RANDOM FOREST FOR MOBILE

In [6]:

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

In [7]:

```
test_df=pd.read_csv(r"C:\Users\RAMADEVI SURIPAKA\Downloads\Mobile_Price_Classification_test.csv
test_df
```

Out[7]:

blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	...	pc	px_height	px_width
1	1.8	1	14	0	5	0.1	193	...	16	226	1412
1	0.5	1	4	1	61	0.8	191	...	12	746	857
1	2.8	0	1	0	27	0.9	186	...	4	1270	1366
0	0.5	1	18	1	25	0.5	96	...	20	295	1752
0	1.4	0	11	1	49	0.5	108	...	18	749	810
...	...	...	...	...	...	...	...	...	...	...	...
1	1.9	0	0	1	54	0.5	170	...	17	644	913
0	1.8	1	0	0	13	0.9	186	...	2	1152	1632
0	1.4	0	1	1	8	0.5	80	...	12	477	825
1	0.5	1	0	0	50	0.4	171	...	12	38	832
1	0.5	0	4	1	35	0.1	140	...	19	457	608

In [38]:

```
train_df=pd.read_csv(r"C:\Users\RAMADEVI SURIPAKA\Downloads\Mobile_Price_Classification_train.
train_df
```

Out[38]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cc
0	842	0	2.2	0	1	0	7	0.6	188	
1	1021	1	0.5	1	0	1	53	0.7	136	
2	563	1	0.5	1	2	1	41	0.9	145	
3	615	1	2.5	0	0	0	10	0.8	131	
4	1821	1	1.2	0	13	1	44	0.6	141	
...	...	...	...	...	...	...	...	...	...	
1995	794	1	0.5	1	0	1	2	0.8	106	
1996	1965	1	2.6	1	0	0	39	0.2	187	
1997	1911	0	0.9	1	1	1	36	0.7	108	
1998	1512	0	0.9	0	4	1	46	0.1	145	
1999	510	1	2.0	1	5	1	45	0.9	168	

2000 rows × 21 columns

In [9]:

```
train_df.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 [10]:

```
test_df.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 [11]:

```
x=train_df.drop('ram',axis=1)
y=train_df['ram']
```

In [12]:

```
x=test_df.drop('ram',axis=1)
y=test_df['ram']
```

In [13]:

```
train_df['dual_sim'].value_counts()
```

Out[13]:

```
dual_sim
1    1019
0     981
Name: count, dtype: int64
```

In [14]:

```
test_df['dual_sim'].value_counts()
```

Out[14]:

```
dual_sim
1    517
0    483
Name: count, dtype: int64
```

In [39]:

```
T={"three_g":{'Yes':1,'No':0}}
train_df=train_df.replace(T)
train_df
```

Out[39]:

dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram	sc_h
0	1	0	7	0.6	188	2	...	20	756	2549	9
1	0	1	53	0.7	136	3	...	905	1988	2631	17
1	2	1	41	0.9	145	5	...	1263	1716	2603	11
0	0	0	10	0.8	131	6	...	1216	1786	2769	16
0	13	1	44	0.6	141	2	...	1208	1212	1411	8
...	...	...	...	...	...	...	...	...	...	...	...
1	0	1	2	0.8	106	6	...	1222	1890	668	13
1	0	0	39	0.2	187	4	...	915	1965	2032	11
1	1	1	36	0.7	108	8	...	868	1632	3057	9
0	4	1	46	0.1	145	5	...	336	670	869	18
1	5	1	45	0.9	168	6	...	483	754	3919	19

In [40]:

```
T={"three_g":{'Yes':1,'No':0}}
test_df=test_df.replace(T)
test_df
```

Out[40]:

blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	...	pc	px_height	px_width
1	1.8	1	14	0	5	0.1	193	...	16	226	1412
1	0.5	1	4	1	61	0.8	191	...	12	746	857
1	2.8	0	1	0	27	0.9	186	...	4	1270	1366
0	0.5	1	18	1	25	0.5	96	...	20	295	1752
0	1.4	0	11	1	49	0.5	108	...	18	749	810
...	...	...	...	...	...	...	...	...	...	...	...
1	1.9	0	0	1	54	0.5	170	...	17	644	913
0	1.8	1	0	0	13	0.9	186	...	2	1152	1632
0	1.4	0	1	1	8	0.5	80	...	12	477	825
1	0.5	1	0	0	50	0.4	171	...	12	38	832
1	0.5	0	4	1	35	0.1	140	...	19	457	608

In [91]:

```
x=train_df.drop('touch_screen',axis=1)
y=train_df['touch_screen']
train_df
```

Out[91]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cc
0	842	0	2.2	0	1	0	7	0.6	188	
1	1021	1	0.5	1	0	1	53	0.7	136	
2	563	1	0.5	1	2	1	41	0.9	145	
3	615	1	2.5	0	0	0	10	0.8	131	
4	1821	1	1.2	0	13	1	44	0.6	141	
...	...	...	...	...	...	...	...	...	...	
1995	794	1	0.5	1	0	1	2	0.8	106	
1996	1965	1	2.6	1	0	0	39	0.2	187	
1997	1911	0	0.9	1	1	1	36	0.7	108	
1998	1512	0	0.9	0	4	1	46	0.1	145	
1999	510	1	2.0	1	5	1	45	0.9	168	

2000 rows × 21 columns

In [93]:

```
x=test_df.drop('touch_screen',axis=1)
y=test_df['touch_screen']
```

In [94]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.7,random_state=42)
x_train.shape,x_test.shape
```

Out[94]:

```
((700, 20), (300, 20))
```

In [95]:

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

Out[95]:

```
RandomForestClassifier
RandomForestClassifier()
```

In [96]:

```
rf=RandomForestClassifier()
```

In [97]:

```
params={'max_depth':[2,3,5,10,20],
       'min_samples_leaf':[5,10,20,50,100,200],
       'n_estimators':[10,25,30,50,100,200]}
```

In [98]:

```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rf,param_grid=params,cv=2,scoring='accuracy')
grid_search.fit(x_train,y_train)
```

Out[98]:

```
GridSearchCV
estimator: RandomForestClassifier
RandomForestClassifier
```

In [99]:

```
grid_search.best_score_
```

Out[99]:

```
0.5171428571428571
```

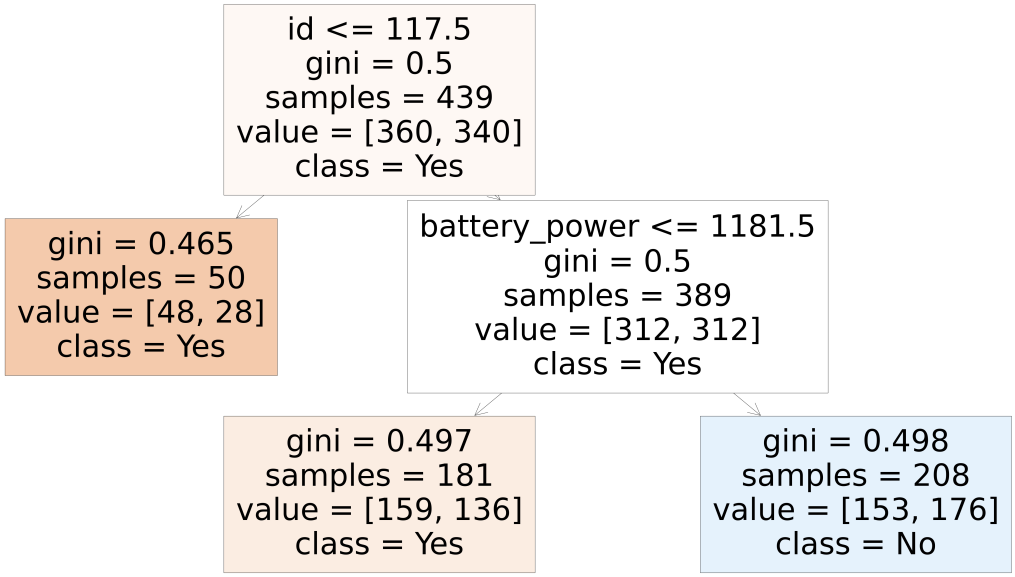
In [100]:

```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=2, min_samples_leaf=50, n_estimators=10)
```

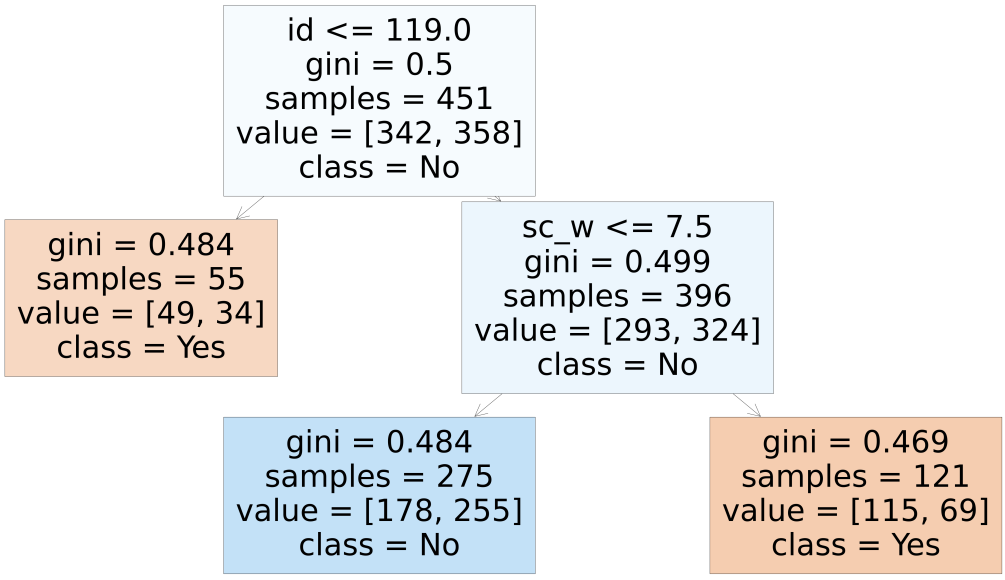
In [101]:

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



In [102]:

```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[7],feature_names=x.columns,class_names=['Yes','No'],filled=True)
```



In [106]:

```
rf_best.feature_importances_
```

Out[106]:

```
array([0.11987777, 0.07440228, 0.        , 0.        , 0.        ,
       0.1303988 , 0.00580689, 0.03909404, 0.        , 0.02121682,
       0.        , 0.06259255, 0.09687229, 0.07520894, 0.13973031,
       0.01461708, 0.08582806, 0.13435419, 0.        , 0.        ])
```

In [107]:

```
imp_df=pd.DataFrame({'Varname':x_train.columns,"Imp":rf_best.feature_importances_})
imp_df.sort_values(by="Imp",ascending=False)
```

Out[107]:

	Varname	Imp
14	ram	0.139730
17	talk_time	0.134354
5	fc	0.130399
0	id	0.119878
12	px_height	0.096872
16	sc_w	0.085828
13	px_width	0.075209
1	battery_power	0.074402
11	pc	0.062593
7	int_memory	0.039094
9	mobile_wt	0.021217
15	sc_h	0.014617
6	four_g	0.005807
18	three_g	0.000000
10	n_cores	0.000000
8	m_dep	0.000000
4	dual_sim	0.000000
3	clock_speed	0.000000
2	blue	0.000000
19	wifi	0.000000

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

In [ ]:

In [ ]:

In [ ]: