In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt,seaborn as sns
```

In [2]:
```
df=pd.read_csv(r"C:\Users\pucha\Downloads\Mobile_Price_Classification_train.csv")
df
```

Out[2]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram	sc_h	s
0	842	0	2.2	0	1	0	7	0.6	188	2	...	20	756	2549	9	
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	905	1988	2631	17	
2	563	1	0.5	1	2	1	41	0.9	145	5	...	1263	1716	2603	11	
3	615	1	2.5	0	0	0	10	0.8	131	6	...	1216	1786	2769	16	
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	1208	1212	1411	8	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1995	794	1	0.5	1	0	1	2	0.8	106	6	...	1222	1890	668	13	
1996	1965	1	2.6	1	0	0	39	0.2	187	4	...	915	1965	2032	11	
1997	1911	0	0.9	1	1	1	36	0.7	108	8	...	868	1632	3057	9	
1998	1512	0	0.9	0	4	1	46	0.1	145	5	...	336	670	869	18	
1999	510	1	2.0	1	5	1	45	0.9	168	6	...	483	754	3919	19	

2000 rows × 21 columns

In [3]:
```
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 [5]:
```
x=df.drop('blue',axis=1)
y=df['blue']
```

In [6]:
```
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[6]:
```
((1400, 20), (600, 20))
```

In [7]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[7]:
```
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 [8]:
```
rf=RandomForestClassifier()
```

In [9]:
```
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 [11]:
```
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[11]:
```
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.

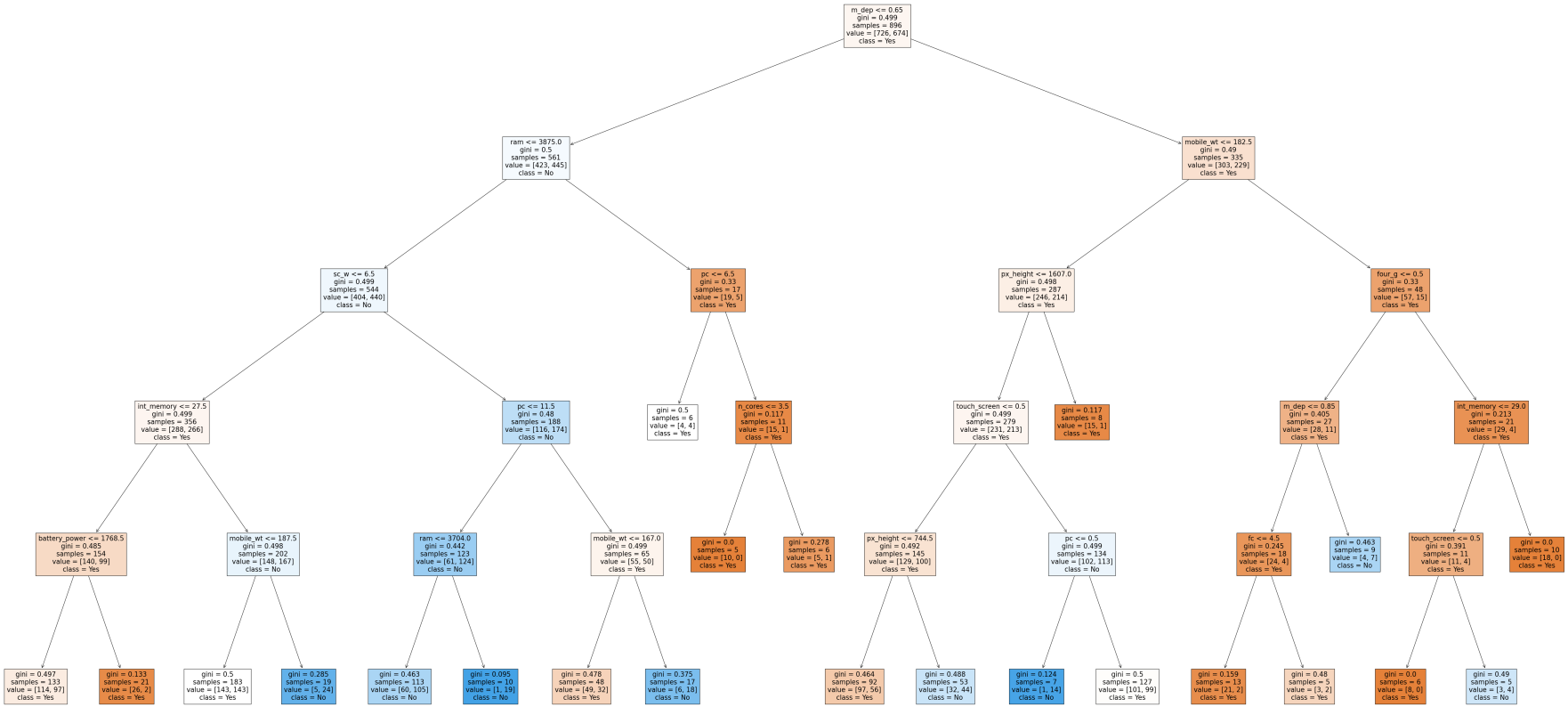
In [12]:
```
grid_search.best_score_
```

Out[12]:
```
0.5321428571428571
```

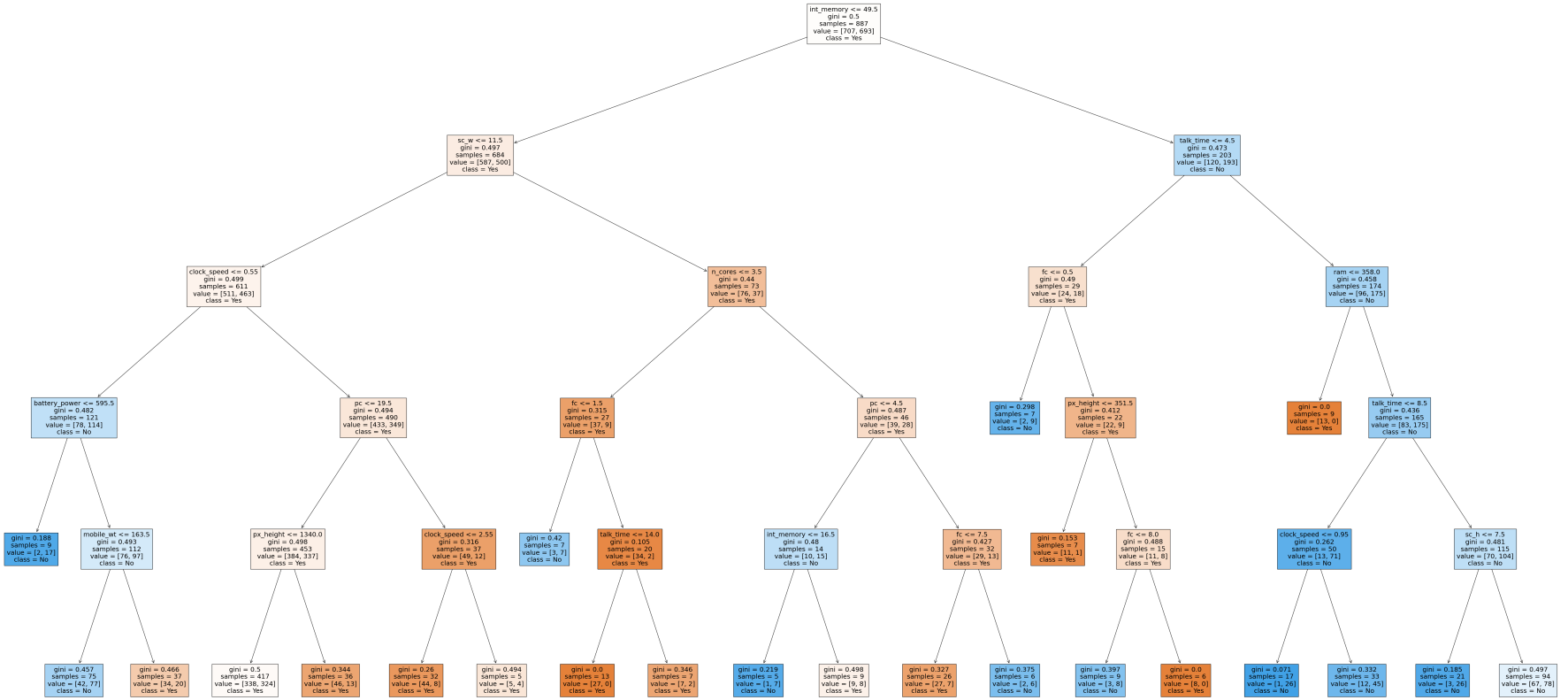
In [13]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=5, min_samples_leaf=5, n_estimators=50)
```

In [14]:
```
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 [15]:
```
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 [16]:
```
rf_best.feature_importances_
```

Out[16]:
```
array([0.09104402, 0.06309331, 0.01562555, 0.04032672, 0.01253739,
       0.0764571 , 0.05188269, 0.07685584, 0.03277826, 0.05444615,
       0.09210705, 0.0980117 , 0.10140007, 0.03438789, 0.06328036,
       0.05263394, 0.00893529, 0.0099836 , 0.00687958, 0.01733349])
```

In [17]:
```
imp_df=pd.DataFrame({"varname":x_train.columns,"Imp":rf_best.feature_importances_})
```

In [18]:
```
imp_df.sort_values(by="Imp",ascending=False)
```

Out[18]:

	varname	Imp
12	ram	0.101400
11	px_width	0.098012
10	px_height	0.092107
0	battery_power	0.091044
7	mobile_wt	0.076856
5	int_memory	0.076457
14	sc_w	0.063280
1	clock_speed	0.063093
9	pc	0.054446
15	talk_time	0.052634
6	m_dep	0.051883
3	fc	0.040327
13	sc_h	0.034388
8	n_cores	0.032778
19	price_range	0.017333
2	dual_sim	0.015626
4	four_g	0.012537
17	touch_screen	0.009984
16	three_g	0.008935
18	wifi	0.006880

In [ ]: