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

In [2]:
```
train_df=pd.read_csv(r"C:\Users\Lenovo\OneDrive\Desktop\Data Sets\Mobile_Price_Classification_tes
train_df
```

Out[2]:

	id	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	...	pc	px_height
0	1	1043	1	1.8	1	14	0	5	0.1	193	...	16	226
1	2	841	1	0.5	1	4	1	61	0.8	191	...	12	746
2	3	1807	1	2.8	0	1	0	27	0.9	186	...	4	1270
3	4	1546	0	0.5	1	18	1	25	0.5	96	...	20	295
4	5	1434	0	1.4	0	11	1	49	0.5	108	...	18	749
...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	996	1700	1	1.9	0	0	1	54	0.5	170	...	17	644
996	997	609	0	1.8	1	0	0	13	0.9	186	...	2	1152
997	998	1185	0	1.4	0	1	1	8	0.5	80	...	12	477
998	999	1533	1	0.5	1	0	0	50	0.4	171	...	12	38
999	1000	1270	1	0.5	0	4	1	35	0.1	140	...	19	457

1000 rows × 21 columns

In [3]:
```
test_df=pd.read_csv(r"C:\Users\Lenovo\OneDrive\Desktop\Data Sets\Mobile_Price_Classification_trai
test_df
```

Out[3]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height
0	842	0	2.2	0	1	0	7	0.6	188	2	...	20
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	905
2	563	1	0.5	1	2	1	41	0.9	145	5	...	1263
3	615	1	2.5	0	0	0	10	0.8	131	6	...	1216
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	1208
...	...	...	...	...	...	...	...	...	...	...	...	...
1995	794	1	0.5	1	0	1	2	0.8	106	6	...	1222
1996	1965	1	2.6	1	0	0	39	0.2	187	4	...	915
1997	1911	0	0.9	1	1	1	36	0.7	108	8	...	868
1998	1512	0	0.9	0	4	1	46	0.1	145	5	...	336
1999	510	1	2.0	1	5	1	45	0.9	168	6	...	483

2000 rows × 21 columns

In [4]:
```
train_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 [5]:
```
test_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: 328.3 KB
```

In [7]:

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

In [9]:

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

In [10]:

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

Out[10]:

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

In [11]:

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

Out[11]:

```
blue
0    1010
1     990
Name: count, dtype: int64
```

In [12]:

```
T={"Home Owner":{"Yes":1,"No":0}}
train_df=train_df.replace(T)
print(train_df)
```

```
       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 [13]:
```
T={"Home Owner":{"Yes":1,"No":0}}
train_df=train_df.replace(T)
print(train_df)
```

```
       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 [14]:
```
T={"Home Owner":{"Yes":1,"No":0}}
test_df=test_df.replace(T)
print(test_df)
```

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

      m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  \
0       0.6        188        2  ...         20       756  2549     9     7   
1       0.7        136        3  ...        905      1988  2631    17     3   
2       0.9        145        5  ...       1263      1716  2603    11     2   
3       0.8        131        6  ...       1216      1786  2769    16     8   
4       0.6        141        2  ...       1208      1212  1411     8     2   
...     ...        ...      ...  ...        ...       ...   ...   ...   ...   
1995    0.8        106        6  ...       1222      1890   668    13     4   
1996    0.2        187        4  ...        915      1965  2032    11    10   
1997    0.7        108        8  ...        868      1632  3057     9     1   
1998    0.1        145        5  ...        336       670   869    18    10   
1999    0.9        168        6  ...        483       754  3919    19     4   

      talk_time  three_g  touch_screen  wifi  price_range  
0            19        0             0     1            1  
1             7        1             1     0            2  
2             9        1             1     0            2  
3            11        1             0     0            2  
4            15        1             1     0            1  
...         ...      ...           ...   ...          ...  
1995         19        1             1     0            0  
1996         16        1             1     1            2  
1997          5        1             1     0            3  
1998         19        1             1     1            0  
1999          2        1             1     1            3  

[2000 rows x 21 columns]
```

In [15]:
```
x=train_df.drop('wifi',axis=1)
y=train_df['wifi']
```

In [16]:
```
x=test_df.drop('wifi',axis=1)
y=test_df['wifi']
```

In [17]:
```
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[17]:
```
((1400, 20), (600, 20))
```

In [18]:

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

Out[18]:

```
▾ RandomForestClassifier

RandomForestClassifier()
```

In [19]:

```
rf = RandomForestClassifier()
```

In [20]:

```
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 [21]:

```
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[21]:

```
▸              GridSearchCV
▸ estimator: RandomForestClassifier
       ▸ RandomForestClassifier
```

In [22]:

```
grid_search.best_score_
```

Out[22]:

```
0.5285714285714285
```

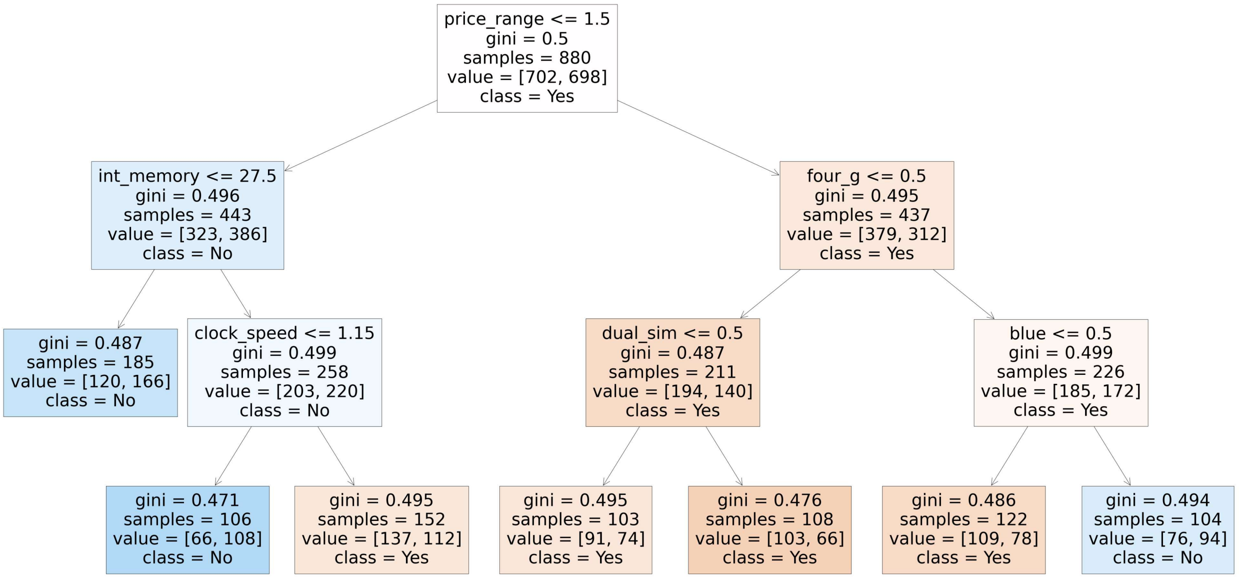
In [23]:

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

```
RandomForestClassifier(max_depth=3, min_samples_leaf=100, n_estimators=10)
```

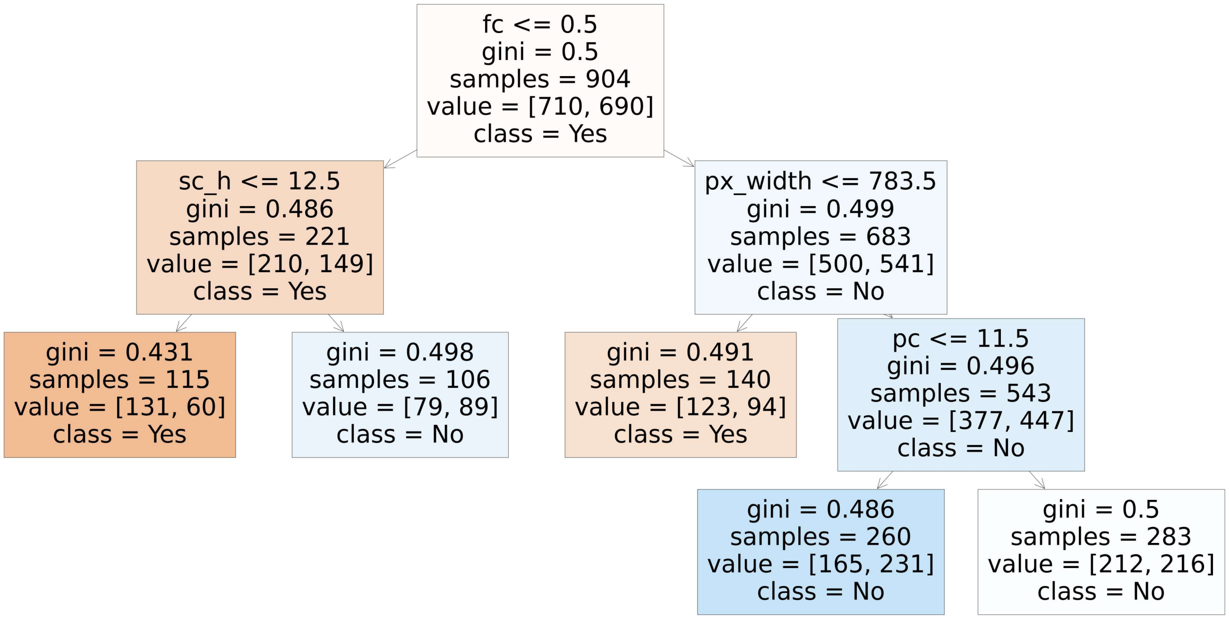
In [24]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[5], feature_names = x.columns,class_names=['Yes',"No"],filled=True
```

Out[24]:
```
[Text(0.4230769230769231, 0.875, 'price_range <= 1.5\ngini = 0.5\nsamples = 880\nvalue = [702,
698]\nclass = Yes'),
 Text(0.15384615384615385, 0.625, 'int_memory <= 27.5\ngini = 0.496\nsamples = 443\nvalue = [32
3, 386]\nclass = No'),
 Text(0.07692307692307693, 0.375, 'gini = 0.487\nsamples = 185\nvalue = [120, 166]\nclass = N
o'),
 Text(0.23076923076923078, 0.375, 'clock_speed <= 1.15\ngini = 0.499\nsamples = 258\nvalue = [2
03, 220]\nclass = No'),
 Text(0.15384615384615385, 0.125, 'gini = 0.471\nsamples = 106\nvalue = [66, 108]\nclass = N
o'),
 Text(0.3076923076923077, 0.125, 'gini = 0.495\nsamples = 152\nvalue = [137, 112]\nclass = Ye
s'),
 Text(0.6923076923076923, 0.625, 'four_g <= 0.5\ngini = 0.495\nsamples = 437\nvalue = [379, 31
2]\nclass = Yes'),
 Text(0.5384615384615384, 0.375, 'dual_sim <= 0.5\ngini = 0.487\nsamples = 211\nvalue = [194, 1
40]\nclass = Yes'),
 Text(0.46153846153846156, 0.125, 'gini = 0.495\nsamples = 103\nvalue = [91, 74]\nclass = Ye
s'),
 Text(0.6153846153846154, 0.125, 'gini = 0.476\nsamples = 108\nvalue = [103, 66]\nclass = Ye
s'),
 Text(0.8461538461538461, 0.375, 'blue <= 0.5\ngini = 0.499\nsamples = 226\nvalue = [185, 172]
\nclass = Yes'),
 Text(0.7692307692307693, 0.125, 'gini = 0.486\nsamples = 122\nvalue = [109, 78]\nclass = Ye
s'),
 Text(0.9230769230769231, 0.125, 'gini = 0.494\nsamples = 104\nvalue = [76, 94]\nclass = No')]
```



In [25]:
```
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)
```

Out[25]:
```
[Text(0.4444444444444444, 0.875, 'fc <= 0.5\ngini = 0.5\nsamples = 904\nvalue = [710, 690]\ncla
ss = Yes'),
 Text(0.2222222222222222, 0.625, 'sc_h <= 12.5\ngini = 0.486\nsamples = 221\nvalue = [210, 149]
\nclass = Yes'),
 Text(0.1111111111111111, 0.375, 'gini = 0.431\nsamples = 115\nvalue = [131, 60]\nclass = Ye
s'),
 Text(0.3333333333333333, 0.375, 'gini = 0.498\nsamples = 106\nvalue = [79, 89]\nclass = No'),
 Text(0.6666666666666666, 0.625, 'px_width <= 783.5\ngini = 0.499\nsamples = 683\nvalue = [500,
541]\nclass = No'),
 Text(0.5555555555555556, 0.375, 'gini = 0.491\nsamples = 140\nvalue = [123, 94]\nclass = Ye
s'),
 Text(0.7777777777777778, 0.375, 'pc <= 11.5\ngini = 0.496\nsamples = 543\nvalue = [377, 447]\n
class = No'),
 Text(0.6666666666666666, 0.125, 'gini = 0.486\nsamples = 260\nvalue = [165, 231]\nclass = N
o'),
 Text(0.8888888888888888, 0.125, 'gini = 0.5\nsamples = 283\nvalue = [212, 216]\nclass = No')]
```



In [26]:
```
rf_best.feature_importances_
```

Out[26]:
```
array([0.02266818, 0.01779351, 0.08494914, 0.00303683, 0.05861945,
       0.05858221, 0.06054766, 0.00585816, 0.04014095, 0.        ,
       0.0543034 , 0.12675396, 0.12083748, 0.14386263, 0.0515548 ,
       0.00618813, 0.05709013, 0.        , 0.02358964, 0.06362373])
```

In [27]:
```
imp_df = pd.DataFrame({"Vername": x_train.columns,"Imp": rf_best.feature_importances_})
imp_df.sort_values(by="Imp", ascending=False)
```

Out[27]:

	Vername	Imp
13	ram	0.143863
11	px_height	0.126754
12	px_width	0.120837
2	clock_speed	0.084949
19	price_range	0.063624
6	int_memory	0.060548
4	fc	0.058619
5	four_g	0.058582
16	talk_time	0.057090
10	pc	0.054303
14	sc_h	0.051555
8	mobile_wt	0.040141
18	touch_screen	0.023590
0	battery_power	0.022668
1	blue	0.017794
15	sc_w	0.006188
7	m_dep	0.005858
3	dual_sim	0.003037
9	n_cores	0.000000
17	three_g	0.000000