Random Forest

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\health.csv")[0:1000]
df
```

Out[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

In [9]:
```
df.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 [10]:
```
df.columns
```

Out[10]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [11]:
```
d['Outcome'].value_counts()
```

Out[11]:
```
0    500
1    268
Name: Outcome, dtype: int64
```

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

In [13]:
```
g1={"Outcome":{"":0,0:1}}
d=d.replace(g1)
print(d)
```

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

[768 rows x 9 columns]
```

In [14]:
```
from sklearn.model_selection import train_test_split
```

In [15]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [16]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [17]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[17]:
```
RandomForestClassifier()
```

In [18]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf':[5,10,15,20,25],
            'n_estimators':[10,20,30,40,50]
}
```

In [19]:
```
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[19]:
```
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 [20]:
```
grid_search.best_score_
```

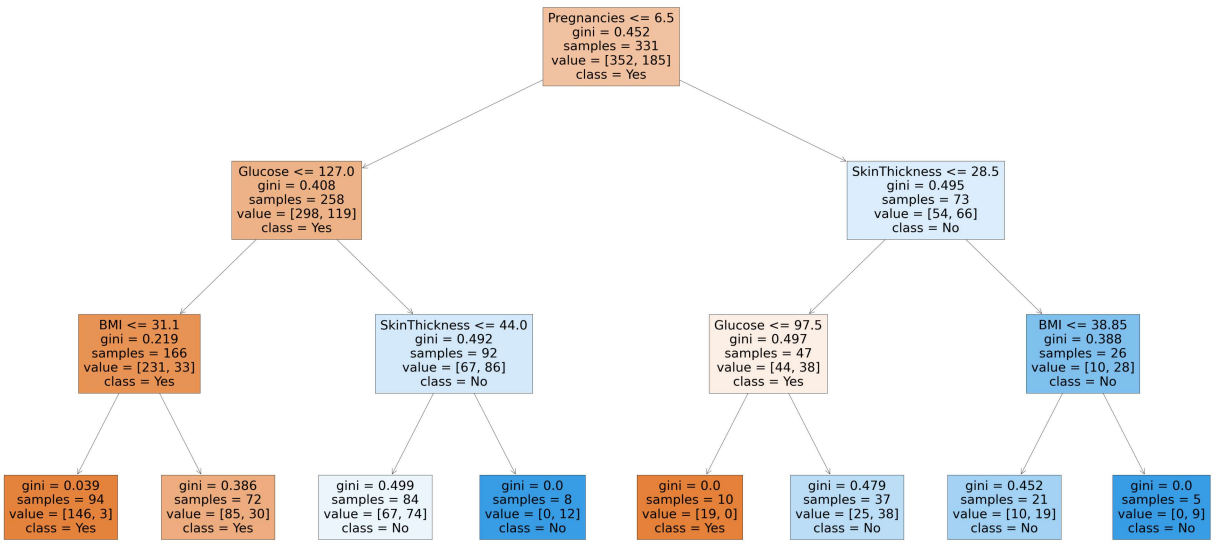
Out[20]:
```
0.7802599456250346
```

In [21]:
```
from sklearn.tree import plot_tree
```

In [22]:
```
rfc_best=grid_search.best_estimator_
```

In [23]:
```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],filled=True
```

Out[23]:
```
[Text(2232.0, 1902.6000000000001, 'Pregnancies <= 6.5\ngini = 0.452\nsamples = 331\nvalue =
[352, 185]\nclass = Yes'),
 Text(1116.0, 1359.0, 'Glucose <= 127.0\ngini = 0.408\nsamples = 258\nvalue = [298, 119]\ncla
ss = Yes'),
 Text(558.0, 815.4000000000001, 'BMI <= 31.1\ngini = 0.219\nsamples = 166\nvalue = [231, 33]
\nclass = Yes'),
 Text(279.0, 271.79999999999995, 'gini = 0.039\nsamples = 94\nvalue = [146, 3]\nclass = Ye
s'),
 Text(837.0, 271.79999999999995, 'gini = 0.386\nsamples = 72\nvalue = [85, 30]\nclass = Ye
s'),
 Text(1674.0, 815.4000000000001, 'SkinThickness <= 44.0\ngini = 0.492\nsamples = 92\nvalue =
[67, 86]\nclass = No'),
 Text(1395.0, 271.79999999999995, 'gini = 0.499\nsamples = 84\nvalue = [67, 74]\nclass = N
o'),
 Text(1953.0, 271.79999999999995, 'gini = 0.0\nsamples = 8\nvalue = [0, 12]\nclass = No'),
 Text(3348.0, 1359.0, 'SkinThickness <= 28.5\ngini = 0.495\nsamples = 73\nvalue = [54, 66]\nc
lass = No'),
 Text(2790.0, 815.4000000000001, 'Glucose <= 97.5\ngini = 0.497\nsamples = 47\nvalue = [44, 3
8]\nclass = Yes'),
 Text(2511.0, 271.79999999999995, 'gini = 0.0\nsamples = 10\nvalue = [19, 0]\nclass = Yes'),
 Text(3069.0, 271.79999999999995, 'gini = 0.479\nsamples = 37\nvalue = [25, 38]\nclass = N
o'),
 Text(3906.0, 815.4000000000001, 'BMI <= 38.85\ngini = 0.388\nsamples = 26\nvalue = [10, 28]
\nclass = No'),
 Text(3627.0, 271.79999999999995, 'gini = 0.452\nsamples = 21\nvalue = [10, 19]\nclass = N
o'),
 Text(4185.0, 271.79999999999995, 'gini = 0.0\nsamples = 5\nvalue = [0, 9]\nclass = No')]
```



In [ ]:
```
```