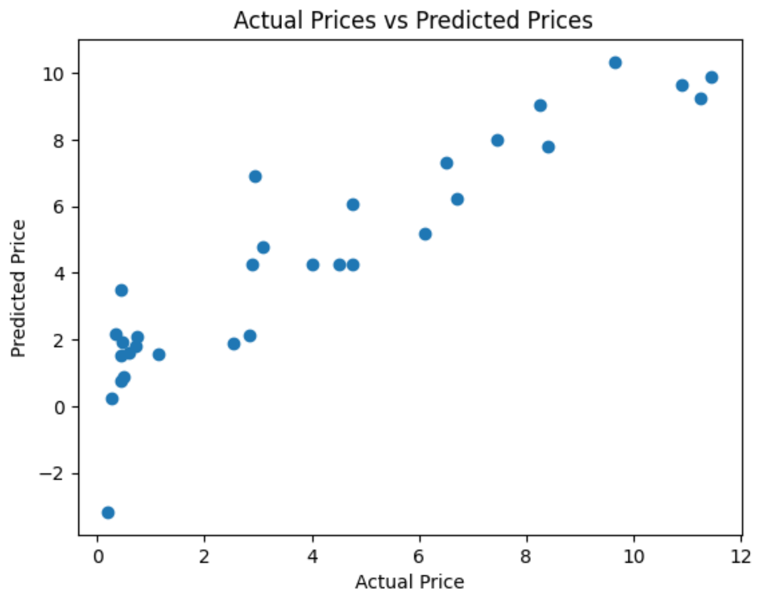
- Linear Regression provided acceptable prediction results
- Lasso Regression reduced overfitting and improved model generalization
- Predicted prices were close to actual prices for most samples

Visualize the actual prices and Predicted prices

```
plt.scatter(Y_train, training_data_prediction)
plt.xlabel("Actual Price")
plt.ylabel("Predicted Price")
plt.title(" Actual Prices vs Predicted Prices")
plt.show()
```

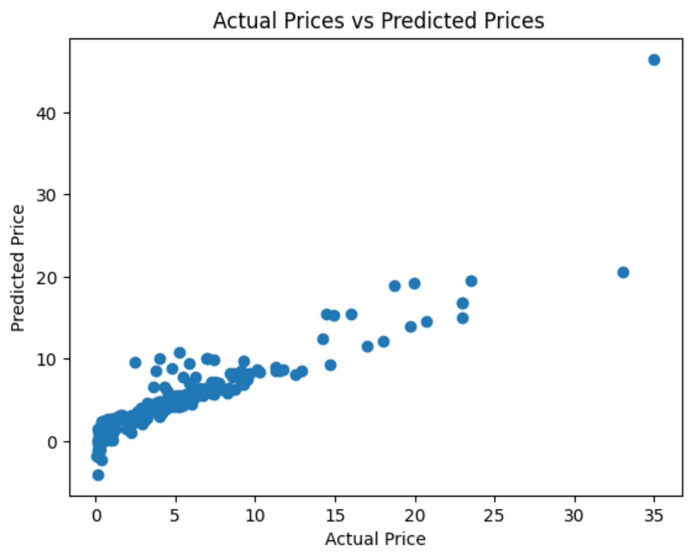


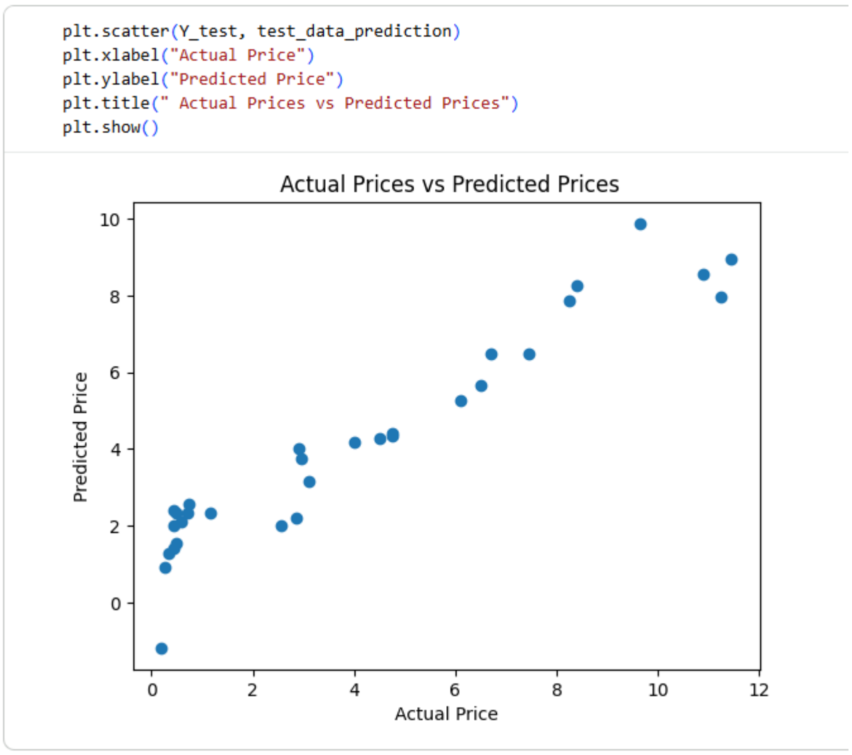
```
plt.scatter(Y_test, test_data_prediction)
plt.xlabel("Actual Price")
plt.ylabel("Predicted Price")
plt.title(" Actual Prices vs Predicted Prices")
plt.show()
```



Visualize the actual prices and Predicted prices

```
plt.scatter(Y_train, training_data_prediction)
plt.xlabel("Actual Price")
plt.ylabel("Predicted Price")
plt.title(" Actual Prices vs Predicted Prices")
plt.show()
```





```
plt.scatter(Y_test, test_data_prediction)
plt.xlabel("Actual Price")
plt.ylabel("Predicted Price")
plt.title(" Actual Prices vs Predicted Prices")
plt.show()
```

Graphical comparisons between **actual vs predicted prices** show the effectiveness of the models.

11. Conclusion

This project demonstrates that machine learning can effectively predict used car prices using regression techniques. Both Linear and Lasso Regression models performed well, with Lasso offering better control over overfitting. The system can be useful for real-world car resale platforms.

12. Future Scope

- Apply advanced algorithms like **Random Forest** or **XGBoost**
- Increase dataset size for better accuracy
- Deploy the model as a web application
- Add more features like car condition and service history