In [13]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [14]:
```
df=pd.read_csv(r"C:\Users\DELL\Downloads\Mobile_Price_Classification_train.csv")
```

In [15]:
```
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 [16]:
```
df.isnull().sum()
```

Out[16]:
```
battery_power    0
blue             0
clock_speed      0
dual_sim         0
fc               0
four_g           0
int_memory       0
m_dep            0
mobile_wt        0
n_cores          0
pc               0
px_height        0
px_width         0
ram              0
sc_h             0
sc_w             0
talk_time        0
three_g          0
touch_screen     0
wifi             0
price_range      0
dtype: int64
```

In [17]:
```
X=df.drop('dual_sim',axis=1)
y=df['dual_sim']
```

In [18]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,train_size=0.4,random_state=40)
X_train.shape,X_test.shape
```

Out[18]:
```
((800, 20), (1200, 20))
```

In [19]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(X_train,y_train)
```

Out[19]:
```
RandomForestClassifier
RandomForestClassifier()
```

In [20]:
```
rf=RandomForestClassifier()
```

In [21]:
```
params = {'max_depth':[1,2,3,4,5],'min_samples_leaf':[5,10,25,50,100,200],'n_estimators':[10,25,30,50,100,20
```

In [22]:
```
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[22]:
```
GridSearchCV
estimator: RandomForestClassifier
RandomForestClassifier
```

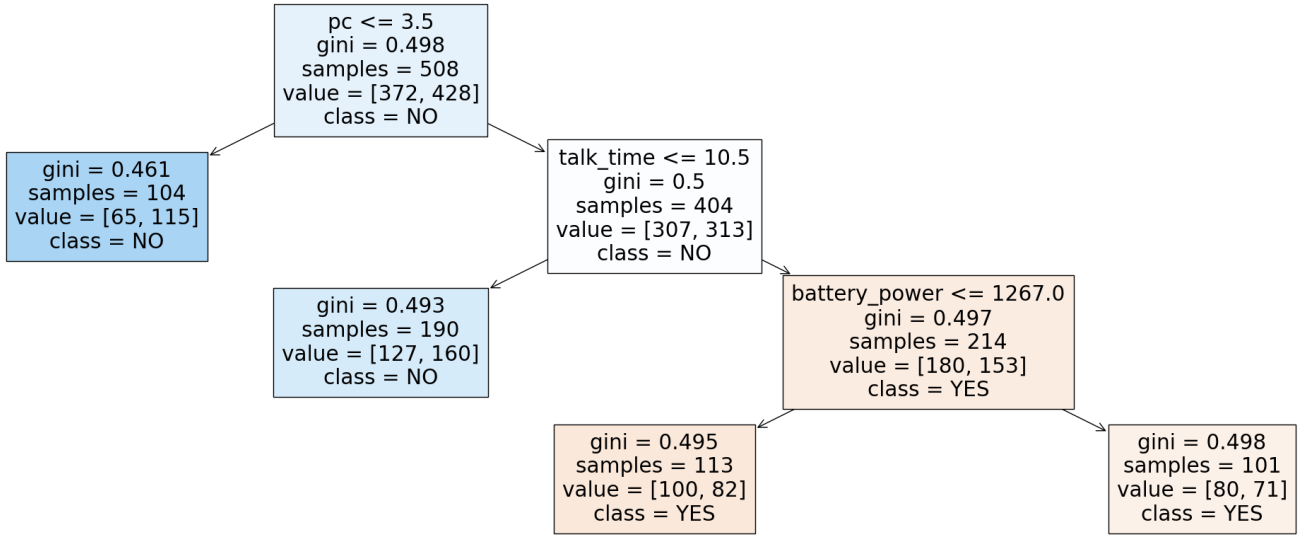
In [23]:
```
grid_search.best_score_
```

Out[23]:
```
0.55125
```

In [24]:
```
rf_best = grid_search.best_estimator_
print(rf_best)
```

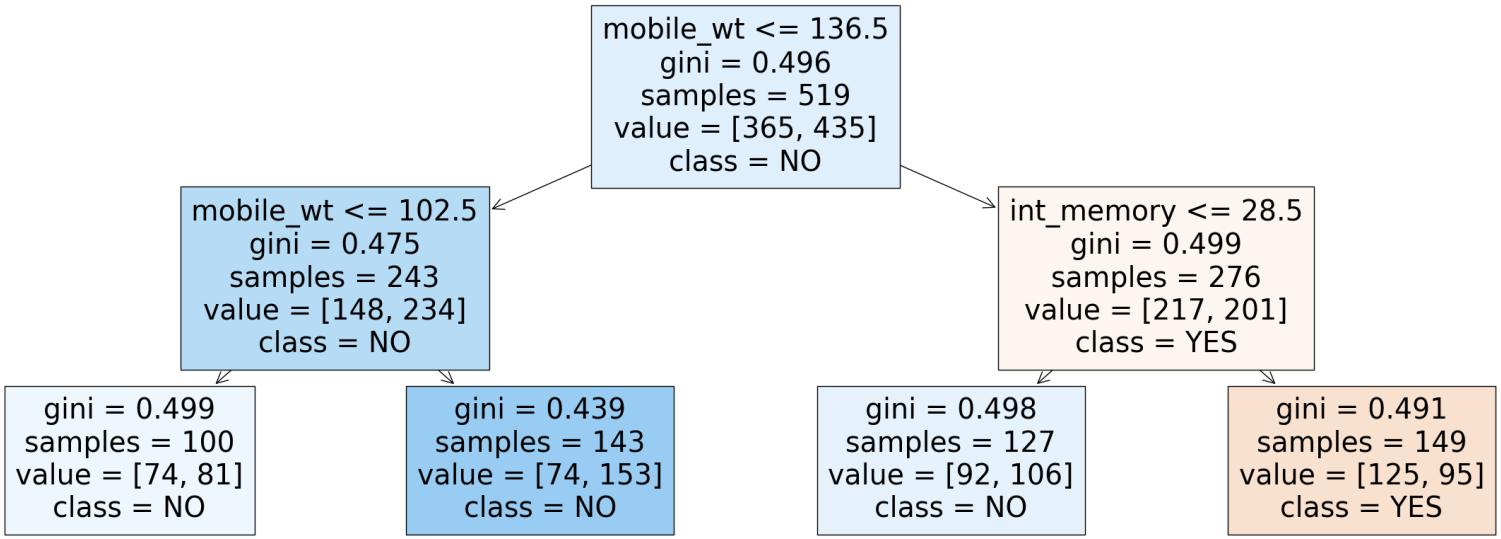
```
RandomForestClassifier(max_depth=5, min_samples_leaf=100)
```

In [25]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(30,10))
plot_tree(rf_best.estimators_[5],feature_names=X.columns,class_names=["YES","NO"],filled=True);
```



In [27]:
```
plt.figure(figsize=(30,10))
plot_tree(rf_best.estimators_[7],feature_names=X.columns,class_names=["YES","NO"],filled=True)
```

Out[27]:
```
[Text(0.5, 0.8333333333333334, 'mobile_wt <= 136.5\ngini = 0.496\nsamples = 519\nvalue = [365, 435]\nclass
= NO'),
 Text(0.25, 0.5, 'mobile_wt <= 102.5\ngini = 0.475\nsamples = 243\nvalue = [148, 234]\nclass = NO'),
 Text(0.125, 0.16666666666666666, 'gini = 0.499\nsamples = 100\nvalue = [74, 81]\nclass = NO'),
 Text(0.375, 0.16666666666666666, 'gini = 0.439\nsamples = 143\nvalue = [74, 153]\nclass = NO'),
 Text(0.75, 0.5, 'int_memory <= 28.5\ngini = 0.499\nsamples = 276\nvalue = [217, 201]\nclass = YES'),
 Text(0.625, 0.16666666666666666, 'gini = 0.498\nsamples = 127\nvalue = [92, 106]\nclass = NO'),
 Text(0.875, 0.16666666666666666, 'gini = 0.491\nsamples = 149\nvalue = [125, 95]\nclass = YES')]
```



In [29]:
```
rf_best.feature_importances_
```

Out[29]:
```
array([0.04346891, 0.011709  , 0.03123387, 0.10019296, 0.00399238,
       0.09685647, 0.10673891, 0.078965  , 0.02735128, 0.13166197,
       0.07919367, 0.06659341, 0.05033528, 0.03504932, 0.01085217,
       0.06740583, 0.00407071, 0.00990489, 0.03239559, 0.01202837])
```

In [30]:
```
imp_df = pd.DataFrame({"Varname":X_train.columns,"Imp":rf_best.feature_importances_})
```

In [31]:
```
imp_df.sort_values(by="Imp",ascending=False)
```

Out[31]:

	Varname	Imp
9	pc	0.131662
6	m_dep	0.106739
3	fc	0.100193
5	int_memory	0.096856
10	px_height	0.079194
7	mobile_wt	0.078965
15	talk_time	0.067406
11	px_width	0.066593
12	ram	0.050335
0	battery_power	0.043469
13	sc_h	0.035049
18	wifi	0.032396
2	clock_speed	0.031234
8	n_cores	0.027351
19	price_range	0.012028
1	blue	0.011709
14	sc_w	0.010852
17	touch_screen	0.009905
16	three_g	0.004071
4	four_g	0.003992

In [ ]: