

```
In [1]: import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
train_df=pd.read_csv(r"C:\Users\DELL\Downloads\Mobile_Price_Classification_train.csv")
train_df
```

Out[1]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_heig
0	842	0	2.2	0	1	0	7	0.6	188	2	...	127
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	90
2	563	1	0.5	1	2	1	41	0.9	145	5	...	127
3	615	1	2.5	0	0	0	10	0.8	131	6	...	127
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	127
...
1995	794	1	0.5	1	0	1	2	0.8	106	6	...	127
1996	1965	1	2.6	1	0	0	39	0.2	187	4	...	90
1997	1911	0	0.9	1	1	1	36	0.7	108	8	...	80
1998	1512	0	0.9	0	4	1	46	0.1	145	5	...	30
1999	510	1	2.0	1	5	1	45	0.9	168	6	...	40

2000 rows × 21 columns

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

Out[2]:

	id	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	...	pc	px_heig
0	1	1043	1	1.8	1	14	0	5	0.1	193	...	16	20
1	2	841	1	0.5	1	4	1	61	0.8	191	...	12	70
2	3	1807	1	2.8	0	1	0	27	0.9	186	...	4	127
3	4	1546	0	0.5	1	18	1	25	0.5	96	...	20	20
4	5	1434	0	1.4	0	11	1	49	0.5	108	...	18	70
...
995	996	1700	1	1.9	0	0	1	54	0.5	170	...	17	60
996	997	609	0	1.8	1	0	0	13	0.9	186	...	2	110
997	998	1185	0	1.4	0	1	1	8	0.5	80	...	12	40
998	999	1533	1	0.5	1	0	0	50	0.4	171	...	12	30
999	1000	1270	1	0.5	0	4	1	35	0.1	140	...	19	40

1000 rows × 21 columns

In [3]: 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 [4]: 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 [5]: x=train_df.drop('wifi',axis=1)
        y=train_df['wifi']
```

```
In [6]: x=test_df.drop('wifi',axis=1)
        y=test_df['wifi']
```

```
In [7]: train_df['blue'].value_counts()
```

```
Out[7]: blue
0      1010
1       990
Name: count, dtype: int64
```

```
In [8]: test_df['blue'].value_counts()
```

```
Out[8]: blue
1       516
0       484
Name: count, dtype: int64
```

```
In [9]: T={"three_g":{'Yes':1,'No':0}}
train_df=train_df.replace(T)
print(train_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 [10]: T={"three_g":{'Yes':1,'No':0}}
test_df=test_df.replace(T)
print(test_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 [11]: x=train_df.drop('dual_sim',axis=1)
y=train_df['dual_sim']
```

```
In [12]: x=test_df.drop('dual_sim',axis=1)
y=test_df['dual_sim']
```

```
In [13]: 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[13]: ((700, 20), (300, 20))
```

```
In [14]: from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

```
Out[14]: ▾ RandomForestClassifier
RandomForestClassifier()
```

```
In [15]: rf=RandomForestClassifier()
```

```
In [16]: 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 [17]: 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[17]: ▸ GridSearchCV
▸ estimator: RandomForestClassifier
    ▸ RandomForestClassifier
```

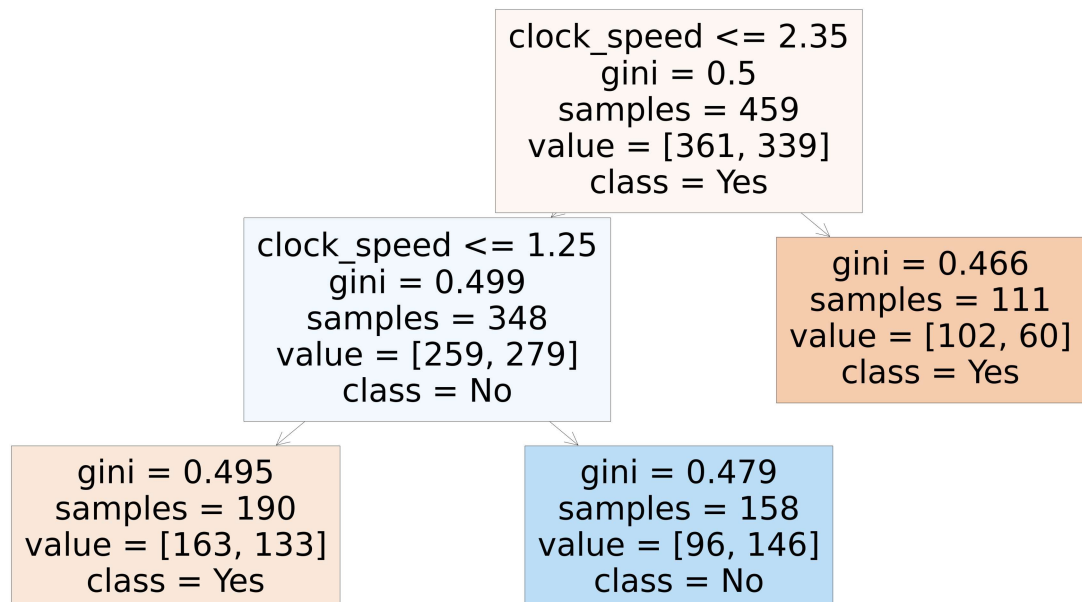
```
In [18]: grid_search.best_score_
```

```
Out[18]: 0.54
```

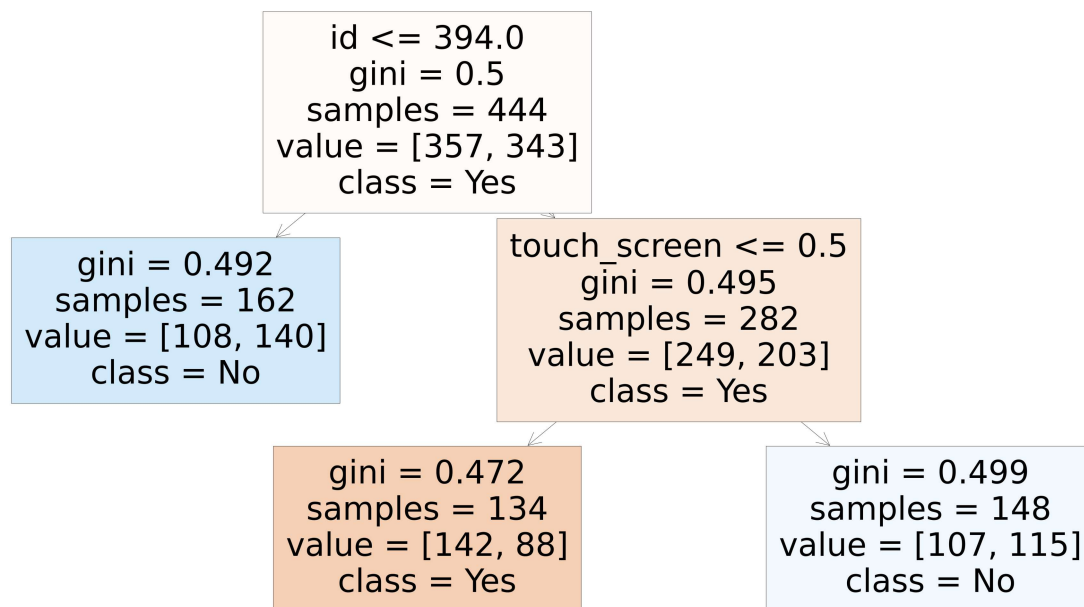
```
In [19]: rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=20, min_samples_leaf=100, n_estimators=25)
```

```
In [20]: 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 [21]: 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)
```



```
In [22]: rf_best.feature_importances_
```

```
Out[22]: array([0.03240378, 0.25463355, 0.02769553, 0.10667286, 0.03422312,  
0.00000000, 0.08003888, 0.00170657, 0.0281155 , 0.02131487,  
0.12666665, 0.03540498, 0.03366157, 0.04305681, 0.0645145 ,  
0.04476571, 0.01870482, 0.00000000, 0.02628394, 0.02013636])
```

```
In [23]: imp_df=pd.DataFrame({'Varname':x_train.columns,"Imp":rf_best.feature_importances_})
imp_df.sort_values(by="Imp",ascending=False)
```

Out[23]:

	Varname	Imp
1	battery_power	0.254634
10	pc	0.126667
3	clock_speed	0.106673
6	int_memory	0.080039
14	sc_h	0.064515
15	sc_w	0.044766
13	ram	0.043057
11	px_height	0.035405
4	fc	0.034223
12	px_width	0.033662
0	id	0.032404
8	mobile_wt	0.028115
2	blue	0.027696
18	touch_screen	0.026284
9	n_cores	0.021315
19	wifi	0.020136
16	talk_time	0.018705
7	m_dep	0.001707
5	four_g	0.000000
17	three_g	0.000000

In []: