In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

In [2]:
```
df=pd.read_csv(r"c7_used_cars.csv")
df
```

Out[2]:

	Unnamed: 0	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	0	T-Roc	2019	25000	Automatic	13904	Diesel	145	49.6	2.0
1	1	T-Roc	2019	26883	Automatic	4562	Diesel	145	49.6	2.0
2	2	T-Roc	2019	20000	Manual	7414	Diesel	145	50.4	2.0
3	3	T-Roc	2019	33492	Automatic	4825	Petrol	145	32.5	2.0
4	4	T-Roc	2019	22900	Semi-Auto	6500	Petrol	150	39.8	1.5
...	...	...	...	...	...	...	...	...	...	...
99182	10663	A3	2020	16999	Manual	4018	Petrol	145	49.6	1.0
99183	10664	A3	2020	16999	Manual	1978	Petrol	150	49.6	1.0
99184	10665	A3	2020	17199	Manual	609	Petrol	150	49.6	1.0
99185	10666	Q3	2017	19499	Automatic	8646	Petrol	150	47.9	1.4
99186	10667	Q3	2016	15999	Manual	11855	Petrol	150	47.9	1.4

99187 rows × 11 columns

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99187 entries, 0 to 99186
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Unnamed: 0    99187 non-null  int64  
 1   model         99187 non-null  object 
 2   year          99187 non-null  int64  
 3   price         99187 non-null  int64  
 4   transmission  99187 non-null  object 
 5   mileage       99187 non-null  int64  
 6   fuelType      99187 non-null  object 
 7   tax           99187 non-null  int64  
 8   mpg           99187 non-null  float64
 9   engineSize    99187 non-null  float64
 10  Make          99187 non-null  object 
dtypes: float64(2), int64(5), object(4)
memory usage: 8.3+ MB
```

In [4]:
```
df=df.dropna()
```

In [5]:
```
df.describe()
```

Out[5]:

	Unnamed: 0	year	price	mileage	tax	mpg	9
count	99187.000000	99187.000000	99187.000000	99187.000000	99187.000000	99187.000000	
mean	6294.413532	2017.087723	16805.347656	23058.914213	120.299838	55.166825	
std	4265.588536	2.123934	9866.773417	21148.523721	63.150926	16.138522	
min	0.000000	1970.000000	450.000000	1.000000	0.000000	0.300000	
25%	2755.000000	2016.000000	9999.000000	7425.000000	125.000000	47.100000	
50%	5591.000000	2017.000000	14495.000000	17460.000000	145.000000	54.300000	
75%	9420.000000	2019.000000	20870.000000	32339.000000	145.000000	62.800000	
max	17964.000000	2060.000000	159999.000000	323000.000000	580.000000	470.800000	

In [6]:
```
df.columns
```

Out[6]:
```
Index(['Unnamed: 0', 'model', 'year', 'price', 'transmission', 'mileage',
       'fuelType', 'tax', 'mpg', 'engineSize', 'Make'],
      dtype='object')
```

In [21]:
```
x=df[['Unnamed: 0', 'year', 'price', 'mileage',
     'tax', 'mpg', 'engineSize']]
y=df['fuelType']
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
763: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[21]:
```
LogisticRegression()
```

In [13]:
```
lr.predict(x_test)
```

Out[13]:
```
array(['Petrol', 'Diesel', 'Diesel', ..., 'Petrol', 'Petrol', 'Diesel'],
      dtype=object)
```

In [14]:
```
lr.score(x_test,y_test)
```

Out[14]:
```
0.6925765366132338
```

In [15]:
```
from sklearn.preprocessing import StandardScaler
fs=StandardScaler().fit_transform(x)
logr=LogisticRegression()
logr.fit(fs,y)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
763: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
```

Out[15]:
```
LogisticRegression()
```

In [17]:
```
o=[[6,7,8,34,52,6,7]]
prediction=logr.predict(o)
print(prediction)
```

```
['Diesel']
```

In [18]:
```
logr.classes_
```

Out[18]:
```
array(['Diesel', 'Electric', 'Hybrid', 'Other', 'Petrol'], dtype=object)
```

In [19]:
```
logr.predict_proba(o)[0][0]
```

Out[19]:
```
0.9795692751051442
```

In [20]:
```
logr.predict_proba(o)[0][1]
```

Out[20]:
```
1.2174728850843716e-22
```

In [ ]:
```

```