In [1]:

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

In [2]:

```
df=pd.read_csv(r"C:\Users\user\Downloads\ionosphere.csv")
df
```

Out[2]:

	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.1	0.03760	...	-0.51171	0
0	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549	...	-0.26569	-0
1	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	...	-0.40220	0
2	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000	...	0.90695	0
3	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	...	-0.65158	0
4	1	0	0.02337	-0.00592	-0.09924	-0.11949	-0.00763	-0.11824	0.14706	0.06637	...	-0.01535	-0
...	...	...	...	...	...	...	...	...	...	...	...	...	
345	1	0	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622	...	-0.04202	0
346	1	0	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	...	0.01361	0
347	1	0	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446	...	0.03193	0
348	1	0	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	...	-0.02099	0
349	1	0	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139	...	-0.15114	0

350 rows × 35 columns

In [3]:

```
df['g'].value_counts()
```

Out[3]:

```
g    224
b    126
Name: g, dtype: int64
```

In [4]:

```
df['g'].value_counts()
```

Out[4]:

```
g    224
b    126
Name: g, dtype: int64
```

In [5]:

```
x=df.drop('g',axis=1)
y=df['g']
```

In [6]:

```
g1={"g":{'g':1,'b':2}}
df=df.replace(g1)
print(df)
```

```
     1  0  0.99539  -0.05889  0.85243  0.02306  0.83398  -0.37708      1.1  \
0    1  0  1.00000  -0.18829  0.93035 -0.36156 -0.10868  -0.93597  1.00000   
1    1  0  1.00000  -0.03365  1.00000  0.00485  1.00000  -0.12062  0.88965   
2    1  0  1.00000  -0.45161  1.00000  1.00000  0.71216  -1.00000  0.00000   
3    1  0  1.00000  -0.02401  0.94140  0.06531  0.92106  -0.23255  0.77152   
4    1  0  0.02337  -0.00592 -0.09924 -0.11949 -0.00763  -0.11824  0.14706   
..  .. ..      ...       ...      ...      ...      ...       ...      ...   
345  1  0  0.83508   0.08298  0.73739 -0.14706  0.84349  -0.05567  0.90441   
346  1  0  0.95113   0.00419  0.95183 -0.02723  0.93438  -0.01920  0.94590   
347  1  0  0.94701  -0.00034  0.93207 -0.03227  0.95177  -0.03431  0.95584   
348  1  0  0.90608  -0.01657  0.98122 -0.01989  0.95691  -0.03646  0.85746   
349  1  0  0.84710   0.13533  0.73638 -0.06151  0.87873   0.08260  0.88928   

     0.03760  ...  -0.51171  0.41078  -0.46168  0.21266  -0.34090  0.42267  \
0   -0.04549  ...  -0.26569 -0.20468  -0.18401 -0.19040  -0.11593 -0.16626   
1    0.01198  ...  -0.40220  0.58984  -0.22145  0.43100  -0.17365  0.60436   
2    0.00000  ...   0.90695  0.51613   1.00000  1.00000  -0.20099  0.25682   
3   -0.16399  ...  -0.65158  0.13290  -0.53206  0.02431  -0.62197 -0.05707   
4    0.06637  ...  -0.01535 -0.03240   0.09223 -0.07859   0.00732  0.00000   
..       ...  ...       ...      ...       ...      ...       ...      ...   
345 -0.04622  ...  -0.04202  0.83479   0.00123  1.00000   0.12815  0.86660   
346  0.01606  ...   0.01361  0.93522   0.04925  0.93159   0.08168  0.94066   
347  0.02446  ...   0.03193  0.92489   0.02542  0.92120   0.02242  0.92459   
348  0.00110  ...  -0.02099  0.89147  -0.07760  0.82983  -0.17238  0.96022   
349 -0.09139  ...  -0.15114  0.81147  -0.04822  0.78207  -0.00703  0.75747   

     -0.54487  0.18641  -0.45300  g  
0    -0.06288 -0.13738  -0.02447  2  
1    -0.24180  0.56045  -0.38238  1  
2     1.00000 -0.32382   1.00000  2  
3    -0.59573 -0.04608  -0.65697  1  
4     0.00000 -0.00039   0.12011  2  
..        ...      ...       ... ..  
345  -0.10714  0.90546  -0.04307  1  
346  -0.00035  0.91483   0.04712  1  
347   0.00442  0.92697  -0.00577  1  
348  -0.03757  0.87403  -0.16243  1  
349  -0.06678  0.85764  -0.06151  1  

[350 rows x 35 columns]
```

In [7]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.70)
```

In [8]:

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

Out[8]:

```
RandomForestClassifier()
```

In [9]:

```
parameters= {
    "max_depth":[1,2,3,4,5],
    "min_samples_leaf":[5,10,15,20,25],
    'n_estimators':[10,20,30,40,50]
}
```

In [11]:

```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
```

```
grid_search.fit(x_train,y_train)
```

```
Out[11]: GridSearchCV(cv=2, estimator=RandomForestClassifier(),
                      param_grid={'max_depth': [1, 2, 3, 4, 5],
                                  'min_samples_leaf': [5, 10, 15, 20, 25],
                                  'n_estimators': [10, 20, 30, 40, 50]},
                      scoring='accuracy')
```

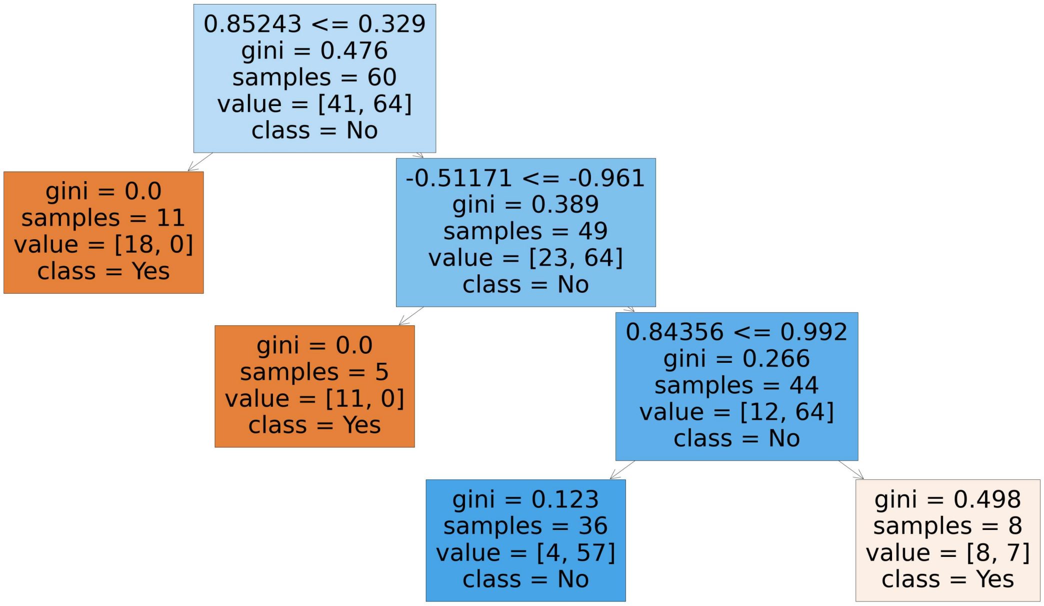
```
In [12]: grid_search.best_score_
```

```
Out[12]: 0.8760885341074021
```

```
In [15]: rfc_best=grid_search.best_estimator_
```

```
In [17]: from sklearn.tree import plot_tree
         plt.figure(figsize=(80,40))
         plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],fill
```

```
Out[17]: [Text(1488.0, 1902.6000000000001, '0.85243 <= 0.329\ngini = 0.476\nsamples = 60\nvalue =
         [41, 64]\nclass = No'),
          Text(744.0, 1359.0, 'gini = 0.0\nsamples = 11\nvalue = [18, 0]\nclass = Yes'),
          Text(2232.0, 1359.0, '-0.51171 <= -0.961\ngini = 0.389\nsamples = 49\nvalue = [23, 64]
         \nclass = No'),
          Text(1488.0, 815.4000000000001, 'gini = 0.0\nsamples = 5\nvalue = [11, 0]\nclass = Ye
         s'),
          Text(2976.0, 815.4000000000001, '0.84356 <= 0.992\ngini = 0.266\nsamples = 44\nvalue =
         [12, 64]\nclass = No'),
          Text(2232.0, 271.79999999999995, 'gini = 0.123\nsamples = 36\nvalue = [4, 57]\nclass =
         No'),
          Text(3720.0, 271.79999999999995, 'gini = 0.498\nsamples = 8\nvalue = [8, 7]\nclass = Ye
         s')]
```



```
In [ ]:
```