In [1]:

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

In [2]:

```
train_df = pd.read_csv(r"C:\Users\jyothi reddy\Downloads\Mobile_Price_Classification_train.csv")
train_df
```

Out[2]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_core
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 [3]:

```
test_df = pd.read_csv(r"C:\Users\jyothi reddy\Downloads\Mobile_Price_Classification_test.csv")
test_df
```

Out[3]:

	id	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	...
0	1	1043	1	1.8	1	14	0	5	0.1	193	...
1	2	841	1	0.5	1	4	1	61	0.8	191	...
2	3	1807	1	2.8	0	1	0	27	0.9	186	...
3	4	1546	0	0.5	1	18	1	25	0.5	96	...
4	5	1434	0	1.4	0	11	1	49	0.5	108	...
...	...	...	...	...	...	...	...	...	...	...	...
995	996	1700	1	1.9	0	0	1	54	0.5	170	...
996	997	609	0	1.8	1	0	0	13	0.9	186	...
997	998	1185	0	1.4	0	1	1	8	0.5	80	...
998	999	1533	1	0.5	1	0	0	50	0.4	171	...
999	1000	1270	1	0.5	0	4	1	35	0.1	140	...

1000 rows × 21 columns

In [4]:

```
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: 328.3 KB
```

In [5]:

```
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 [6]:

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

In [7]:

```
x=test_df.drop('wifi',axis=1)
y=test_df['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 [14]:

```
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[14]:

```
GridSearchCV
estimator: RandomForestClassifier
RandomForestClassifier
```

In [13]:

```
grid_search.best_score_
```

Out[13]:

```
0.5528571428571429
```

In [15]:

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

Out[15]:

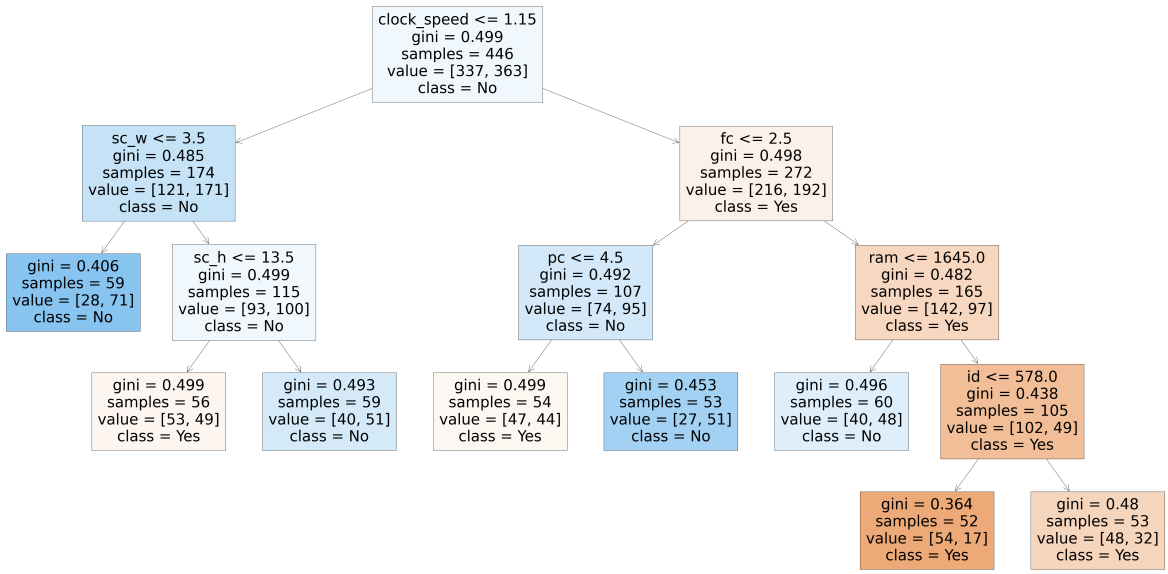
```
RandomForestClassifier
RandomForestClassifier(max_depth=10, min_samples_leaf=50, n_estimators=25)
```

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
```

Out[16]:

```
[Text(0.39285714285714285, 0.9, 'clock_speed <= 1.15\ngini = 0.499\nsamples = 446
\nvalue = [337, 363]\nclass = No'),
 Text(0.14285714285714285, 0.7, 'sc_w <= 3.5\ngini = 0.485\nsamples = 174\nvalue =
[121, 171]\nclass = No'),
 Text(0.07142857142857142, 0.5, 'gini = 0.406\nsamples = 59\nvalue = [28, 71]\ncla
ss = No'),
 Text(0.21428571428571427, 0.5, 'sc_h <= 13.5\ngini = 0.499\nsamples = 115\nvalue
= [93, 100]\nclass = No'),
 Text(0.14285714285714285, 0.3, 'gini = 0.499\nsamples = 56\nvalue = [53, 49]\ncla
ss = Yes'),
 Text(0.2857142857142857, 0.3, 'gini = 0.493\nsamples = 59\nvalue = [40, 51]\nclas
s = No'),
 Text(0.6428571428571429, 0.7, 'fc <= 2.5\ngini = 0.498\nsamples = 272\nvalue = [2
16, 192]\nclass = Yes'),
 Text(0.5, 0.5, 'pc <= 4.5\ngini = 0.492\nsamples = 107\nvalue = [74, 95]\nclass =
No'),
 Text(0.42857142857142855, 0.3, 'gini = 0.499\nsamples = 54\nvalue = [47, 44]\ncla
ss = Yes'),
 Text(0.5714285714285714, 0.3, 'gini = 0.453\nsamples = 53\nvalue = [27, 51]\nclas
s = No'),
 Text(0.7857142857142857, 0.5, 'ram <= 1645.0\ngini = 0.482\nsamples = 165\nvalue
= [142, 97]\nclass = Yes'),
 Text(0.7142857142857143, 0.3, 'gini = 0.496\nsamples = 60\nvalue = [40, 48]\nclas
s = No'),
 Text(0.8571428571428571, 0.3, 'id <= 578.0\ngini = 0.438\nsamples = 105\nvalue =
[102, 49]\nclass = Yes'),
 Text(0.7857142857142857, 0.1, 'gini = 0.364\nsamples = 52\nvalue = [54, 17]\nclas
s = Yes'),
 Text(0.9285714285714286, 0.1, 'gini = 0.48\nsamples = 53\nvalue = [48, 32]\nclass
= Yes')]
```

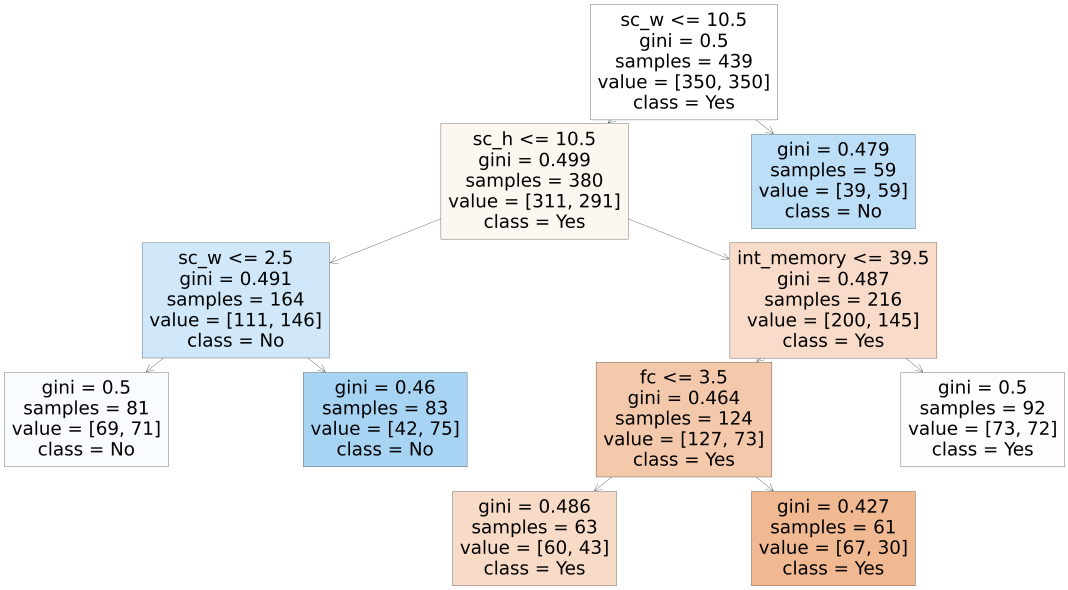


In [17]:

```
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[17]:

```
[Text(0.625, 0.9, 'sc_w <= 10.5\ngini = 0.5\nsamples = 439\nvalue = [350, 350]\ncl
ass = Yes'),
 Text(0.5, 0.7, 'sc_h <= 10.5\ngini = 0.499\nsamples = 380\nvalue = [311, 291]\ncl
ass = Yes'),
 Text(0.25, 0.5, 'sc_w <= 2.5\ngini = 0.491\nsamples = 164\nvalue = [111, 146]\ncl
ass = No'),
 Text(0.125, 0.3, 'gini = 0.5\nsamples = 81\nvalue = [69, 71]\nclass = No'),
 Text(0.375, 0.3, 'gini = 0.46\nsamples = 83\nvalue = [42, 75]\nclass = No'),
 Text(0.75, 0.5, 'int_memory <= 39.5\ngini = 0.487\nsamples = 216\nvalue = [200, 1
45]\nclass = Yes'),
 Text(0.625, 0.3, 'fc <= 3.5\ngini = 0.464\nsamples = 124\nvalue = [127, 73]\nclas
s = Yes'),
 Text(0.5, 0.1, 'gini = 0.486\nsamples = 63\nvalue = [60, 43]\nclass = Yes'),
 Text(0.75, 0.1, 'gini = 0.427\nsamples = 61\nvalue = [67, 30]\nclass = Yes'),
 Text(0.875, 0.3, 'gini = 0.5\nsamples = 92\nvalue = [73, 72]\nclass = Yes'),
 Text(0.75, 0.7, 'gini = 0.479\nsamples = 59\nvalue = [39, 59]\nclass = No')]
```



In [18]:

```
rf_best.feature_importances_
```

Out[18]:

```
array([0.04726883, 0.11638729, 0.01252884, 0.09638047, 0.0091454 ,
       0.07220407, 0.02046292, 0.06236104, 0.0576862 , 0.09768758,
       0.        , 0.03919055, 0.03440459, 0.13943153, 0.07695983,
       0.02756412, 0.05546919, 0.02813257, 0.        , 0.00673496])
```

In [19]:

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

Out[19]:

	Varname	Imp
13	px_width	0.139432
1	battery_power	0.116387
9	mobile_wt	0.097688
3	clock_speed	0.096380
14	ram	0.076960
5	fc	0.072204
7	int_memory	0.062361
8	m_dep	0.057686
16	sc_w	0.055469
0	id	0.047269
11	pc	0.039191
12	px_height	0.034405
17	talk_time	0.028133
15	sc_h	0.027564
6	four_g	0.020463
2	blue	0.012529
4	dual_sim	0.009145
19	touch_screen	0.006735
18	three_g	0.000000
10	n_cores	0.000000

In [ ]: