Vijay 02.08.2023

In [ ]:

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

In [10]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\c7_used_cars.csv")
a
```

Out[10]:

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

99187 rows × 11 columns

In [11]:

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

In [12]:
```
a=a.head(99180)
a
```

Out[12]:

	Unnamed: 0	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize	Make
0	0	T-Roc	2019	25000	Automatic	13904	Diesel	145	49.6	2.0	VW
1	1	T-Roc	2019	26883	Automatic	4562	Diesel	145	49.6	2.0	VW
2	2	T-Roc	2019	20000	Manual	7414	Diesel	145	50.4	2.0	VW
3	3	T-Roc	2019	33492	Automatic	4825	Petrol	145	32.5	2.0	VW
4	4	T-Roc	2019	22900	Semi-Auto	6500	Petrol	150	39.8	1.5	VW
...	...	...	...	...	...	...	...	...	...	...	...
99175	10656	A3	2016	15495	Semi-Auto	52500	Hybrid	0	176.6	1.4	Audi
99176	10657	A4	2016	20995	Semi-Auto	23700	Diesel	30	61.4	2.0	Audi
99177	10658	A3	2016	14995	Manual	39750	Petrol	30	57.6	1.4	Audi
99178	10659	A6	2018	27995	Semi-Auto	27500	Petrol	150	39.8	2.0	Audi
99179	10660	A4	2011	9995	Automatic	78000	Diesel	305	39.8	3.0	Audi

99180 rows × 11 columns

In [13]:
```
a.columns
```

Out[13]:
```
Index(['Unnamed: 0', 'model', 'year', 'price', 'transmission', 'mileage',
       'fuelType', 'tax', 'mpg', 'engineSize', 'Make'],
      dtype='object')
```

In [14]:

```
b=a[['Unnamed: 0', 'mileage', 'tax', 'mpg', 'engineSize']]
b
```

Out[14]:

	Unnamed: 0	mileage	tax	mpg	engineSize
0	0	13904	145	49.6	2.0
1	1	4562	145	49.6	2.0
2	2	7414	145	50.4	2.0
3	3	4825	145	32.5	2.0
4	4	6500	150	39.8	1.5
...	...	...	...	...	...
99175	10656	52500	0	176.6	1.4
99176	10657	23700	30	61.4	2.0
99177	10658	39750	30	57.6	1.4
99178	10659	27500	150	39.8	2.0
99179	10660	78000	305	39.8	3.0

99180 rows × 5 columns

In [15]:

```
c=b.iloc[:,0:11]
d=a.iloc[:,-1]
```

In [16]:

```
c.shape
```

Out[16]:

```
(99180, 5)
```

In [17]:

```
d.shape
```

Out[17]:

```
(99180,)
```

```
In [18]: from sklearn.preprocessing import StandardScaler
```

```
In [19]: fs=StandardScaler().fit_transform(c)
```

```
In [20]: logr=LogisticRegression()
         logr.fit(fs,d)
```

```
Out[20]: LogisticRegression()
```

```
In [21]: e=[[2,5,77,5,7]]
```

```
In [22]: prediction=logr.predict(e)
         prediction
```

```
Out[22]: array(['BMW'], dtype=object)
```

```
In [23]: logr.classes_
```

```
Out[23]: array(['Audi', 'BMW', 'VW', 'ford', 'hyundi', 'merc', 'skoda', 'toyota',
                'vauxhall'], dtype=object)
```

```
In [24]: logr.predict_proba(e)[0][0]
```

```
Out[24]: 1.251245706258745e-16
```

```
In [25]: logr.predict_proba(e)[0][1]
```

```
Out[25]: 0.9906163407167053
```

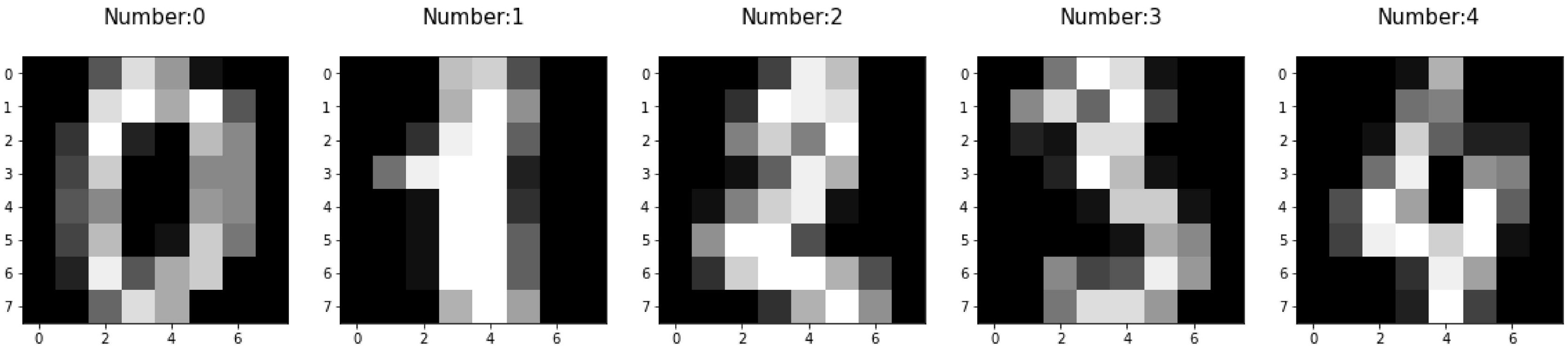
In [26]:
```
import re
from sklearn.datasets import load_digits
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

In [27]:
```
digits=load_digits()
digits
```

```
 'target': array([0, 1, 2, ..., 8, 9, 8]),
 'frame': None,
 'feature_names': ['pixel_0_0',
  'pixel_0_1',
  'pixel_0_2',
  'pixel_0_3',
  'pixel_0_4',
  'pixel_0_5',
  'pixel_0_6',
  'pixel_0_7',
  'pixel_1_0',
  'pixel_1_1',
  'pixel_1_2',
  'pixel_1_3',
  'pixel_1_4',
  'pixel_1_5',
  'pixel_1_6',
  'pixel_1_7',
  'pixel_2_0',
  'pixel_2_1'
```

In [28]:

```
plt.figure(figsize=(20,4))
for index,(image,label)in enumerate(zip(digits.data[0:5],digits.target[0:5])):
 plt.subplot(1,5,index+1)
 plt.imshow(np.reshape(image,(8,8)),cmap=plt.cm.gray)
 plt.title('Number:%i\n'%label,fontsize=15)
```



In [29]:

```
x_train,x_test,y_train,y_test=train_test_split(digits.data,digits.target,test_size=0.30)
```

In [30]:

```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(1257, 64)
(540, 64)
(1257,)
(540,)
```

In [31]:

```
logre=LogisticRegression(max_iter=10000)
logre.fit(x_train,y_train)
```

Out[31]:

```
LogisticRegression(max_iter=10000)
```

In [32]:

```
logre.predict(x_test)
```

Out[32]:

```
array([8, 4, 3, 1, 7, 4, 0, 9, 4, 7, 4, 9, 5, 4, 2, 6, 2, 5, 9, 7, 7, 5,
       7, 0, 3, 4, 9, 8, 6, 7, 1, 4, 1, 9, 1, 2, 2, 9, 9, 3, 4, 8, 9, 5,
       3, 0, 3, 0, 0, 4, 4, 9, 1, 2, 2, 7, 5, 9, 8, 2, 0, 4, 0, 2, 1, 4,
       0, 5, 8, 1, 8, 3, 5, 9, 3, 6, 7, 7, 0, 0, 5, 3, 5, 5, 6, 8, 8, 4,
       8, 8, 2, 5, 6, 5, 5, 8, 1, 6, 5, 6, 1, 3, 6, 6, 7, 6, 0, 7, 1, 6,
       9, 3, 0, 0, 4, 3, 9, 5, 3, 7, 2, 4, 7, 5, 6, 6, 0, 6, 6, 7, 1, 1,
       1, 0, 4, 8, 4, 9, 9, 1, 2, 1, 5, 0, 2, 6, 7, 5, 3, 4, 9, 2, 7, 5,
       1, 8, 5, 6, 0, 4, 1, 6, 2, 8, 9, 2, 1, 3, 7, 6, 5, 3, 4, 1, 3, 3,
       8, 7, 9, 6, 7, 9, 5, 4, 0, 2, 0, 5, 1, 1, 8, 9, 1, 7, 2, 4, 6, 7,
       4, 4, 2, 7, 9, 4, 7, 1, 3, 2, 7, 9, 7, 1, 9, 8, 2, 0, 1, 8, 6, 8,
       5, 1, 7, 8, 6, 0, 5, 0, 4, 9, 2, 6, 2, 8, 1, 3, 3, 6, 9, 4, 2, 4,
       5, 7, 4, 6, 5, 4, 6, 4, 0, 2, 4, 1, 9, 3, 7, 3, 0, 3, 9, 9, 6, 6,
       0, 1, 3, 2, 6, 2, 3, 3, 7, 8, 8, 4, 5, 5, 9, 6, 9, 0, 9, 2, 4, 9,
       4, 6, 2, 2, 1, 7, 0, 3, 5, 5, 2, 2, 4, 1, 6, 1, 8, 6, 5, 9, 7, 7,
       1, 8, 6, 7, 8, 8, 5, 9, 1, 5, 4, 4, 2, 5, 8, 4, 0, 7, 6, 2, 5, 6,
       9, 0, 6, 7, 7, 5, 0, 9, 6, 3, 1, 2, 8, 3, 3, 6, 5, 7, 7, 0, 8, 6,
       6, 3, 7, 8, 1, 0, 5, 9, 8, 4, 4, 6, 8, 3, 4, 6, 5, 7, 8, 4, 7, 3,
       2, 5, 1, 3, 0, 6, 7, 9, 8, 4, 9, 8, 1, 2, 8, 8, 6, 2, 0, 1, 5, 1,
       3, 3, 0, 5, 1, 3, 7, 1, 8, 9, 6, 0, 2, 3, 4, 3, 1, 8, 7, 1, 2, 8,
       1, 2, 0, 5, 5, 9, 2, 2, 9, 4, 8, 4, 4, 4, 2, 2, 9, 3, 7, 4, 5, 5,
       7, 8, 5, 8, 9, 0, 2, 6, 0, 5, 1, 8, 9, 6, 3, 4, 4, 3, 7, 8, 3, 6,
       5, 9, 5, 3, 6, 3, 0, 3, 8, 3, 2, 1, 6, 6, 8, 7, 2, 1, 6, 6, 5, 8,
       3, 6, 6, 3, 0, 3, 1, 3, 1, 9, 5, 1, 4, 0, 0, 0, 8, 7, 1, 0, 5, 2,
       5, 2, 8, 4, 6, 7, 0, 9, 3, 0, 0, 3, 9, 9, 0, 3, 0, 1, 0, 7, 7, 3,
       4, 7, 2, 6, 8, 1, 9, 1, 3, 8, 3, 8])
```

In [33]:

```
logre.score(x_test,y_test)
```

Out[33]:

```
0.9518518518518518
```

In [34]:

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

In [35]:

```
b=a.head(10)
b
```

Out[35]:

	Unnamed: 0	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize	Make
0	0	T-Roc	2019	25000	Automatic	13904	Diesel	145	49.6	2.0	VW
1	1	T-Roc	2019	26883	Automatic	4562	Diesel	145	49.6	2.0	VW
2	2	T-Roc	2019	20000	Manual	7414	Diesel	145	50.4	2.0	VW
3	3	T-Roc	2019	33492	Automatic	4825	Petrol	145	32.5	2.0	VW
4	4	T-Roc	2019	22900	Semi-Auto	6500	Petrol	150	39.8	1.5	VW
5	5	T-Roc	2020	31895	Manual	10	Petrol	145	42.2	1.5	VW
6	6	T-Roc	2020	27895	Manual	10	Petrol	145	42.2	1.5	VW
7	7	T-Roc	2020	39495	Semi-Auto	10	Petrol	145	32.5	2.0	VW
8	8	T-Roc	2019	21995	Manual	10	Petrol	145	44.1	1.0	VW
9	9	T-Roc	2019	23285	Manual	10	Petrol	145	42.2	1.5	VW

In [36]:

```
b=b[['Unnamed: 0', 'mileage', 'tax', 'mpg', 'engineSize','Make']]
b
```

Out[36]:

	Unnamed: 0	mileage	tax	mpg	engineSize	Make
0	0	13904	145	49.6	2.0	VW
1	1	4562	145	49.6	2.0	VW
2	2	7414	145	50.4	2.0	VW
3	3	4825	145	32.5	2.0	VW
4	4	6500	150	39.8	1.5	VW
5	5	10	145	42.2	1.5	VW
6	6	10	145	42.2	1.5	VW
7	7	10	145	32.5	2.0	VW
8	8	10	145	44.1	1.0	VW
9	9	10	145	42.2	1.5	VW

In [37]:

```
b['Make'].value_counts()
```

Out[37]:

```
VW    10
Name: Make, dtype: int64
```

In [38]:

```
x=b.drop('Make',axis=1)
y=b['Make']
```

In [39]:

```
g1={"Make":{'Make':1,'b':2}}
b=b.replace(g1)
print(b)
```

```
   Unnamed: 0  mileage  tax   mpg  engineSize Make
0           0    13904  145  49.6         2.0   VW
1           1     4562  145  49.6         2.0   VW
2           2     7414  145  50.4         2.0   VW
3           3     4825  145  32.5         2.0   VW
4           4     6500  150  39.8         1.5   VW
5           5       10  145  42.2         1.5   VW
6           6       10  145  42.2         1.5   VW
7           7       10  145  32.5         2.0   VW
8           8       10  145  44.1         1.0   VW
9           9       10  145  42.2         1.5   VW
```

In [40]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [41]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [42]:

```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[42]:

```
RandomForestClassifier()
```

In [43]:

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

In [44]:

```
from sklearn.model_selection import GridSearchCV
```

```
In [45]: grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
         grid_search.fit(x_train,y_train)
```

```
Out[45]: 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 [46]: grid_search.best_score_
```

```
Out[46]: 1.0
```

```
In [47]: rfc_best=grid_search.best_estimator_
```

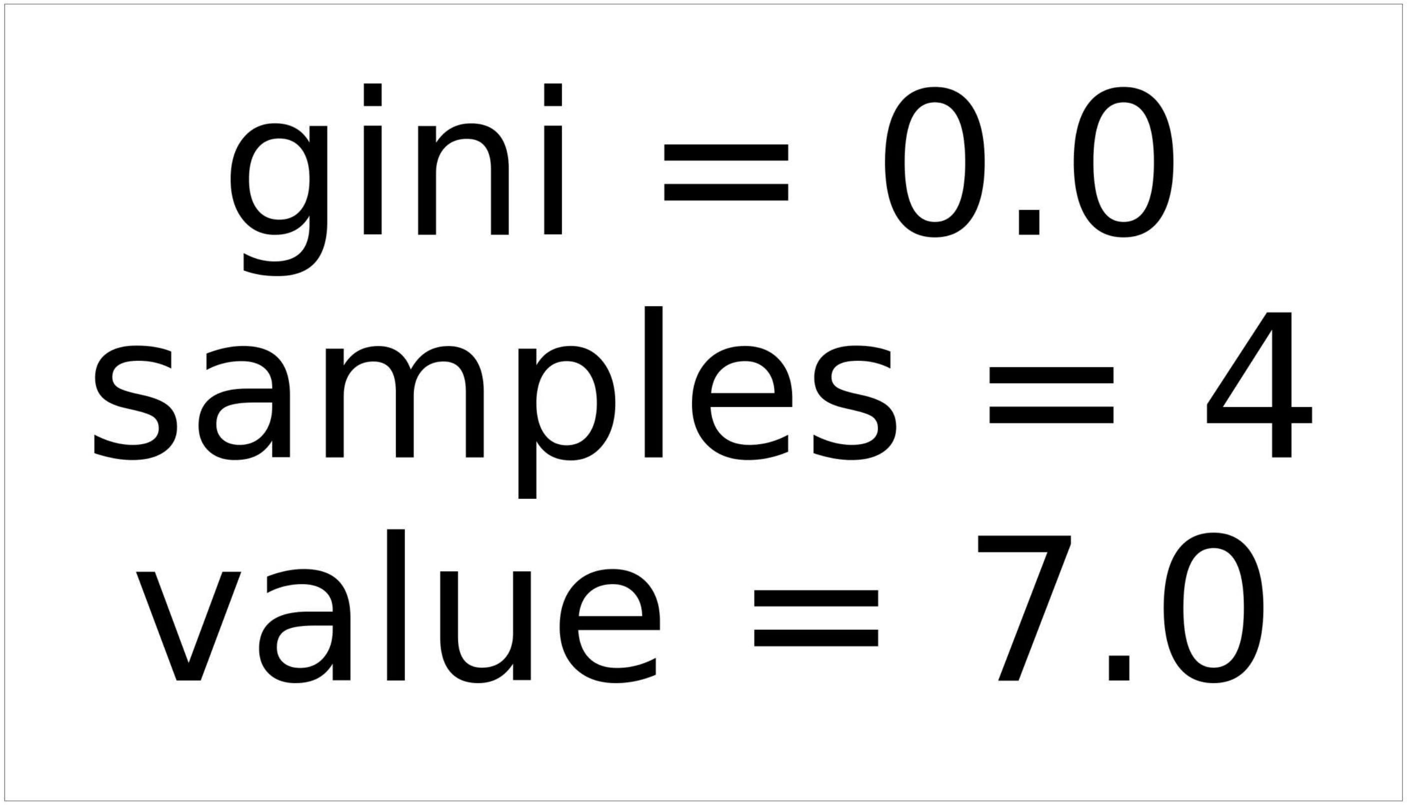
```
In [48]: from sklearn.tree import plot_tree
```

In [50]:

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],filled=True)
```

Out[50]:

```
[Text(2232.0, 1087.2, 'gini = 0.0\nsamples = 4\nvalue = 7.0')]
```



In [ ]: