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\Teju\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_cor
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\Teju\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: 328.2 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 [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]:

```
0.5628571428571428
```

In [14]:

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

Out[14]:

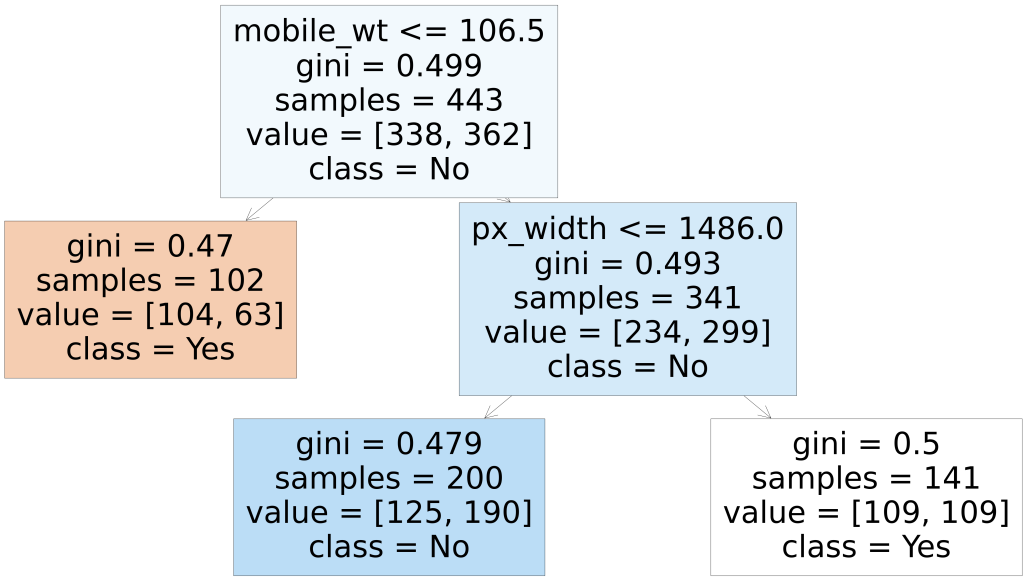
```
RandomForestClassifier
RandomForestClassifier(max_depth=20, min_samples_leaf=100)
```

In [15]:

```
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=Tru
```

Out[15]:

```
[Text(0.4, 0.8333333333333334, 'mobile_wt <= 106.5\ngini = 0.499\nsamples = 443\n
value = [338, 362]\nclass = No'),
 Text(0.2, 0.5, 'gini = 0.47\nsamples = 102\nvalue = [104, 63]\nclass = Yes'),
 Text(0.6, 0.5, 'px_width <= 1486.0\ngini = 0.493\nsamples = 341\nvalue = [234, 2
99]\nclass = No'),
 Text(0.4, 0.16666666666666666, 'gini = 0.479\nsamples = 200\nvalue = [125, 190]
\nclass = No'),
 Text(0.8, 0.16666666666666666, 'gini = 0.5\nsamples = 141\nvalue = [109, 109]\nc
lass = Yes')]
```

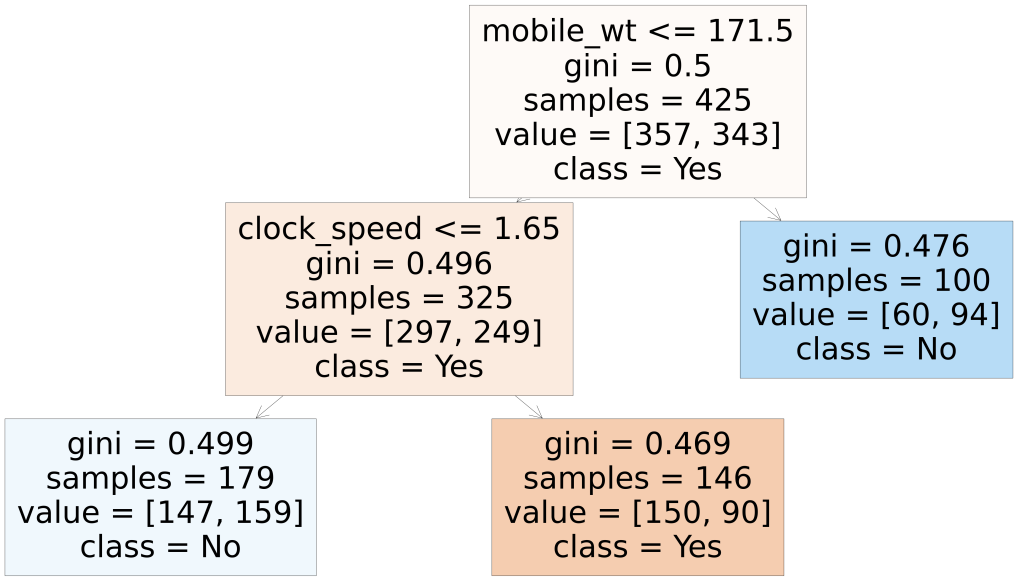


In [16]:

```
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=Tru
```

Out[16]:

```
[Text(0.6, 0.8333333333333334, 'mobile_wt <= 171.5\ngini = 0.5\nsamples = 425\nva
lue = [357, 343]\nclass = Yes'),
 Text(0.4, 0.5, 'clock_speed <= 1.65\ngini = 0.496\nsamples = 325\nvalue = [297,
249]\nclass = Yes'),
 Text(0.2, 0.16666666666666666, 'gini = 0.499\nsamples = 179\nvalue = [147, 159]
\nclass = No'),
 Text(0.6, 0.16666666666666666, 'gini = 0.469\nsamples = 146\nvalue = [150, 90]\n
class = Yes'),
 Text(0.8, 0.5, 'gini = 0.476\nsamples = 100\nvalue = [60, 94]\nclass = No')]
```



In [17]:

```
rf_best.feature_importances_
```

Out[17]:

```
array([0.05544549, 0.05821866, 0.00450128, 0.10635431, 0.00296943,
       0.05768001, 0.03398014, 0.0887642 , 0.05372867, 0.09271759,
       0.00961947, 0.02561059, 0.07041091, 0.17798303, 0.08951605,
       0.02074446, 0.0173649 , 0.02452633, 0.0069719 , 0.00289259])
```

In [18]:

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

Out[18]:

	Varname	Imp
13	px_width	0.177983
3	clock_speed	0.106354
9	mobile_wt	0.092718
14	ram	0.089516
7	int_memory	0.088764
12	px_height	0.070411
1	battery_power	0.058219
5	fc	0.057680
0	id	0.055445
8	m_dep	0.053729
6	four_g	0.033980
11	pc	0.025611
17	talk_time	0.024526
15	sc_h	0.020744
16	sc_w	0.017365
10	n_cores	0.009619
18	three_g	0.006972
2	blue	0.004501
4	dual_sim	0.002969
19	touch_screen	0.002893