In [5]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [6]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\c7_used_cars.csv")
df
```

Out[6]:

	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 [7]:
```
d=df[['Unnamed: 0','year','price','mileage','Make']]
d
```

Out[7]:

	Unnamed: 0	year	price	mileage	Make
0	0	2019	25000	13904	VW
1	1	2019	26883	4562	VW
2	2	2019	20000	7414	VW
3	3	2019	33492	4825	VW
4	4	2019	22900	6500	VW
...	...	...	...	...	...
99182	10663	2020	16999	4018	Audi
99183	10664	2020	16999	1978	Audi
99184	10665	2020	17199	609	Audi
99185	10666	2017	19499	8646	Audi
99186	10667	2016	15999	11855	Audi

99187 rows × 5 columns

In [8]:
```
ge=d.fillna(20)
ge
```

Out[8]:

	Unnamed: 0	year	price	mileage	Make
0	0	2019	25000	13904	VW
1	1	2019	26883	4562	VW
2	2	2019	20000	7414	VW
3	3	2019	33492	4825	VW
4	4	2019	22900	6500	VW
...	...	...	...	...	...
99182	10663	2020	16999	4018	Audi
99183	10664	2020	16999	1978	Audi
99184	10665	2020	17199	609	Audi
99185	10666	2017	19499	8646	Audi
99186	10667	2016	15999	11855	Audi

99187 rows × 5 columns

In [9]:
```
go=ge.head(100)
```

In [10]:
```
go['Make'].value_counts()
```

Out[10]:
```
VW    100
Name: Make, dtype: int64
```

In [11]:
```
x=d.drop('Make',axis=1)
y=d['Make']
```

In [12]:
```
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 [13]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[13]:
```
RandomForestClassifier()
```

In [14]:
```
paramets = {'max_depth':[1,2,3,4,5],
              'min_samples_leaf':[5,10,15,20,25],
              'n_estimators':[10,20,30,40,50]}
```

In [16]:
```
from sklearn.model_selection import GridSearchCV
grid_search= GridSearchCV(estimator = rfc,param_grid=paramets,cv=2,scoring="acc
grid_search.fit(x_train,y_train)
```

Out[16]:
```
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 [17]:
```
grid_search.best_score_
```

Out[17]:
```
0.3780642373613712
```

In [18]:
```
rfc_best=grid_search.best_estimator_
```

In [19]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','N
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-19-d9a5e60a8034> in <module>
      1 from sklearn.tree import plot_tree
      2 plt.figure(figsize=(80,40))
----> 3 plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names
=['Yes','No'],filled=True)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inn
er_f(*args, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in plot_tr
ee(decision_tree, max_depth, feature_names, class_names, label, filled, impur
ity, node_ids, proportion, rotate, rounded, precision, ax, fontsize)
    192         proportion=proportion, rotate=rotate, rounded=rounded,
    193         precision=precision, fontsize=fontsize)
--> 194     return exporter.export(decision_tree, ax=ax)
    195
    196

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in export
(self, decision_tree, ax)
    582         ax.clear()
    583         ax.set_axis_off()
--> 584         my_tree = self._make_tree(0, decision_tree.tree_,
    585                                   decision_tree.criterion)
    586         draw_tree = buchheim(my_tree)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in _make_t
ree(self, node_id, et, criterion, depth)
    563         # traverses _tree.Tree recursively, builds intermediate
    564         # "_reingold_tilford.Tree" object
--> 565         name = self.node_to_str(et, node_id, criterion=criterion)
    566         if (et.children_left[node_id] != _tree.TREE_LEAF
    567                 and (self.max_depth is None or depth <= self.max_dept
h)):

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in node_to
_str(self, tree, node_id, criterion)
    353                 node_string += 'class = '
    354             if self.class_names is not True:
--> 355                 class_name = self.class_names[np.argmax(value)]
    356             else:
    357                 class_name = "y%s%s%s" % (characters[1],

IndexError: list index out of range
```

In [ ]: