In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```
from sklearn.linear_model import LogisticRegression
```

In [3]:

```
df=pd.read_csv(r"E:\154\c7_used_cars.csv")
df
```

Out[3]:

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

Out[4]:

	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 [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 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: 9.1+ MB
```

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

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

In [7]:
```
feature_matrix=df[['Unnamed: 0','year', 'price',  'mileage',
        'tax', 'mpg', 'engineSize']]
target_vector=df['Make']
```

In [8]:
```
feature_matrix.shape
```

Out[8]:
```
(99187, 7)
```

In [9]:
```
target_vector.shape
```

Out[9]:
```
(99187,)
```

In [10]:
```
from sklearn.preprocessing import StandardScaler
```

In [11]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [12]:
```
logr=LogisticRegression()
logr.fit(fs,target_vector)
```

Out[12]:
```
LogisticRegression()
```

In [13]:
```
observation=[[1,2,3,4,5,6,7]]
```

In [14]:
```
prediction=logr.predict(observation)
print(prediction)
```

```
['BMW']
```

In [15]:
```
logr.classes_
```

Out[15]:
```
array(['Audi', 'BMW', 'VW', 'ford', 'hyundi', 'merc', 'skoda', 'toyota',
       'vauxhall'], dtype=object)
```

In [16]:
```
logr.predict_proba(observation)[0][0]
```

Out[16]:
```
2.7412293064875042e-05
```

In [17]:
```
logr.predict_proba(observation)
```

Out[17]:
```
array([[2.74122931e-05, 9.36836737e-01, 2.51395992e-08, 5.85008303e-09,
        3.09237182e-12, 6.31357545e-02, 6.44018883e-09, 5.85474765e-08,
        7.49581427e-16]])
```

In [18]:
```
df['Make'].value_counts()
```

Out[18]:
```
ford        17965
VW          15157
vauxhall    13632
merc        13119
BMW         10781
Audi        10668
toyota       6738
skoda        6267
hyundi       4860
Name: Make, dtype: int64
```

In [19]:
```
x=df[['Unnamed: 0','year', 'price',  'mileage',
        'tax', 'mpg', 'engineSize']]
y=df['Make']
```

In [20]:
```
g1={ 'Make':{'Audi':1, 'BMW':2, 'VW':3, 'ford':4, 'hyundi':5, 'merc':6, 'skoda
       'vauxhall':9}}
df=df.replace(g1)
df
```

Out[20]:

	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 [21]:
```
from sklearn.model_selection import train_test_split
```

In [22]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [23]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [24]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[24]:
```
RandomForestClassifier()
```

In [25]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf':[5,10,15,20,25],
            'n_estimators':[10,20,30,40,50]
}
```

In [26]:
```
from sklearn.model_selection import GridSearchCV
grid_search =GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="ac
grid_search.fit(x_train,y_train)
```

Out[26]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

In [27]:
```
grid_search.best_score_
```

Out[27]:
```
0.5128762782658793
```

In [28]:
```
rfc_best=grid_search.best_estimator_
```

In [29]:
```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['a','b'
```

```
= [2287, 2028, 2445, 1366, 123, 1108, 597, 145, 243]\nclass = c'),
 Text(2371.5, 543.5999999999999, 'Unnamed: 0 <= 10779.0\ngini = 0.821\nsamp
les = 2688\nvalue = [651, 694, 940, 1152, 75, 233, 194, 99, 241]\nclass =
d'),
 Text(2301.75, 181.19999999999982, 'gini = 0.824\nsamples = 1958\nvalue =
[651, 694, 553, 655, 75, 171, 194, 99, 37]\nclass = b'),
 Text(2441.25, 181.19999999999982, 'gini = 0.666\nsamples = 730\nvalue =
[0, 0, 387, 497, 0, 62, 0, 0, 204]\nclass = d'),
 Text(2650.5, 543.5999999999999, 'year <= 2017.5\ngini = 0.791\nsamples = 3
862\nvalue = [1636, 1334, 1505, 214, 48, 875, 403, 46, 2]\nclass = a'),
 Text(2580.75, 181.19999999999982, 'gini = 0.752\nsamples = 344\nvalue = [1
85, 64, 141, 20, 0, 140, 12, 0, 0]\nclass = a'),
 Text(2720.25, 181.19999999999982, 'gini = 0.791\nsamples = 3518\nvalue =
[1451, 1270, 1364, 194, 48, 735, 391, 46, 2]\nclass = a'),
 Text(3069.0, 906.0, 'mpg <= 44.35\ngini = 0.749\nsamples = 3293\nvalue =
[969, 1728, 322, 90, 256, 1639, 3, 188, 19]\nclass = b'),
 Text(2929.5, 543.5999999999999, 'price <= 22707.0\ngini = 0.783\nsamples =
2195\nvalue = [777, 989, 322, 90, 222, 906, 3, 127, 15]\nclass = b'),
 Text(2859.75, 181.19999999999982, 'gini = 0.775\nsamples = 460\nvalue = [7
3, 278, 53, 11, 136, 90, 3, 58, 15]\nclass = b'),
```

In [ ]:

```
In [ ]:
```