In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt,seaborn as sb
from sklearn.model_selection import  train_test_split
from sklearn.tree import DecisionTreeClassifier
```

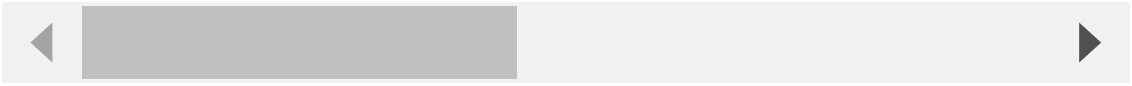
In [2]:

```
traindf=pd.read_csv(r"C:\Users\joel\Downloads\Mobile_Price_Classification_train.c
traindf
```

Out[2]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobi
0	842	0	2.2	0	1	0	7	0.6	
1	1021	1	0.5	1	0	1	53	0.7	
2	563	1	0.5	1	2	1	41	0.9	
3	615	1	2.5	0	0	0	10	0.8	
4	1821	1	1.2	0	13	1	44	0.6	
...	...	...	...	...	...	...	...	...	
1995	794	1	0.5	1	0	1	2	0.8	
1996	1965	1	2.6	1	0	0	39	0.2	
1997	1911	0	0.9	1	1	1	36	0.7	
1998	1512	0	0.9	0	4	1	46	0.1	
1999	510	1	2.0	1	5	1	45	0.9	

2000 rows × 21 columns



In [3]:

```
testdf=pd.read_csv(r"C:\Users\kunam\Downloads\Mobile_Price_Classification_test.csv
testdf
```

Out[3]:

	id	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep
0	1	1043	1	1.8	1	14	0	5	0.1
1	2	841	1	0.5	1	4	1	61	0.8
2	3	1807	1	2.8	0	1	0	27	0.9
3	4	1546	0	0.5	1	18	1	25	0.5
4	5	1434	0	1.4	0	11	1	49	0.5
...	...	...	...	...	...	...	...	...	...
995	996	1700	1	1.9	0	0	1	54	0.5
996	997	609	0	1.8	1	0	0	13	0.9
997	998	1185	0	1.4	0	1	1	8	0.5
998	999	1533	1	0.5	1	0	0	50	0.4
999	1000	1270	1	0.5	0	4	1	35	0.1

1000 rows × 21 columns

In [4]:

```
1 traindf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   battery_power  2000 non-null   int64  
 1   blue           2000 non-null   int64  
 2   clock_speed    2000 non-null   float64
 3   dual_sim       2000 non-null   int64  
 4   fc             2000 non-null   int64  
 5   four_g         2000 non-null   int64  
 6   int_memory     2000 non-null   int64  
 7   m_dep          2000 non-null   float64
 8   mobile_wt      2000 non-null   int64  
 9   n_cores        2000 non-null   int64  
 10  pc             2000 non-null   int64  
 11  px_height      2000 non-null   int64  
 12  px_width       2000 non-null   int64  
 13  ram            2000 non-null   int64  
 14  sc_h           2000 non-null   int64  
 15  sc_w           2000 non-null   int64  
 16  talk_time      2000 non-null   int64  
 17  three_g        2000 non-null   int64  
 18  touch_screen   2000 non-null   int64  
 19  wifi           2000 non-null   int64  
 20  price_range    2000 non-null   int64  
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```

In [5]:

```
1 testdf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   id             1000 non-null   int64  
 1   battery_power  1000 non-null   int64  
 2   blue           1000 non-null   int64  
 3   clock_speed    1000 non-null   float64
 4   dual_sim       1000 non-null   int64  
 5   fc             1000 non-null   int64  
 6   four_g         1000 non-null   int64  
 7   int_memory     1000 non-null   int64  
 8   m_dep          1000 non-null   float64
 9   mobile_wt      1000 non-null   int64  
 10  n_cores        1000 non-null   int64  
 11  pc             1000 non-null   int64  
 12  px_height      1000 non-null   int64  
 13  px_width       1000 non-null   int64  
 14  ram            1000 non-null   int64  
 15  sc_h           1000 non-null   int64  
 16  sc_w           1000 non-null   int64  
 17  talk_time      1000 non-null   int64  
 18  three_g        1000 non-null   int64  
 19  touch_screen   1000 non-null   int64  
 20  wifi           1000 non-null   int64  
dtypes: float64(2), int64(19)
memory usage: 164.2 KB
```

In [6]:

```
1 traindf.shape
```

Out[6]:

```
(2000, 21)
```

In [7]:

```
1 testdf.shape
```

Out[7]:

```
(1000, 21)
```

In [8]:

```
1 traindf=traindf.head(1000)
```

In [18]:

```
x=testdf
y=traindf['price_range']
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.7,random_state=42
```

In [19]:

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[19]:

```
RandomForestClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [20]:

```
params={'max_depth':[2,3,5,10,20],'min_samples_leaf':[5,10,20,50,100,200],'n_esti
```

In [21]:

```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=params,cv=2,scoring="accuracy")
```

In [22]:

```
grid_search.fit(x_train,y_train)
```

Out[22]:

```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [2, 3, 5, 10, 20],
                         'min_samples_leaf': [5, 10, 20, 50, 100, 200],
                         'n_estimators': [10, 25, 30, 50, 100, 200]},
             scoring='accuracy')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [23]:

```
grid_search.best_score_
```

Out[23]:

```
0.2885714285714286
```

In [24]:

```
rf_best=grid_search.best_estimator_
rf_best
```

Out[24]:

```
RandomForestClassifier(max_depth=20, min_samples_leaf=20, n_estimators=
10)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [25]:

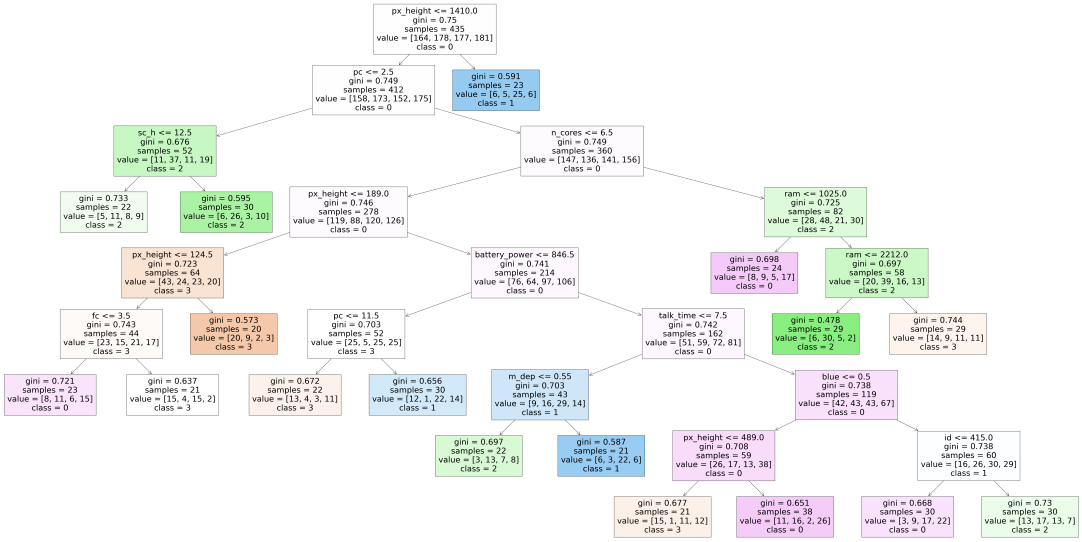
```
traindf['price_range'].value_counts()
```

Out[25]:

```
price_range
3    276
2    248
0    242
1    234
Name: count, dtype: int64
```

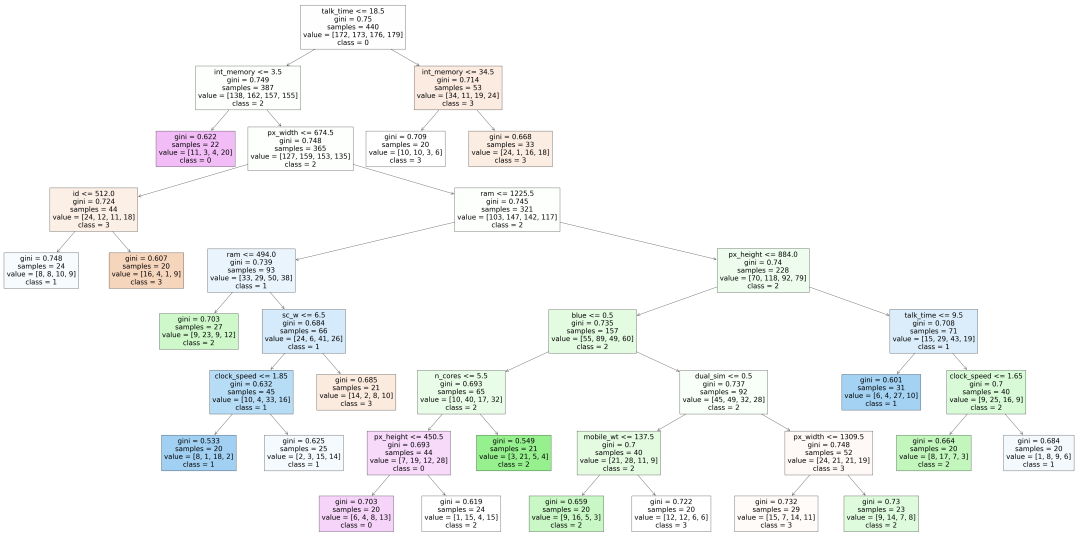
In [27]:

```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[4],feature_names=x.columns,class_names=['3','2','1'
```



In [29]:

```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[5],feature_names=x.columns,class_names=['3','2','1'
```



In [30]:

```
rf_best.feature_importances_
```

Out[30]:

```
array([0.07270562, 0.05216245, 0.01970191, 0.01964258, 0.01827917,
       0.01970525, 0.00435999, 0.07418107, 0.04931824, 0.07133815,
       0.07048471, 0.07348234, 0.11411824, 0.06122906, 0.10886214,
       0.02680924, 0.06651904, 0.0488326 , 0.00514388, 0.00557765,
       0.01754668])
```

In [32]:

```
imp_df=pd.DataFrame({"Varname":x_train.columns,"Imp":rf_best.feature_importances_
```

In [33]:

```
imp_df.sort_values(by="Imp",ascending=False)
```

Out[33]:

	Varname	Imp
12	px_height	0.114118
14	ram	0.108862
7	int_memory	0.074181
11	pc	0.073482
0	id	0.072706
9	mobile_wt	0.071338
10	n_cores	0.070485
16	sc_w	0.066519
13	px_width	0.061229
1	battery_power	0.052162
8	m_dep	0.049318
17	talk_time	0.048833
15	sc_h	0.026809
5	fc	0.019705
2	blue	0.019702
3	clock_speed	0.019643
4	dual_sim	0.018279
20	wifi	0.017547
19	touch_screen	0.005578
18	three_g	0.005144
6	four_g	0.004360

In [ ]:

```
1
```