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

In [2]:
```
s=pd.read_csv(r"C:\Users\DELL E5490\Downloads\Mobile_Price_Classification_test.csv")
s
```

Out[2]:
```
       id  battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  ...  pc  px_height  px_width   ram  sc_h  sc
  0     1           1043     1          1.8         1  14       0           5    0.1        193  ...  16        226      1412  3476    12
  1     2            841     1          0.5         1   4       1          61    0.8        191  ...  12        746       857  3895     6
  2     3           1807     1          2.8         0   1       0          27    0.9        186  ...   4       1270      1366  2396    17
  3     4           1546     0          0.5         1  18       1          25    0.5         96  ...  20        295      1752  3893    10
  4     5           1434     0          1.4         0  11       1          49    0.5        108  ...  18        749       810  1773    15
...   ...            ...   ...          ...       ...  ...    ...         ...    ...        ...  ...  ...       ...       ...   ...   ...
995   996           1700     1          1.9         0   0       1          54    0.5        170  ...  17        644       913  2121    14
996   997            609     0          1.8         1   0       0          13    0.9        186  ...   2       1152      1632  1933     8
997   998           1185     0          1.4         0   1       1           8    0.5         80  ...  12        477       825  1223     5
998   999           1533     1          0.5         1   0       0          50    0.4        171  ...  12         38       832  2509    15
999  1000           1270     1          0.5         0   4       1          35    0.1        140  ...  19        457       608  2828     9

1000 rows × 21 columns
```

In [3]:
```
s.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 [4]:
```
x=s.drop('wifi',axis=1)
y=s['wifi']
```

In [5]:

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

Out[5]:

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

In [6]:

```
m={"three_g":{"Yes":1,"No":0}}
s=s.replace(m)
print(s)
```

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

     m_dep  mobile_wt  ...  pc  px_height  px_width   ram  sc_h  sc_w  \
0      0.1        193  ...  16        226      1412  3476    12     7
1      0.8        191  ...  12        746       857  3895     6     0
2      0.9        186  ...   4       1270      1366  2396    17    10
3      0.5         96  ...  20        295      1752  3893    10     0
4      0.5        108  ...  18        749       810  1773    15     8
..     ...        ...  ...  ..        ...       ...   ...   ...   ...
995    0.5        170  ...  17        644       913  2121    14     8
996    0.9        186  ...   2       1152      1632  1933     8     1
997    0.5         80  ...  12        477       825  1223     5     0
998    0.4        171  ...  12         38       832  2509    15    11
999    0.1        140  ...  19        457       608  2828     9     2

     talk_time  three_g  touch_screen  wifi
0            2        0             1     0
1            7        1             0     0
2           10        0             1     1
3            7        1             1     0
4            7        1             0     1
..         ...      ...           ...   ...
995         15        1             1     0
996         19        0             1     1
997         14        1             0     0
998          6        0             1     0
999          3        1             0     1

[1000 rows x 21 columns]
```

In [7]:
```
x=s.drop('wifi',axis=1)
y=s['wifi']
```

In [8]:
```
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[8]:
```
((700, 20), (300, 20))
```

In [9]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[9]:
```
▾ RandomForestClassifier

RandomForestClassifier()
```

In [10]:
```
rf=RandomForestClassifier()
```

In [11]:
```
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 [12]:
```
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[12]:
```
▸          GridSearchCV
▸ estimator: RandomForestClassifier
      ▸ RandomForestClassifier
```

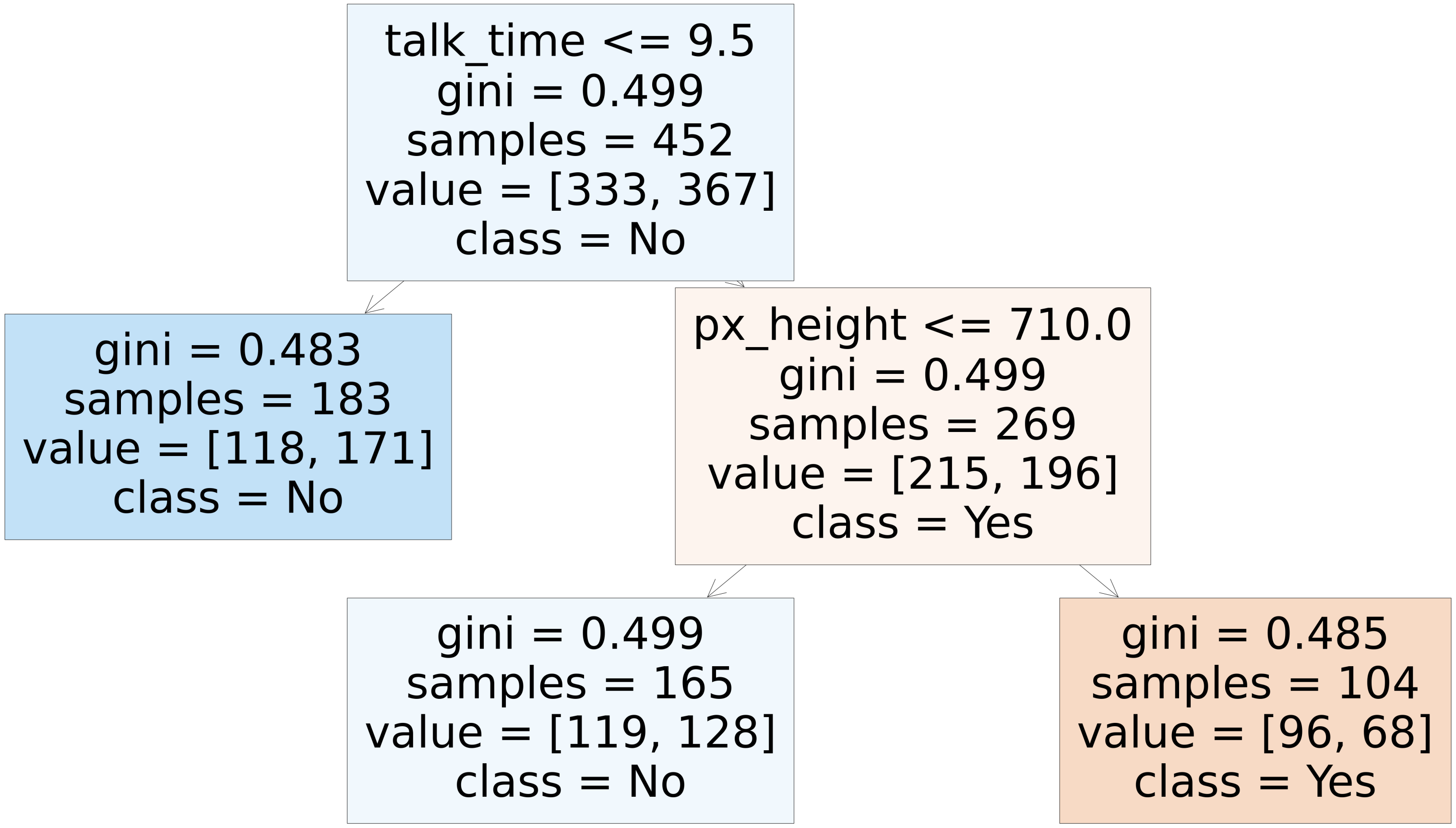
In [13]:
```
grid_search.best_score_
```

Out[13]:
```
0.5571428571428572
```

In [14]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=2, min_samples_leaf=100, n_estimators=25)
```

In [16]:
```
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 [17]:
```
rf_best.feature_importances_
```

Out[17]:
```
array([0.02594591, 0.03366646, 0.00431089, 0.14117196, 0.        ,
       0.08205576, 0.03873293, 0.08617024, 0.04401796, 0.14204191,
       0.        , 0.03652785, 0.01284107, 0.15935068, 0.05934018,
       0.01058171, 0.02032306, 0.07430056, 0.01419332, 0.01442756])
```

In [18]:

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

Out[18]:

	Varname	IMP
13	px_width	0.159351
9	mobile_wt	0.142042
3	clock_speed	0.141172
7	int_memory	0.086170
5	fc	0.082056
17	talk_time	0.074301
14	ram	0.059340
8	m_dep	0.044018
6	four_g	0.038733
11	pc	0.036528
1	battery_power	0.033666
0	id	0.025946
16	sc_w	0.020323
19	touch_screen	0.014428
18	three_g	0.014193
12	px_height	0.012841
15	sc_h	0.010582
2	blue	0.004311
4	dual_sim	0.000000
10	n_cores	0.000000

In [ ]: