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

In [199]:
```
data=pd.read_csv(r"C:\Users\user\Desktop\vicky\C5_health care diabetes.csv")
```

In [200]:
```
data.fillna(value=1)
```

Out[200]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunctio
0	6	148	72	35	0	33.6	0.62
1	1	85	66	29	0	26.6	0.35
2	8	183	64	0	0	23.3	0.67
3	1	89	66	23	94	28.1	0.16
4	0	137	40	35	168	43.1	2.28
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.17
764	2	122	70	27	0	36.8	0.34
765	5	121	72	23	112	26.2	0.24
766	1	126	60	0	0	30.1	0.34
767	1	93	70	31	0	30.4	0.31

768 rows × 9 columns

In [201]:
```
data.head()
```

Out[201]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction
0	6	148	72	35	0	33.6	0.627
1	1	85	66	29	0	26.6	0.351
2	8	183	64	0	0	23.3	0.672
3	1	89	66	23	94	28.1	0.167
4	0	137	40	35	168	43.1	2.288

In [202]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [204]:
```
data.columns
```

Out[204]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [203]:

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-203-1a842c172fec> in <module>
----> 1 data1=data[['User ID','Retweet Count','Mention Count','Mention Coun
t','Bot Label']]

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\frame.py in __getitem_
_(self, key)
   3028             if is_iterator(key):
   3029                 key = list(key)
-> 3030             indexer = self.loc._get_listlike_indexer(key, axis=1, rai
se_missing=True)[1]
   3031
   3032         # take() does not accept boolean indexers

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\indexing.py in _get_li
stlike_indexer(self, key, axis, raise_missing)
   1264             keyarr, indexer, new_indexer = ax._reindex_non_unique(key
arr)
   1265
-> 1266         self._validate_read_indexer(keyarr, indexer, axis, raise_miss
ing=raise_missing)
   1267         return keyarr, indexer
   1268

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\indexing.py in _valida
te_read_indexer(self, key, indexer, axis, raise_missing)
   1306             if missing == len(indexer):
   1307                 axis_name = self.obj._get_axis_name(axis)
-> 1308                 raise KeyError(f"None of [{key}] are in the [{axis_na
me}]")
   1309
   1310             ax = self.obj._get_axis(axis)

KeyError: "None of [Index(['User ID', 'Retweet Count', 'Mention Count', 'Ment
ion Count',\n       'Bot Label'],\n      dtype='object')] are in the [column
s]"
```

In [205]:

```
data['Outcome'].value_counts()
```

Out[205]:

```
0    500
1    268
Name: Outcome, dtype: int64
```

In [206]:

```
x=data.drop('Outcome',axis=1)
y=data['Outcome']
```

In [207]:

```
g1={"Outcome":{0:2,1:3}}
data=data.replace(g1)
print(data)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6   
1              1       85             66             29        0  26.6   
2              8      183             64              0        0  23.3   
3              1       89             66             23       94  28.1   
4              0      137             40             35      168  43.1   
..           ...      ...            ...            ...      ...   ...   
763           10      101             76             48      180  32.9   
764            2      122             70             27        0  36.8   
765            5      121             72             23      112  26.2   
766            1      126             60              0        0  30.1   
767            1       93             70             31        0  30.4   

     DiabetesPedigreeFunction  Age  Outcome  
0                       0.627   50        3  
1                       0.351   31        2  
2                       0.672   32        3  
3                       0.167   21        2  
4                       2.288   33        3  
..                        ...  ...      ...  
763                     0.171   63        2  
764                     0.340   27        2  
765                     0.245   30        2  
766                     0.349   47        3  
767                     0.315   23        2  

[768 rows x 9 columns]
```

In [208]:

```
from sklearn.model_selection import train_test_split
```

In [209]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [210]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [211]:

```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[211]:

```
RandomForestClassifier()
```

In [212]:

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


}
```

In [213]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="acc
grid_search.fit(x_train,y_train)
```

Out[213]:
```
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 [214]:
```
grid_search.best_score_
```

Out[214]:
```
0.7728458081340509
```

In [215]:
```
from sklearn.tree import plot_tree
```

In [216]:
```
rfc_best=grid_search.best_estimator_
```

In [217]:
```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','
```

Out[217]:
```
[Text(2418.0, 1902.6000000000001, 'Insulin <= 118.0\ngini = 0.466\nsamples =
341\nvalue = [339, 198]\nclass = Yes'),
 Text(1488.0, 1359.0, 'Insulin <= 15.0\ngini = 0.424\nsamples = 247\nvalue =
[269, 118]\nclass = Yes'),
 Text(744.0, 815.4000000000001, 'Pregnancies <= 5.5\ngini = 0.482\nsamples =
159\nvalue = [148, 101]\nclass = Yes'),
 Text(372.0, 271.79999999999995, 'gini = 0.433\nsamples = 100\nvalue = [110,
51]\nclass = Yes'),
 Text(1116.0, 271.79999999999995, 'gini = 0.491\nsamples = 59\nvalue = [38, 5
0]\nclass = No'),
 Text(2232.0, 815.4000000000001, 'BMI <= 26.2\ngini = 0.216\nsamples = 88\nva
lue = [121, 17]\nclass = Yes'),
 Text(1860.0, 271.79999999999995, 'gini = 0.0\nsamples = 29\nvalue = [47, 0]
\nclass = Yes'),
 Text(2604.0, 271.79999999999995, 'gini = 0.304\nsamples = 59\nvalue = [74, 1
7]\nclass = Yes'),
 Text(3348.0, 1359.0, 'Glucose <= 117.0\ngini = 0.498\nsamples = 94\nvalue =
[70, 80]\nclass = No'),
 Text(2976.0, 815.4000000000001, 'gini = 0.278\nsamples = 23\nvalue = [30, 6]
\nclass = Yes'),
 Text(3720.0, 815.4000000000001, 'Pregnancies <= 6.5\ngini = 0.456\nsamples =
71\nvalue = [40, 74]\nclass = No'),
 Text(3348.0, 271.79999999999995, 'gini = 0.487\nsamples = 54\nvalue = [36, 5
0]\nclass = No'),
 Text(4092.0, 271.79999999999995, 'gini = 0.245\nsamples = 17\nvalue = [4, 2
4]\nclass = No')]
```

In [ ]:

In [ ]: