Machine Learning Project: Car Price Prediction

Introduction:

In this transcript, we will explore a machine learning project that focuses on predicting the market price of a car based on its different features. Before delving into the project, we will first introduce some fundamental machine learning concepts to ensure a clear understanding of the project.

Machine Learning Basics:

1. What is Machine Learning?

- Definition: Machine learning is a branch of AI that utilizes data and algorithms to mimic human learning, enabling computers to learn without explicit programming. Basically, in simple words, it is the way to train a machine that how a human mind is learned from everything by itself

- Mostly we use two types of machine learning;

1. Supervised learning

2. Unsupervised Learning

Let's Understand what is supervised and what is unsupervised learning.

Supervised Learning:

1. Definition: Supervised learning involves training a model with labeled data, where each example consists of an input object and a desired output value.

2. Example: Categorizing objects (apples, mangoes, oranges, etc.) and training a model to recognize and label them accurately.

Unsupervised Learning:

1. Definition: Unsupervised learning involves training a model with unlabeled data, where the algorithm must discover patterns and relationships within the data.

2. Example: Grouping objects (mangoes, apples, balls, cubes) based on their characteristics without explicitly defining their categories.

-We are going to do this project through a supervised learning, so supervised learning also have two branches, one is regression analysis and one another is classification analysis.

Regression and Classification Analysis:

1. Regression Analysis: A type of supervised learning used to predict continuous real-value outputs, such as predicting the price of a car based on its features.

2. Classification Analysis: Another type of supervised learning used to predict discrete categories or classes, such as classifying emails as spam or non-spam based on their content.

Project Explanation:

Project Approach: This project employs supervised learning with a focus on regression analysis.

Data Preprocessing: The data set, obtained from Kaggle.com, undergoes preprocessing to prepare it for training.

- Utilizing Python libraries: Pandas for data manipulation and NumPy for numerical computations.

- Converting CSV file to a data frame using the pandas library.

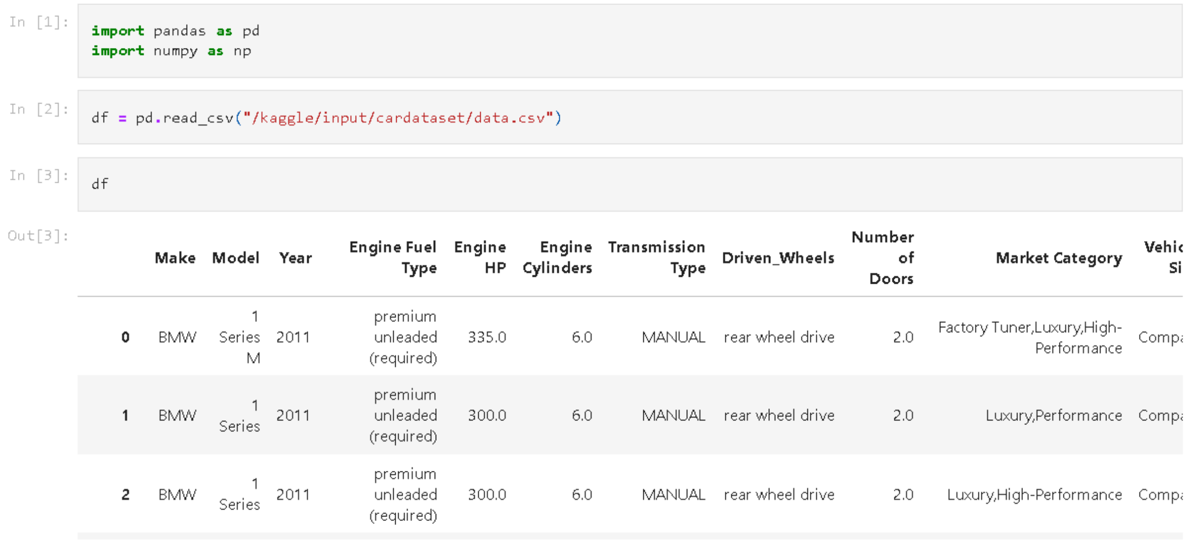
- Handling missing values, transforming categorical data into numerical representation, and ensuring data types are suitable for training.

Lets Understand what is Python Libraries and what is the basic uses of Pandas and Numpy?

- **Python Library:** A Python library is a collection of pre-written code modules or packages that provide various functionalities and tools to simplify and streamline programming tasks. These libraries contain reusable functions, classes, and methods that can be imported into Python programs to extend their capabilities. Python libraries are designed to address specific tasks or domains, such as data manipulation, numerical computations, web development, machine learning, and more.
- **Pandas:** Pandas is a powerful open-source Python library widely used for data manipulation, analysis, and cleaning tasks. It provides easy-to-use data structures and data analysis tools, making it a popular choice among data scientists, analysts, and researchers.
- **Numpy:** NumPy is a fundamental Python library for numerical computing. It provides a powerful array object, along with functions and tools for working with arrays and performing numerical operations efficiently. NumPy is widely used in scientific computing, data analysis, and machine learning.

Project Guiding Report

1.Data Loading and Exploration:



```
In [1]: import pandas as pd
        import numpy as np
```

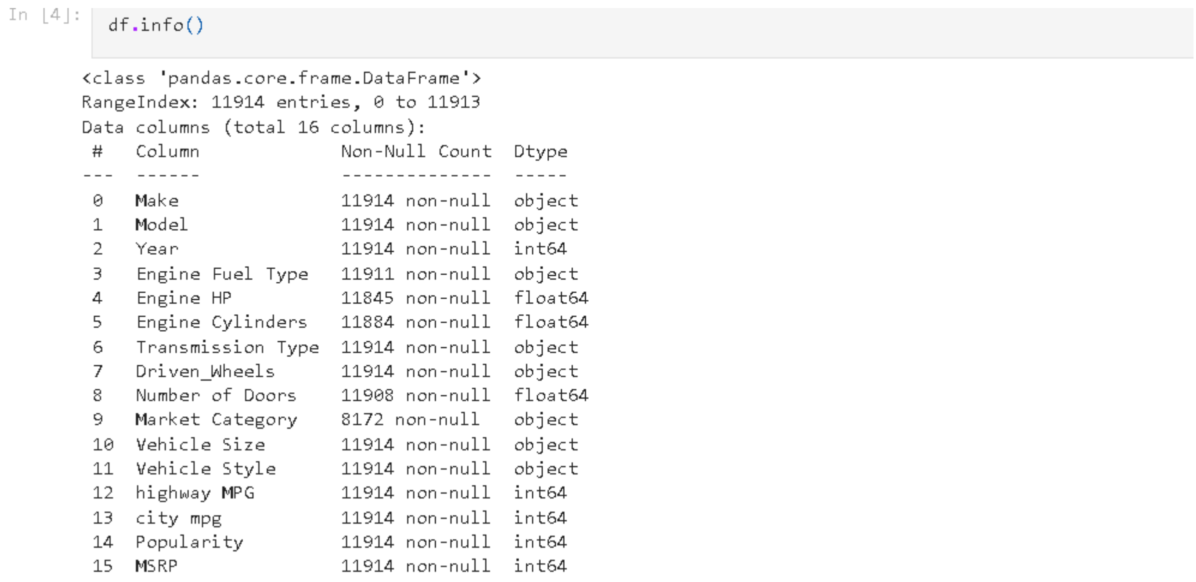
```
In [2]: df = pd.read_csv("/kaggle/input/cardataset/data.csv")
```

```
In [3]: df
```

Out[3]:

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehic Si
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compa
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compa
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compa

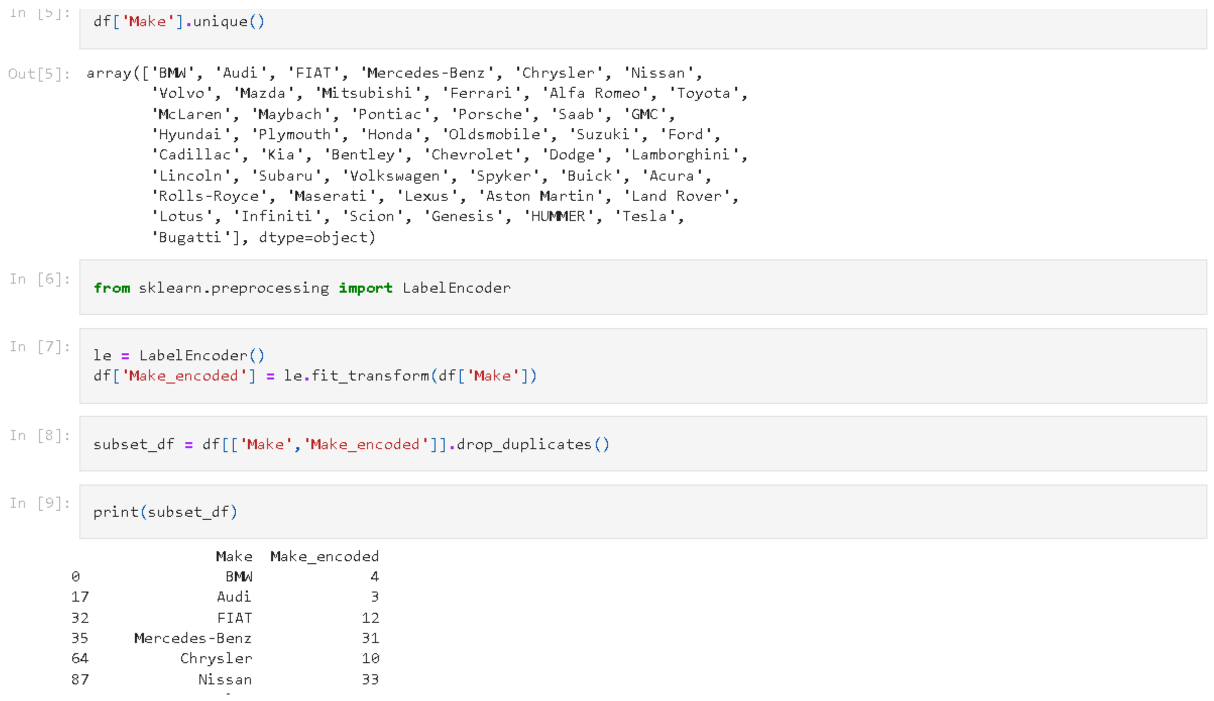
- The code begins by importing the necessary libraries, including pandas and numpy.
- It reads a CSV file called "data.csv" using the pandas library and assigns it to the variable df.
- The code then displays the DataFrame, df, which provides an overview of the dataset's structure and the first few rows of data.



```
In [4]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Make               11914 non-null  object
 1   Model              11914 non-null  object
 2   Year               11914 non-null  int64
 3   Engine Fuel Type   11911 non-null  object
 4   Engine HP          11845 non-null  float64
 5   Engine Cylinders   11884 non-null  float64
 6   Transmission Type  11914 non-null  object
 7   Driven_Wheels      11914 non-null  object
 8   Number of Doors    11908 non-null  float64
 9   Market Category    8172 non-null   object
 10  Vehicle Size       11914 non-null  object
 11  Vehicle Style      11914 non-null  object
 12  highway MPG        11914 non-null  int64
 13  city mpg           11914 non-null  int64
 14  Popularity         11914 non-null  int64
 15  MSRP               11914 non-null  int64
```

- The df.info() function provides information about the DataFrame, including the data types of each column and the count of non-null values.

2.Preprocessing and Label Encoding:



```
In [5]: df['Make'].unique()

Out[5]: array(['BMW', 'Audi', 'FIAT', 'Mercedes-Benz', 'Chrysler', 'Nissan',
               'Volvo', 'Mazda', 'Mitsubishi', 'Ferrari', 'Alfa Romeo', 'Toyota',
               'McLaren', 'Maybach', 'Pontiac', 'Porsche', 'Saab', 'GMC',
               'Hyundai', 'Plymouth', 'Honda', 'Oldsmobile', 'Suzuki', 'Ford',
               'Cadillac', 'Kia', 'Bentley', 'Chevrolet', 'Dodge', 'Lamborghini',
               'Lincoln', 'Subaru', 'Volkswagen', 'Spyker', 'Buick', 'Acura',
               'Rolls-Royce', 'Maserati', 'Lexus', 'Aston Martin', 'Land Rover',
               'Lotus', 'Infiniti', 'Scion', 'Genesis', 'HUMMER', 'Tesla',
               'Bugatti'], dtype=object)
```

```
In [6]: from sklearn.preprocessing import LabelEncoder
```

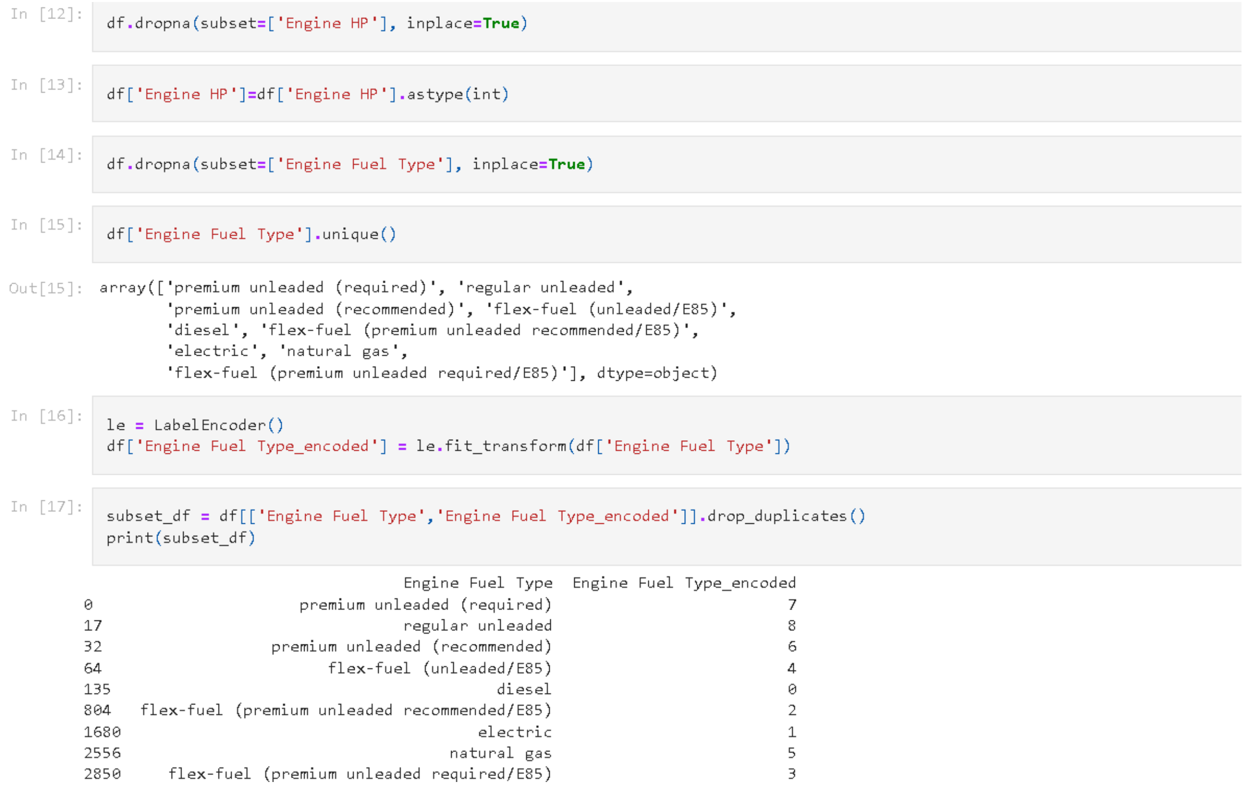
```
In [7]: le = LabelEncoder()
        df['Make_encoded'] = le.fit_transform(df['Make'])
```

```
In [8]: subset_df = df[['Make','Make_encoded']].drop_duplicates()
```

```
In [9]: print(subset_df)

             Make  Make_encoded
0             BMW             4
17           Audi             3
32           FIAT            12
35  Mercedes-Benz            31
64       Chrysler            10
87         Nissan            33
```

- The code performs preprocessing steps on specific columns to make them suitable for training.
- The 'Make' column, representing the car's brand, is encoded using LabelEncoder, which assigns a unique integer value to each brand.
- The 'Engine Fuel Type' column is also encoded using LabelEncoder to convert the fuel types into corresponding integers.
- The subset_df DataFrame is created to display the unique values of the 'Make' column and their corresponding encoded values.

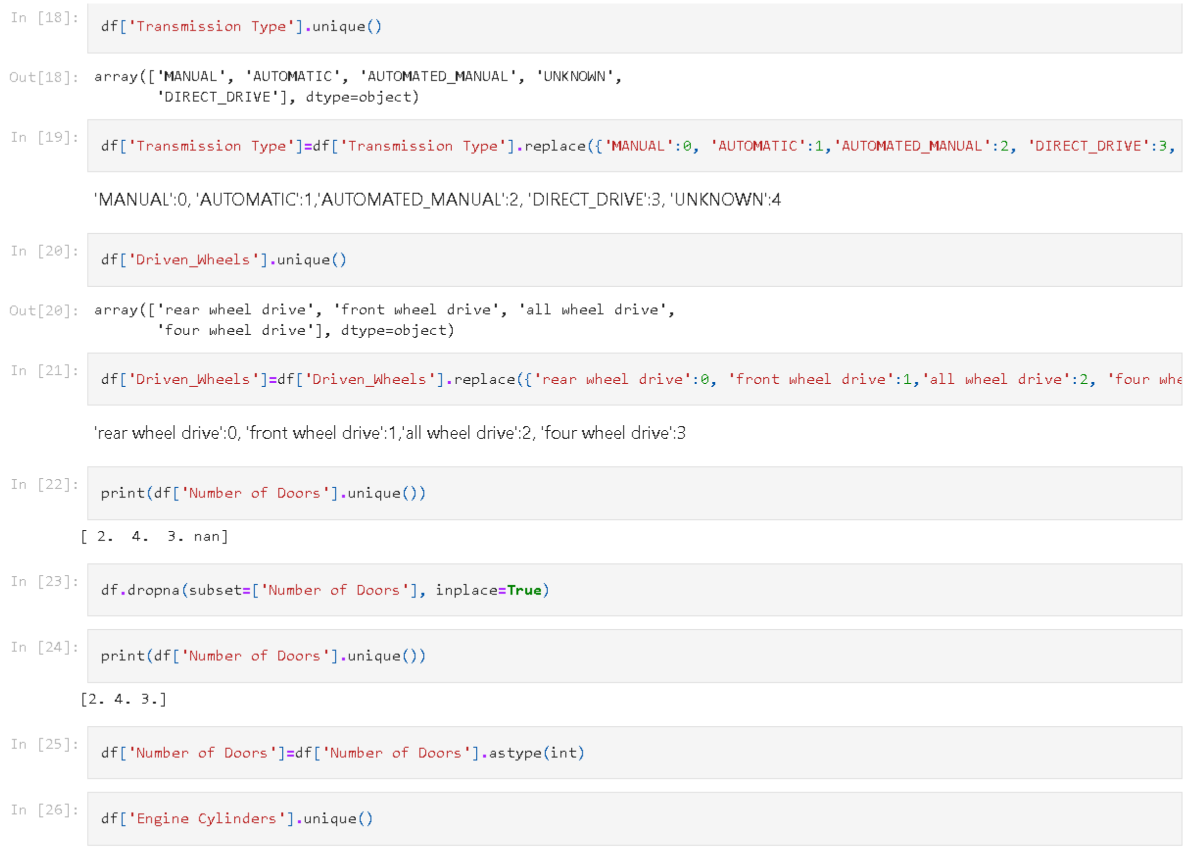
3.Data Cleaning and Formatting:



```
In [12]:  df.dropna(subset=['Engine HP'], inplace=True)

In [13]:  df['Engine HP']=df['Engine HP'].astype(int)

In [14]:  df.dropna(subset=['Engine Fuel Type'], inplace=True)

In [15]:  df['Engine Fuel Type'].unique()

Out[15]: array(['premium unleaded (required)', 'regular unleaded',
                'premium unleaded (recommended)', 'flex-fuel (unleaded/E85)',
                'diesel', 'flex-fuel (premium unleaded recommended/E85)',
                'electric', 'natural gas',
                'flex-fuel (premium unleaded required/E85)'], dtype=object)

In [16]:  le = LabelEncoder()
          df['Engine Fuel Type_encoded'] = le.fit_transform(df['Engine Fuel Type'])

In [17]:  subset_df = df[['Engine Fuel Type','Engine Fuel Type_encoded']].drop_duplicates()
          print(subset_df)

                                  Engine Fuel Type  Engine Fuel Type_encoded
0                      premium unleaded (required)                         7
17                                regular unleaded                         8
32                  premium unleaded (recommended)                         6
64                        flex-fuel (unleaded/E85)                         4
135                                         diesel                         0
804   flex-fuel (premium unleaded recommended/E85)                         2
1680                                      electric                         1
2556                                   natural gas                         5
2850     flex-fuel (premium unleaded required/E85)                         3
```

- The code drops rows with missing values in the 'Engine HP' column and converts the column to the integer data type.
- Similarly, it drops rows with missing values in the 'Engine Fuel Type' column.
- The unique values of the 'Engine Fuel Type' column and their encoded values are displayed using the subset_df DataFrame.

4.Encoding Categorical Columns:



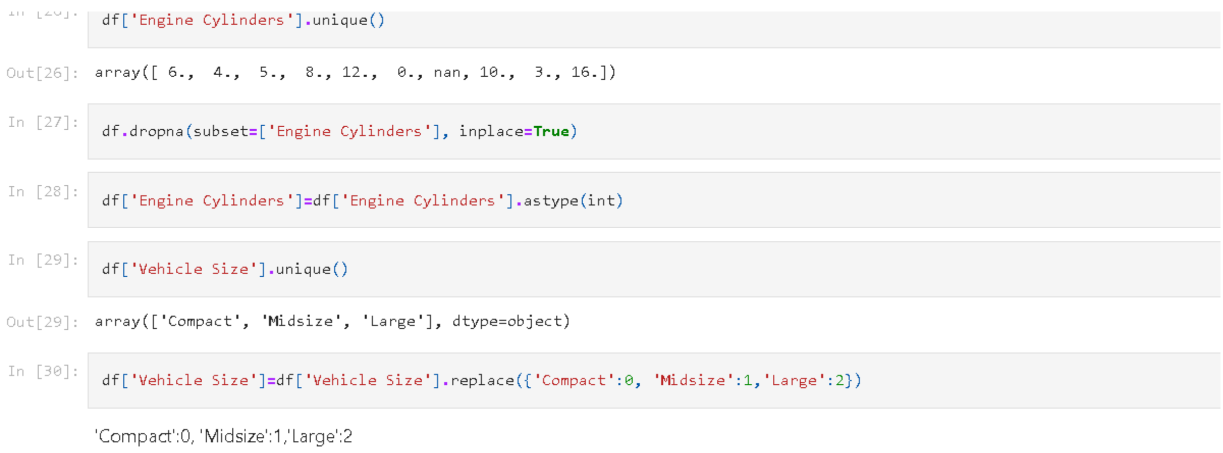
```
In [18]:  df['Transmission Type'].unique()

Out[18]: array(['MANUAL', 'AUTOMATIC', 'AUTOMATED_MANUAL', 'UNKNOWN',
                'DIRECT_DRIVE'], dtype=object)

In [19]:  df['Transmission Type']=df['Transmission Type'].replace({'MANUAL':0, 'AUTOMATIC':1,'AUTOMATED_MANUAL':2, 'DIRECT_DRIVE':3,
```

'MANUAL':0, 'AUTOMATIC':1,'AUTOMATED_MANUAL':2, 'DIRECT_DRIVE':3, 'UNKNOWN':4

```
In [20]:  df['Driven_Wheels'].unique()

Out[20]: array(['rear wheel drive', 'front wheel drive', 'all wheel drive',
                'four wheel drive'], dtype=object)

In [21]:  df['Driven_Wheels']=df['Driven_Wheels'].replace({'rear wheel drive':0, 'front wheel drive':1,'all wheel drive':2, 'four whe
```

'rear wheel drive':0, 'front wheel drive':1,'all wheel drive':2, 'four wheel drive':3

```
In [22]:  print(df['Number of Doors'].unique())

         [ 2.  4.  3. nan]

In [23]:  df.dropna(subset=['Number of Doors'], inplace=True)

In [24]:  print(df['Number of Doors'].unique())

         [2. 4. 3.]

In [25]:  df['Number of Doors']=df['Number of Doors'].astype(int)

In [26]:  df['Engine Cylinders'].unique()
```

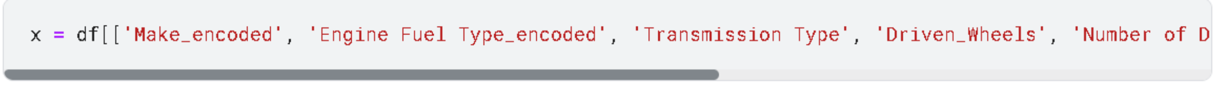


```
In [26]:  df['Engine Cylinders'].unique()

Out[26]: array([ 6.,  4.,  5.,  8., 12.,  0., nan, 10.,  3., 16.])

In [27]:  df.dropna(subset=['Engine Cylinders'], inplace=True)

In [28]:  df['Engine Cylinders']=df['Engine Cylinders'].astype(int)

In [29]:  df['Vehicle Size'].unique()

Out[29]: array(['Compact', 'Midsize', 'Large'], dtype=object)

In [30]:  df['Vehicle Size']=df['Vehicle Size'].replace({'Compact':0, 'Midsize':1,'Large':2})
```

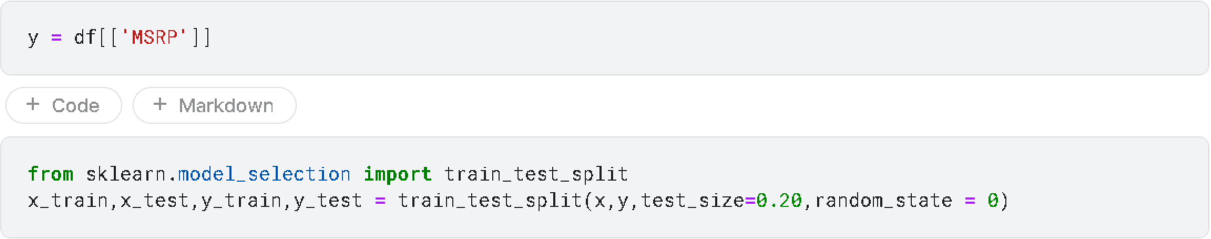
'Compact':0, 'Midsize':1,'Large':2

- The 'Transmission Type' column is encoded by replacing specific values with integers representing different transmission types.
- The 'Driven_Wheels' column is encoded by replacing specific values with integers representing different wheel drive types.
- The 'Number of Doors' column is converted to integers by dropping rows with missing values and converting the column to the integer data type.
- The 'Engine Cylinders' column is converted to integers by dropping rows with missing values and converting the column to the integer data type.
- The 'Vehicle Size' column is encoded by replacing specific values with integers representing different vehicle sizes.

5.Feature Selection and Splitting:



```
x = df[['Make_encoded', 'Engine Fuel Type_encoded', 'Transmission Type', 'Driven_Wheels', 'Number of D
```

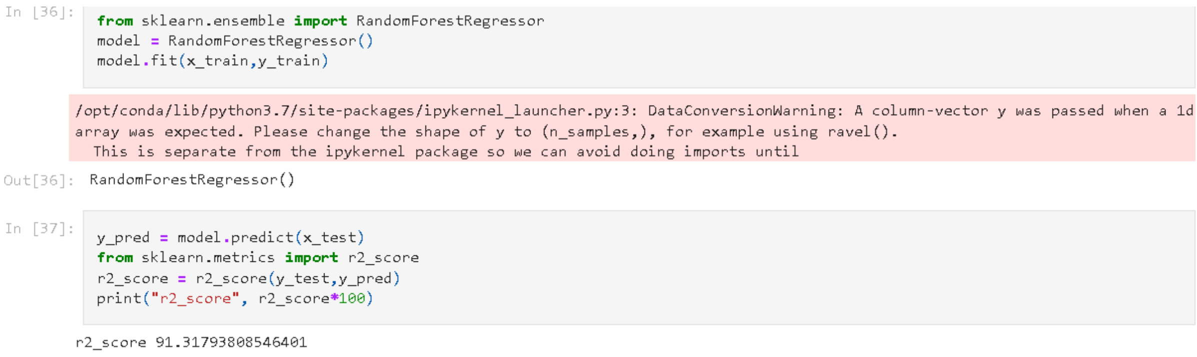


```
y = df[['MSRP']]
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.20,random_state = 0)
```

- The code selects the relevant features ('Make_encoded', 'Engine Fuel Type_encoded', 'Transmission Type', 'Driven_Wheels', 'Number of Doors', 'Engine Cylinders', 'Vehicle Size', 'highway MPG', 'Engine HP') from the DataFrame and assigns them to the variable x.
- The target variable ('MSRP') is assigned to the variable y.
- The dataset is split into training and testing sets using the train_test_split function from sklearn.model_selection.

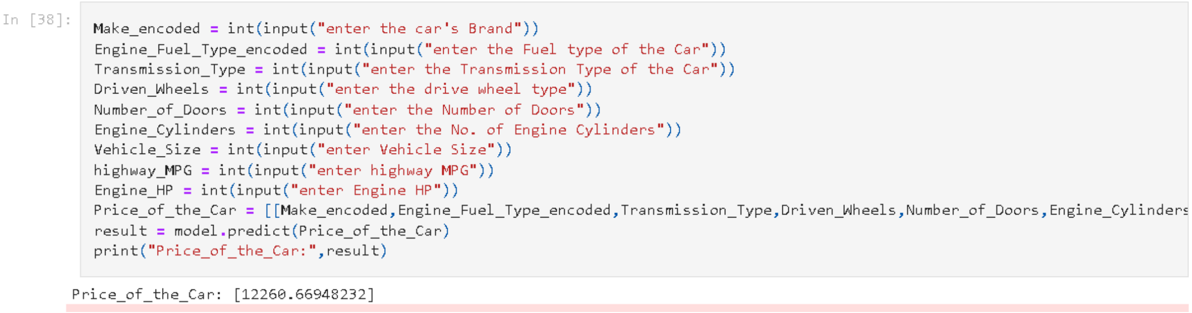
6.Model Training and Evaluation:



```
In [36]:  from sklearn.ensemble import RandomForestRegressor
          model = RandomForestRegressor()
          model.fit(x_train,y_train)

/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:3: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  This is separate from the ipykernel package so we can avoid doing imports until

Out[36]:  RandomForestRegressor()

In [37]:  y_pred = model.predict(x_test)
          from sklearn.metrics import r2_score
          r2_score = r2_score(y_test,y_pred)
          print("r2_score", r2_score*100)

r2_score 91.31793808546401
```

- The code imports the RandomForestRegressor model from the sklearn.ensemble module.
- The model is trained on the training data using the fit method.
- Predictions are made on the testing data using the predict method, and the r2_score is calculated to evaluate the model's performance.
- The r2_score, representing the coefficient of determination, is printed to assess how well the model predicts the car prices.

7.Price Prediction:



```
In [38]:  Make_encoded = int(input("enter the car's Brand"))
          Engine_Fuel_Type_encoded = int(input("enter the Fuel type of the Car"))
          Transmission_Type = int(input("enter the Transmission Type of the Car"))
          Driven_Wheels = int(input("enter the drive wheel type"))
          Number_of_Doors = int(input("enter the Number of Doors"))
          Engine_Cylinders = int(input("enter the No. of Engine Cylinders"))
          Vehicle_Size = int(input("enter Vehicle Size"))
          highway_MPG = int(input("enter highway MPG"))
          Engine_HP = int(input("enter Engine HP"))
          Price_of_the_Car = [[Make_encoded,Engine_Fuel_Type_encoded,Transmission_Type,Driven_Wheels,Number_of_Doors,Engine_Cylinders
          result = model.predict(Price_of_the_Car)
          print("Price_of_the_Car:",result)

Price_of_the_Car: [12260.66948232]
```

- The code prompts the user to enter various features of a car for which they want to predict the price.
- The entered features are stored in the 'Price_of_the_Car' variable.
- The trained model is used to predict the price based on the entered features.
- The predicted price is displayed as the output.

Conclusion:

This transcript covered a machine learning project focused on predicting car market prices. By understanding key concepts, performing data preprocessing, training the model, and evaluating its accuracy, we can utilize supervised learning techniques to predict car prices based on their features effectively.