CAR PRICE PREDICTION

Using Machine Learning in Python

A MINOR-II PROJECT REPORT

Submitted by

ALI AHMAD ABDULLAH 2020-310-019

In partial fulfillment for the award of the degree of

B. TECH COMPUTER SCIENCE & ENGINEERING

Under the supervision of

Dr. PARUL AGGARWAL

Department of Computer Science & Engineering

School of Engineering Sciences & Technology

JAMIA HAMDARD

New Delhi-
110062

(2023)
DECLARATION

I, **Mr. Ali Ahmad Abdullah** a student of **Bachelor of Technology Computer Science & Engineering (B. Tech CSE), Enrolment No: 2020-310-019** hereby declare that the Project/Dissertation entitled **"Car Price Prediction using Machine Learning in Python"** which is being submitted by me to the Department of Computer Science, Jamia Hamdard, New Delhi in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology Computer Science & Engineering (B. Tech CSE),** is my original work and has not been submitted anywhere else for the award of any Degree,

Diploma, Associateship, Fellowship or other similar title or recognition.

Date: 17 NOVEMBER 2023 **ALI AHMAD ABDULLAH**

Place: New Delhi **2020-310-019**

ACKNOWLEDGEMENT

I express my sincere thanks to **Dr Parul Aggarwal** (Dept. of Computer Science and Engineering), my project in charge, who guided me through the project for her valuable suggestions and guidance for completing the project. This project has been a success only because of her guidance.

I deeply express my sincere thanks to our Head of Department **Dr. Farheen Siddiqui** for encouraging me and allowing me to present the project on the topic "**Car Price Prediction using Machine Learning in Python**" at the department premises for the partial fulfilment of the requirements leading to the award of Bachelor of Technology degree.

I am also thankful to the whole computer science and engineering department for providing the technical support to carry out the project work, letting us utilize all the necessary facilities of the institute and providing guidance at each & every step during the project work.

INDEX

OBJECTIVE

The objective of this machine learning project is to develop a predictive model that accurately predicts the price of a car using a dataset containing various car features. The model should be able to analyze various features such as to accurately predict the price of a car.

These features are: -

- **YEAR**
- **PRESENT PRICE**
- **KILOMETERS DRIVEN**
- **FUEL TYPE**
- **SELLER TYPE**
- **TRANSMISSION**
- **OWNER**
- **ENGINE(CC)**
- **SEATS**

The project aims to leverage machine learning algorithms to improve car price prediction using data of approximately 4320 cars.
This could lead to better prediction of car prices.

In order to get the best potential accuracy and interpretability in the predictions, the project may additionally comprise feature selection, model evaluation, and optimization. The objective is to develop a scalable and trustworthy model that can be applied to automated sectors, such multinational automakers, to help them decide on future vehicle costs.

INTRODUCTION
CAR PRICE

Car prices are influenced by a multitude of factors, ranging from manufacturing costs and technological advancements to market demand and economic conditions. Automakers carefully assess production expenses, including materials, labour, and research and development, to determine the baseline cost of a vehicle. Additionally, innovations in automotive technology, safety features, and fuel efficiency can impact the overall pricing strategy.

Market dynamics, consumer preferences, and competitive landscapes also play a pivotal role in setting car prices. In an ever-evolving industry, **understanding** and **forecasting** these elements is crucial for automakers to make strategic decisions that align with market trends and consumer expectations.

In the automobile sector, accurate and timely price forecasting can be essential for efficient problem-solving and risk mitigation. Since machine learning algorithms can handle complicated datasets and produce reliable predictions, they have emerged as potential tools for price prediction. Examples of these algorithms are Lasso Regression and Linear Regression.

In the field of artificial intelligence (AI), machine learning entails creating models and algorithms that let computers learn from data and make judgments or predictions without needing to be explicitly programmed. Machine learning algorithms are extensively utilized in many different applications, including image recognition, natural language processing, recommendation systems, fraud detection, and more. They are made to find patterns, correlations, and insights from massive volumes of data.

There are several types of machine learning algorithms, including:

1. **Supervised Learning**: When an algorithm is trained on labeled data—data that provides the proper output or outcome—supervised learning takes place. With this labeled data, the algorithm learns to predict or classify things. Support vector machines (SVM), decision

trees, logistic regression, and linear regression are a few types of supervised learning techniques.

2. **Unsupervised Learning**: When an algorithm is taught on unlabelled data—data that lacks an output or outcome—unsupervised learning takes place. Without any labelled instruction, the program finds patterns, groupings, or correlations within the data. Clustering is one type of unsupervised learning technique.

3. **Reinforcement Learning**: Reinforcement learning involves the algorithm learning by making mistakes and getting feedback in the form of incentives or punishments for its actions in the environment. Over time, the algorithm gains the ability to decide or behave in a way that maximizes the overall cumulative reward. Video games, autonomous systems, and robots all make extensive use of reinforcement learning.

4. **Deep Learning**: A form of machine learning known as "deep learning" uses multiple-layered artificial neural networks to automatically process data and extract features. In several applications, deep learning algorithms—like recurrent neural networks (RNNs) for sequence data and convolution neural networks (CNNs) for image recognition—have produced state-of-the-art results.

5. **Other Algorithms**: Depending on their advantages and disadvantages, a variety of additional machine learning algorithms, including naive Bayes, random forests, k-nearest neighbours (KNN), decision trees, and support vector machines (SVM), are employed for different applications.

These are only a few instances of the many different machine learning algorithms that are employed in the industry. The particular problem, dataset, and task requirements all play a role in choosing the best method. Researchers and practitioners are always creating new methods and strategies to enhance the functionality and performance of machine learning models, and machine learning algorithms are a dynamic field.

Our goal in this project is to use the Lasso Regression and Linear Regression algorithm to create a car price prediction model. We will make use of a dataset that includes the selling price and a variety of attributes specific to each automobile. In order to manage missing values, feature scaling, and categorical variable encoding, the dataset will first undergo pre-processing. Subsequently, we will investigate the data using statistical analysis and visualization to learn more about the features of the dataset.

Next, we will implement the Lasso Regression and Linear Regression algorithms for car price prediction. Linear regression models the relationship between the car's price and a set of features using a linear equation. It aims to find the best-fit line that minimizes the difference between predicted and actual prices. Lasso regression, unlike linear regression, includes a regularization term that penalizes the absolute values of the coefficients. This encourages the model to shrink less important features to zero, effectively performing feature selection.

Utilizing linear and Lasso regression in car price prediction allows for the creation of models that not only predict prices but also provide insights into the most influential factors affecting those prices, aiding decision-making processes in the automotive industry.

LINEAR REGRESSION ALGORITHM

Linear regression is a fundamental statistical technique used for modelling the relationship between a dependent variable and one or more independent variables. The method assumes a linear relationship, meaning that a change in the independent variable(s) is associated with a constant change in the dependent variable. The model takes the form of a linear equation, typically represented as:

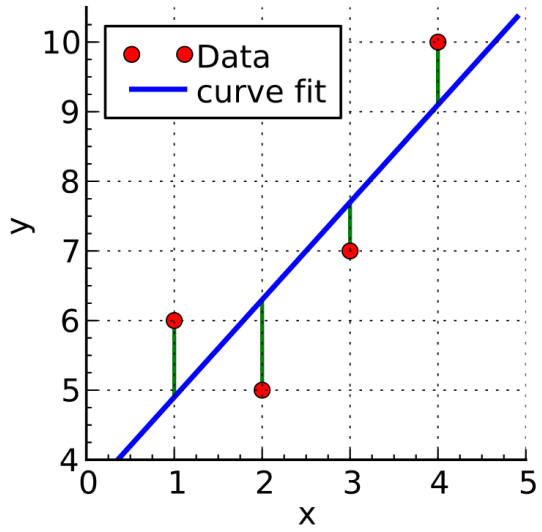
$Y=b0 +b1\ X1 +b2\ X2 +\ldots+bn\ Xn +\varepsilon$

Here:

- Y is the dependent variable.
- b0 is the intercept.
- $b1,b2,\ldots,bn$ are the coefficients of the independent variables $X1,X2,\ldots,Xn$.
- ε presents the error term, capturing unobserved factors affecting Y.

A linear regression graph visually represents the relationship between the independent variable(s) and the dependent variable in a linear regression model. Here are the key components:

- **Scatterplot:** The graph typically starts with a scatterplot, where each point on the plot represents an observation in your dataset. If you have one independent variable, the x-axis corresponds to that variable, and the y-axis represents the dependent variable. Each point represents a pair of values (x, y) from your dataset.
- **Regression Line:** The regression line is a straight line that best fits the data points. It's determined by the coefficients obtained during the model training phase. For simple linear regression (one independent variable), the equation of the line is $Y=b0+b1X+\varepsilon$, where $b0$ is the intercept, $b1$ is the slope, X is the independent variable, and ε is the error term. Below is the representation of a regression line.



[1]

In linear regression, the observations (**red**) are assumed to be the result of random deviations (**green**) from an underlying relationship (**blue**) between a dependent variable (y) and an independent variable (x).

LASSO REGRESSION ALGORITHM

Lasso regression, or Least Absolute Shrinkage and Selection Operator, is a linear regression technique that incorporates a regularization term to improve the model's predictive performance and prevent overfitting. Like linear regression, Lasso aims to model the relationship between a dependent variable and one or more independent variables. However, Lasso introduces a penalty term that encourages

the model to select a subset of the most important features while shrinking the coefficients of less influential variables towards zero.

The mathematical expression for the Lasso regression model is:

$Y=b0+b1X1+b2X2+...+bnXn+\lambda\sum i=1n|bi|+\varepsilon$

Here:

- Y is the dependent variable.
- $b0$ is the intercept.
- $b1,b2,...,bn$ are the coefficients of the independent variables $X1,X2,...,X$.
- λ is the regularization parameter that controls the strength of the penalty term.
- ∑i=1n |bi | represents the absolute values of the coefficients.

A Lasso regression graph visually represents the relationship between the independent variable(s) and the dependent variable, similar to a regular linear regression plot. Here's a brief explanation:

1. **Scatterplot:** The graph begins with a scatterplot, where each point on the plot represents an observation in the dataset. If there is one independent variable, it is typically represented on the x-axis, and the dependent variable is on the y-axis. Each point denotes a pair of values (x, y) from the dataset.
2. **Regression Line:** Like linear regression, Lasso regression includes a regression line that best fits the data points. This line is determined by the coefficients obtained during the model training, which include a regularization term.

COEFFICIENT OF DETERMINATION

The coefficient of determination, denoted as R-squared (R2), serves as a metric indicating the quality of fit for a model. In the realm of regression analysis, it functions as a statistical indicator of the extent to which the regression line accurately represents the observed data.

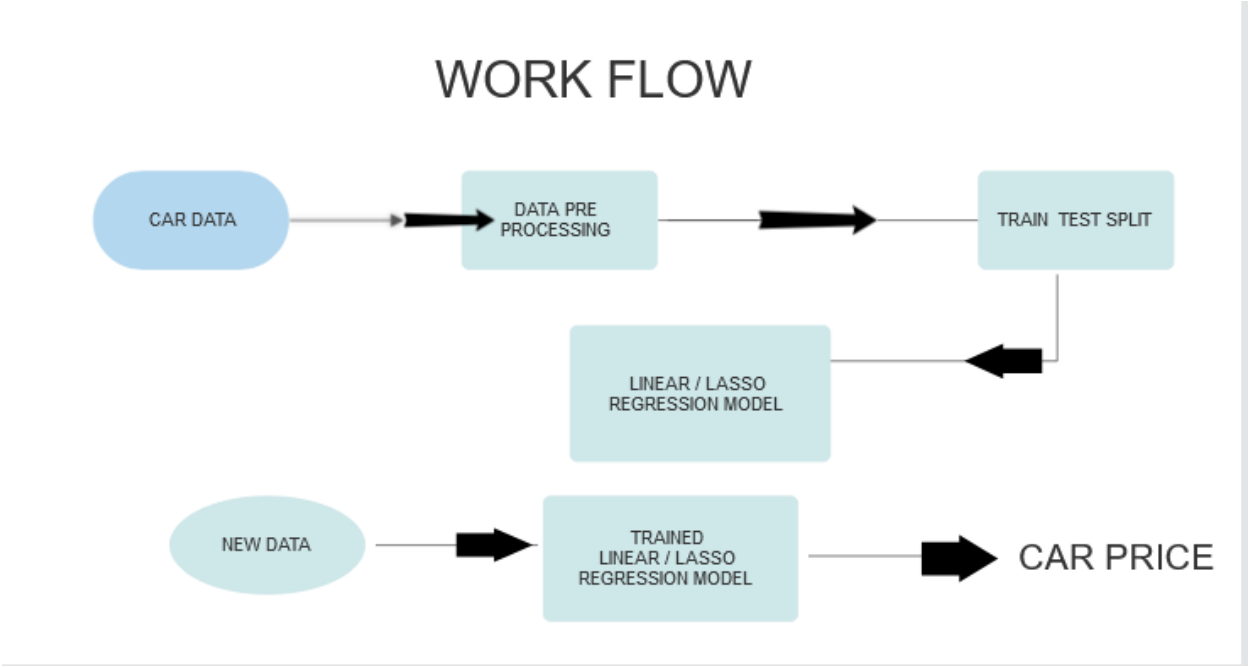
We will be following various steps towards making our Car Price Prediction Project and which involves several steps, including data preprocessing, model training, and model evaluation. Here's an overview of the process:

- **Data Preprocessing**: First, you need to load and preprocess the Car's dataset. This typically involves importing the dataset into a

suitable data structure (e.g., a Data Frame in Python), handling missing values, and splitting the dataset into features (X) and labels (y).

- **Encoding Data:** To enhance the computer's understanding, convert specific data into a more computationally friendly format. For instance, transform repetitive data into numerical representations to facilitate comprehension by the model.
- **Splitting the Dataset**: Next, you need to split the dataset into training and testing sets. The training set is used to train the SVM model, while the testing set is used to evaluate the model's performance.
- **Model Training**: Once the dataset has been pre-processed and split, you can train a Linear Regression model on the training set. You can import Linear Regression and Lasso from the Scikit-learn library in Python and train the model using the training data.
- **Model Evaluation**: After training the model, you need to evaluate its performance on the testing set. You can use various evaluation metrics such as accuracy, precision, recall, and F1score to assess the model's performance. You can also use techniques like cross-validation to get a more reliable estimate of the model's performance.
- **Visualization**: Visualize the actual prices and Predicted prices. Visualisation can be done by many graphs such as scatter plot, linear regression line, etcetera
- **Model Deployment**: Once you are satisfied with the model's performance, you can deploy the model to a production environment and use it for making predictions on new, unseen data.

BASIC DIAGRAM OF THE WORK FLOW



Furthermore, we will explore feature selection techniques to identify the most relevant features that contribute to the prediction of car's price. Feature selection is an essential step in machine learning as it helps to reduce the dimensionality of the dataset, improve model performance, and interpret the model's results.

Finally, we will interpret the Linear and Lasso Regression models to gain insights into which one is better at predicting car prices. We will also discuss the limitations of the proposed model and potential avenues for future research to further improve the accuracy and applicability of the model in real-world settings.

PROBLEM STATEMENT

In the context of the automotive industry, the goal is to develop a predictive model for car prices.

This model should leverage historical data on various car attributes such as model year, distance driven, fuel type, engine, and additional features to accurately estimate the market value of cars. The objective is to create a robust and scalable solution that provides reliable predictions, aiding car manufacturers, dealerships, and consumers in making informed decisions about pricing and purchasing.

The challenge involves selecting and implementing appropriate machine learning techniques, such as linear regression or advanced methods like Lasso regression, to achieve accurate and timely predictions in a dynamic and competitive market.

SOFTWARE REQUIREMENT SPECIFICATIONS

➢ PYTHON:

Python is a well-liked computer programming language used for data analysis, process automation, and the creation of websites and applications. Since Python is a general-purpose language, it may be used to create a wide range of programs without being specifically designed for any one problem. Due to its versatility and ease of use for beginners, it has become the most popular programming language available today.

Python is widely used for work automation, data analysis, data visualization, and website and application creation. Numerous non-programmers, such as Scientists and accountants have embraced Python because of its related easy to use and versatile for a range of everyday tasks, such as controlling finances.

What can we do with python? Some things include:

- Data analysis and machine learning
- Web development
- Automation or scripting
- Software testing and prototyping

➢ GOOGLE COLAB:

Colab is a free Jupyter notebook environment that runs entirely in the cloud. Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members – just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook.

What Colab Offers You?

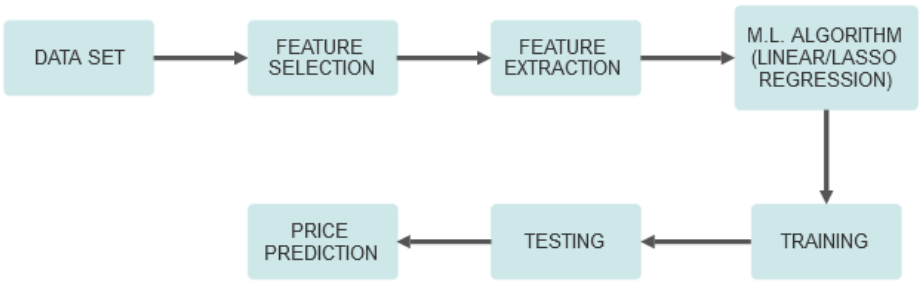
As a programmer, you can perform the following using Google Colab.

- Write and execute code in Python
- Document your code that supports mathematical equations
- Create/Upload/Share notebooks
- Import/Save notebooks from/to Google Drive
- Import/Publish notebooks from GitHub
- Import external datasets e.g., from Kaggle
- Integrate PyTorch, TensorFlow, Keras, OpenCV

➢ Some libraries (seaborn, matplot, Pandas, Scikits Learn etc):

1. **NumPy:**
   It is used for working with arrays and linear algebra. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices, and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.
2. **Pandas:**
   It is used for data analysis and data pre-processing, CSV file I/O (e.g. *pd.read_csv).* It is built on top of another package named Numpy, which provides support for multi-dimensional arrays.
3. **Sklearn:**
   It is used for making use of Machine learning tools. Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.
4. **Matplotlib:**
   It is a collection of functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.
5. **Seaborn:**
   Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. For a brief introduction to the ideas behind the library, you can read the introductory notes or the paper.

ENITITY RELATIONSHIP DIAGRAM



- **STEP 1: Retrieve Dataset**
- **STEP 2: Feature Selection**
- **STEP 3: Feature Extraction**
- **STEP 4: Applying SVM Algorithm**
- **STEP 5: Training**
- **STEP 6: Testing**
- **STEP 7: Obtain Result**

SNAPSHOTS OF THE DIFFERENT INPUT AND OUTPUT SCREENS

1. IMPORTING THE DEPENDENCIES

Importing Dependencies

Libraries

1]:

```
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
```

2. DATA COLLECTION AND ANALYSIS

]:

```
# loading the data from csv file to pandas dataframe
car_dataset = pd.read_csv('/content/car data edited.csv')
```

]:

```
# inspecting the first 5 rows of the dataframe
car_dataset.head()
```

]:

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	engine(CC)	seats
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0	1248	5
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0	1498	5
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0	1497	5
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0	1396	5
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0	1298	5

:

```
# checking the number of rows and columns
car_dataset.shape
```

:

```
(301, 11)
```

:

```
# getting some information about the dataset
car_dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 11 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
 9   engine(CC)     301 non-null    int64
 10  seats          301 non-null    int64
dtypes: float64(2), int64(5), object(4)
memory usage: 26.0+ KB
```

```
# checking the number of missing values
car_dataset.isnull().sum()
```

```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Kms_Driven       0
Fuel_Type        0
Seller_Type      0
Transmission     0
Owner            0
engine(CC)       0
seats            0
dtype: int64
```

```
# checking the distribution of categorical data
print(car_dataset.Fuel_Type.value_counts())
print(car_dataset.Seller_Type.value_counts())
print(car_dataset.Transmission.value_counts())
```

```
Petrol    239
Diesel     60
CNG         2
Name: Fuel_Type, dtype: int64
Dealer        195
Individual    106
Name: Seller_Type, dtype: int64
Manual       261
Automatic     40
Name: Transmission, dtype: int64
```

3. DATA PRE-PROCESING

```
# 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	engine(CC)	seats
0	ritz	2014	3.35	5.59	27000	0	0	0	0	1248	5
1	sx4	2013	4.75	9.54	43000	1	0	0	0	1498	5
2	ciaz	2017	7.25	9.85	6900	0	0	0	0	1497	5
3	wagon r	2011	2.85	4.15	5200	0	0	0	0	1396	5
4	swift	2014	4.60	6.87	42450	1	0	0	0	1298	5

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

```
print(Y)
```

```
0       3.35
1       4.75
2       7.25
3       2.85
4       4.60
       ...
296     9.50
297     4.00
298     3.35
299    11.50
300     5.30
Name: Selling_Price, Length: 301, dtype: float64
```

```
print(X)
```

```
     Year  Present_Price  Kms_Driven  Fuel_Type  Seller_Type  Transmission  \
0    2014           5.59       27000          0            0             0
1    2013           9.54       43000          1            0             0
2    2017           9.85        6900          0            0             0
3    2011           4.15        5200          0            0             0
4    2014           6.87       42450          1            0             0
..    ...            ...         ...        ...          ...           ...
296  2016          11.60       33988          1            0             0
297  2015           5.90       60000          0            0             0
298  2009          11.00       87934          0            0             0
299  2017          12.50        9000          1            0             0
300  2016           5.90        5464          0            0             0

     Owner  engine(CC)  seats
0        0        1248      5
1        0        1498      5
2        0        1497      5
3        0        1396      5
4        0        1298      5
..     ...         ...    ...
296      0        1197      5
297      0        1199      5
298      0        1199      5
299      0         796      4
300      0        1498      5

[301 rows x 9 columns]
```

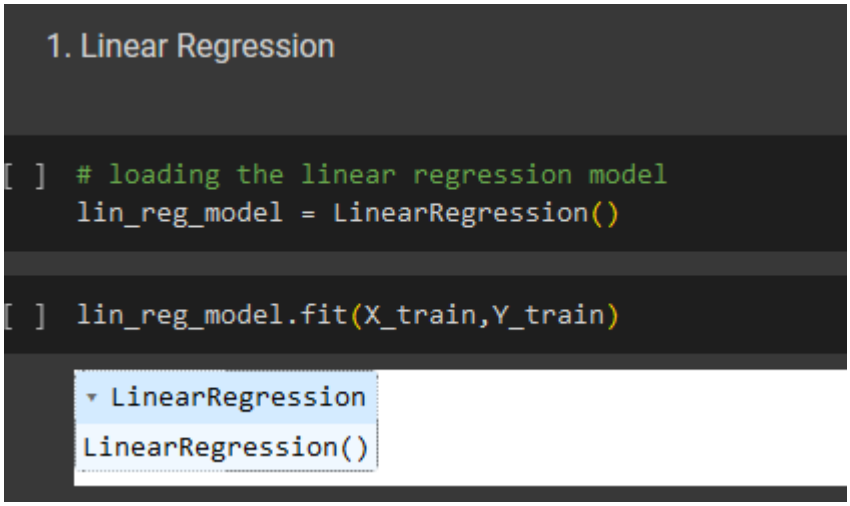
4. SPLITTING THE DATA

Splitting Training and Test data

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

5. MODEL TRAINING

a LINEAR REGRESSION



a1. MODEL EVALUATION (ACCURACY SCORE)

Model Evaluation

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

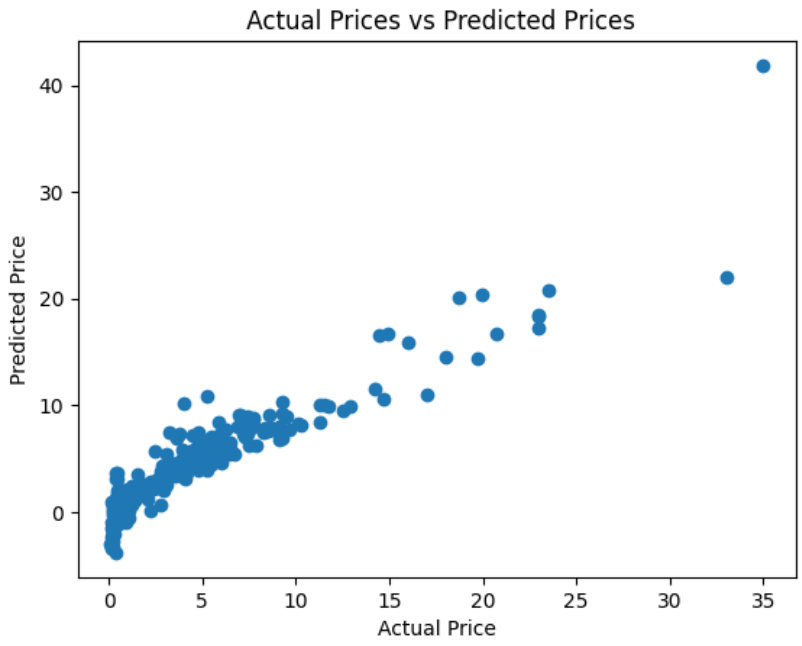
```
# R squared Error
error_score = metrics.r2_score(Y_train, training_data_prediction)
print("R squared Error : ", error_score)
```

```
R squared Error :  0.8805572443600341
```

a2. VISUALISE THE ACTUAL AND THE PREDICTED PRICE

i. Training data prediction

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



ii. Testing data prediction

```
# 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 :  0.8404518564905336
```

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

b LASSO REGRESSION

2. Lasso Regression

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

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

```
Lasso()
```

b1. MODEL EVALUATION (ACCURACY SCORE)

Model Evaluation

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

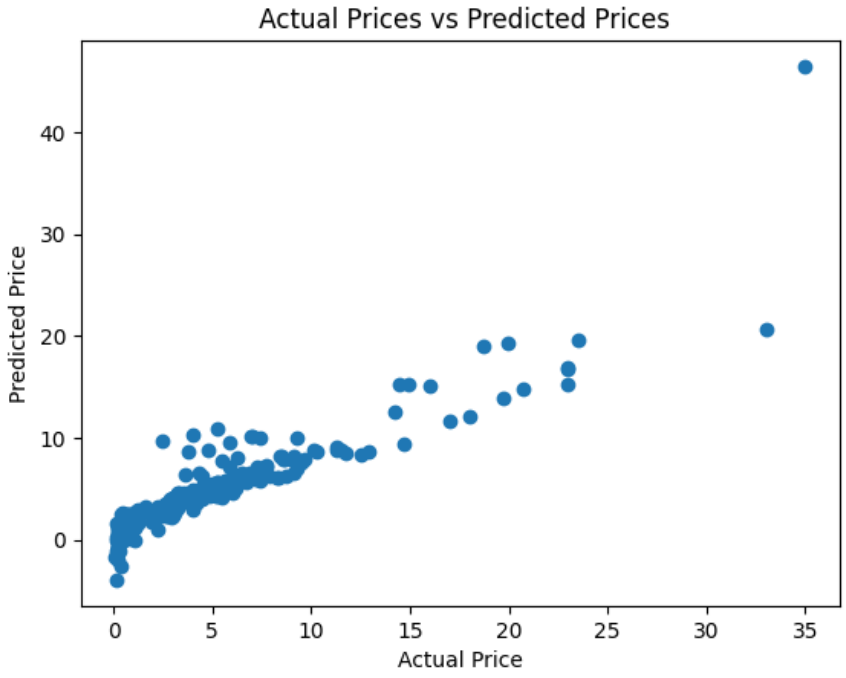
```
# R squared Error
error_score = metrics.r2_score(Y_train, training_data_prediction)
print("R squared Error : ", error_score)
```

```
R squared Error :  0.843903559389372
```

b2. VISUALISE THE ACTUAL AND THE PREDICTED PRICE

i. Training data prediction

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



ii. Testing data prediction

```
# 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 :  0.8746461685347716
```

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

CONCLUSION

In conclusion, my machine learning project on Car Price Prediction using Linear Regression and Lasso Regression has provided valuable insights and outcomes. Through the analysis and implementation of these two popular algorithms, I have learned several key points:

1. **Feature selection**: Selecting relevant features is crucial for improving the accuracy of machine learning models. I learned that careful feature selection, such as using domain knowledge and feature importance analysis, can significantly impact the performance of Linear and Lasso Regressing in predicting a car's price.
2. **Model performance**: I have gained an understanding of how Linear Regression and Lasso Regression can be utilized for car price prediction, and how their performance can be evaluated using metrics such as accuracy. The error score of the Lasso Regression model was better compared to the Linear Regression model, making the Lasso model a better choice for prediction in this case.
3. **Visualization:** Visualisation was done by plotting a scatter graph which gives us a better understanding of the relations between the actual and predicted price.

Overall, this project has helped me grasp the fundamental concepts of machine learning, including feature selection, data preprocessing, encoding data, splitting the data, model evaluation and interpretation. I can now confidently apply these concepts and techniques to future machine learning projects and continue to refine my skills in data analysis and predictive modelling.

LIMITATIONS

Machine learning is a powerful tool that can be used to analyze and make predictions from car's datasets.

However, like any other approach, it has its limitations. Here are some limitations of using machine learning on car's datasets:

1. **Feature Selection**: Selecting the most relevant features or variables from the car's dataset can be challenging. If important features are overlooked or irrelevant features are included, it can affect the model's performance and interpretability.
2. **Changing Dynamics**: A car is a complex and dynamic structure that will always evolve over time. Machine learning models trained on a specific dataset may become less accurate as new data becomes available, requiring regular updates and monitoring to maintain their performance
3. **Lack of Data Diversity**: For machine learning algorithms to effectively generalize, they need a variety of representative data sets. The model might not function well on real-world data that is different from the training data if the car's dataset, which was used for training, is not sufficiently diverse in terms of the car's body condition, engine sustainability, or other pertinent characteristics.
4. **Data Imbalance**: Car's datasets may be imbalanced, meaning that the distribution of current price and selling price may be disproportionate. This can lead to biased model predictions, as the model may be more accurate in predicting the majority class and less accurate in predicting the minority class.
5. **Overfitting**: Machine learning models can be prone to Overfitting, where the model learns to perform well on the training data but does not generalize well to new, unseen data. This can happen if the model is too complex or if the dataset used for training is small, leading to a high risk of Overfitting.
6. **Limited Data Quality**: The accuracy and dependability of the outcomes might be strongly impacted by the quality of the automobile dataset employed in machine learning. Predictions may be skewed or incorrect if the dataset is corrupted, missing values exist, or is incomplete.

7. **Limited Interpretability**: Some machine learning algorithms, such as deep learning models, may lack interpretability, making it difficult to understand and explain the reasons behind their

predictions. This can be a limitation when it comes to gaining trust and acceptance from stakeholders, including customers and regulatory authorities.

It is important to be aware of these limitations when using machine learning on car's datasets and to carefully evaluate the predictions and interpretations. Collaborating with professionals and domain experts can help mitigate some of these limitations and ensure that the machine learning models are used responsibly and effectively.

BIBLIOGRAPHY

- [1] https://en.wikipedia.org/wiki/Linear_regression#/media/File:Linear_least_squares_example2.svg
- https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho
- https://stackoverflow.com/