In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [34]:
```
df=pd.read_csv("1_ionosphere - 1_ionosphere.csv")
df
```

Out[34]:

	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.1	0.0376	...	-0.51171	0.41078	-0.461(
0	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549	...	-0.26569	-0.20468	-0.184(
1	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	...	-0.40220	0.58984	-0.221(
2	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000	...	0.90695	0.51613	1.000(
3	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	...	-0.65158	0.13290	-0.532(
4	1	0	0.02337	-0.00592	-0.09924	-0.11949	-0.00763	-0.11824	0.14706	0.06637	...	-0.01535	-0.03240	0.092(
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
345	1	0	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622	...	-0.04202	0.83479	0.001(
346	1	0	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	...	0.01361	0.93522	0.049(
347	1	0	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446	...	0.03193	0.92489	0.025(
348	1	0	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	...	-0.02099	0.89147	-0.077(
349	1	0	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139	...	-0.15114	0.81147	-0.048(

350 rows × 35 columns

In [35]:
```
df1=df.fillna(value=0)
df1
```

Out[35]:

	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.1	0.0376	...	-0.51171	0.41078	-0.461(
0	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549	...	-0.26569	-0.20468	-0.184(
1	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	...	-0.40220	0.58984	-0.221(
2	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000	...	0.90695	0.51613	1.000(
3	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	...	-0.65158	0.13290	-0.532(
4	1	0	0.02337	-0.00592	-0.09924	-0.11949	-0.00763	-0.11824	0.14706	0.06637	...	-0.01535	-0.03240	0.092(
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
345	1	0	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622	...	-0.04202	0.83479	0.001(
346	1	0	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	...	0.01361	0.93522	0.049(
347	1	0	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446	...	0.03193	0.92489	0.025(
348	1	0	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	...	-0.02099	0.89147	-0.077(
349	1	0	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139	...	-0.15114	0.81147	-0.048(

350 rows × 35 columns

In [36]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 350 entries, 0 to 349
Data columns (total 35 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   1          350 non-null    int64  
 1   0          350 non-null    int64  
 2   0.99539    350 non-null    float64
 3   -0.05889   350 non-null    float64
 4   0.85243    350 non-null    float64
 5   0.02306    350 non-null    float64
 6   0.83398    350 non-null    float64
 7   -0.37708   350 non-null    float64
 8   1.1        350 non-null    float64
 9   0.0376     350 non-null    float64
 10  0.85243.1  350 non-null    float64
 11  -0.17755   350 non-null    float64
 12  0.59755    350 non-null    float64
 13  -0.44945   350 non-null    float64
 14  0.60536    350 non-null    float64
 15  -0.38223   350 non-null    float64
 16  0.84356    350 non-null    float64
 17  -0.38542   350 non-null    float64
 18  0.58212    350 non-null    float64
 19  -0.32192   350 non-null    float64
 20  0.56971    350 non-null    float64
 21  -0.29674   350 non-null    float64
 22  0.36946    350 non-null    float64
 23  -0.47357   350 non-null    float64
 24  0.56811    350 non-null    float64
 25  -0.51171   350 non-null    float64
 26  0.41078    350 non-null    float64
 27  -0.46168   350 non-null    float64
 28  0.21266    350 non-null    float64
 29  -0.3409    350 non-null    float64
 30  0.42267    350 non-null    float64
 31  -0.54487   350 non-null    float64
 32  0.18641    350 non-null    float64
 33  -0.453     350 non-null    float64
 34  g          350 non-null    object 
dtypes: float64(32), int64(2), object(1)
memory usage: 95.8+ KB
```

In [37]:
```
df['g'].value_counts()
```

Out[37]:
```
g    224
b    126
Name: g, dtype: int64
```

In [38]:
```
x=df.drop('g',axis=1)
y=df['g']
```

In [39]:
```
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

      0.0376  ...  -0.51171  0.41078  -0.46168  0.21266  -0.3409  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   -0.453  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 [40]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [41]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[41]:
```
RandomForestClassifier()
```

In [42]:
```
parameters = {'max_depth':[1,2,3,4,5],
              'min_samples_leaf':[5,10,15,20,25],
               'n_estimators':[10,20,30,40,50]}
```

In [43]:
```
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[43]:
```
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 [44]:
```
grid_search.best_score_
```

Out[44]:
```
0.9347594295615087
```

In [45]:
```
rfc_best =grid_search.best_estimator_
```

In [46]:
```
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'],filled=True
```

Out[46]:
```
[Text(1785.6, 1956.96, '0.41078 <= 0.996\ngini = 0.467\nsamples = 151\nvalue = [91, 154]\ncla
ss = No'),
 Text(892.8, 1522.0800000000002, '0.0376 <= -0.395\ngini = 0.279\nsamples = 110\nvalue = [29,
144]\nclass = No'),
 Text(446.4, 1087.2, 'gini = 0.469\nsamples = 5\nvalue = [5, 3]\nclass = Yes'),
 Text(1339.1999999999998, 1087.2, '0.85243 <= 0.107\ngini = 0.249\nsamples = 105\nvalue = [2
4, 141]\nclass = No'),
 Text(892.8, 652.3200000000002, 'gini = 0.0\nsamples = 10\nvalue = [19, 0]\nclass = Yes'),
 Text(1785.6, 652.3200000000002, '-0.51171 <= 0.161\ngini = 0.066\nsamples = 95\nvalue = [5,
141]\nclass = No'),
 Text(1339.1999999999998, 217.44000000000005, 'gini = 0.034\nsamples = 73\nvalue = [2, 115]\n
class = No'),
 Text(2232.0, 217.44000000000005, 'gini = 0.185\nsamples = 22\nvalue = [3, 26]\nclass = No'),
 Text(2678.3999999999996, 1522.0800000000002, '0.83398 <= 0.722\ngini = 0.239\nsamples = 41\n
value = [62, 10]\nclass = Yes'),
 Text(2232.0, 1087.2, 'gini = 0.0\nsamples = 19\nvalue = [34, 0]\nclass = Yes'),
 Text(3124.7999999999997, 1087.2, '0.0376 <= -0.106\ngini = 0.388\nsamples = 22\nvalue = [28,
10]\nclass = Yes'),
 Text(2678.3999999999996, 652.3200000000002, 'gini = 0.0\nsamples = 7\nvalue = [15, 0]\nclass
= Yes'),
 Text(3571.2, 652.3200000000002, '1.1 <= 0.9\ngini = 0.491\nsamples = 15\nvalue = [13, 10]\nc
lass = Yes'),
 Text(3124.7999999999997, 217.44000000000005, 'gini = 0.198\nsamples = 5\nvalue = [8, 1]\ncla
ss = Yes'),
 Text(4017.6, 217.44000000000005, 'gini = 0.459\nsamples = 10\nvalue = [5, 9]\nclass = No')]
```