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

In [2]:
```
from sklearn.linear_model import LogisticRegression
```

In [3]:
```
df=pd.read_csv("bmi.csv").dropna()
df
```

Out[3]:

	Gender	Height	Weight	Index
0	Male	174	96	4
1	Male	189	87	2
2	Female	185	110	4
3	Female	195	104	3
4	Male	149	61	3
...	...	...	...	...
495	Female	150	153	5
496	Female	184	121	4
497	Female	141	136	5
498	Male	150	95	5
499	Male	173	131	5

500 rows × 4 columns

In [4]:
```
df.dropna(inplace=True)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 500 entries, 0 to 499
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Gender  500 non-null    object
 1   Height  500 non-null    int64 
 2   Weight  500 non-null    int64 
 3   Index   500 non-null    int64 
dtypes: int64(3), object(1)
memory usage: 19.5+ KB
```

In [6]:
```
feature_matrix = df[['Height','Weight','Index']]
target_vector = df['Gender']
```

In [7]:
```
feature_matrix.shape
```

Out[7]: (500, 3)

In [8]:
```
target_vector.shape
```

Out[8]: (500,)

In [9]:
```
from sklearn.preprocessing import StandardScaler
```

In [10]:
```
fs = StandardScaler().fit_transform(feature_matrix)
```

In [11]:
```
logr = LogisticRegression()
logr.fit(fs,target_vector)
```

Out[11]: LogisticRegression()

In [12]:
```
feature_matrix.shape
```

Out[12]: (500, 3)

In [13]:
```
target_vector.shape
```

Out[13]: (500,)

In [14]:
```
from sklearn.preprocessing import StandardScaler
```

In [15]:
```
fs = StandardScaler().fit_transform(feature_matrix)
```

In [16]:
```
logr = LogisticRegression()
logr.fit(fs,target_vector)
```

Out[16]: LogisticRegression()

In [17]:
```
observation=df[['Height','Weight','Index']]
```

In [18]:
```
prediction = logr.predict(observation)
prediction
```

Out[18]:
```
array(['Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
```

```
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Male', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Male', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Male', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Male', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
```

```
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Male', 'Female', 'Female', 'Female', 'Female',
       'Female', 'Female', 'Female'], dtype=object)
```

In [19]:
```
logr.classes_
```

Out[19]: `array(['Female', 'Male'], dtype=object)`

In [20]:
```
logr.predict_proba(observation)[0][1]
```

Out[20]: `0.013560340101419254`

In [27]:
```
df['Gender'].value_counts()
```

Out[27]:
```
Female    255
Male      245
Name: Gender, dtype: int64
```

In [30]:
```
x=df.drop('Gender', axis=1)
y=df['Gender']
```

In [31]:
```
g1={"Gender":{"Male":1, "Female":2}}
df=df.replace(g1)
df
```

Out[31]:

	Gender	Height	Weight	Index
0	1	174	96	4
1	1	189	87	2
2	2	185	110	4
3	2	195	104	3
4	1	149	61	3
...	...	...	...	...
495	2	150	153	5
496	2	184	121	4
497	2	141	136	5
498	1	150	95	5

	Gender	Height	Weight	Index
499	1	173	131	5

500 rows × 4 columns

In [32]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [33]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[33]:
```
RandomForestClassifier()
```

In [34]:
```
parameters = {'max_depth':[1,2,3,4,5],'min_samples_leaf':[5,10,15,20,25],'n_estimators'
```

In [35]:
```
from sklearn.model_selection import GridSearchCV
grid_search = GridSearchCV(estimator=rfc,param_grid= parameters,cv=2,scoring = "accurac
grid_search.fit(x_train,y_train)
```

Out[35]:
```
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')
```

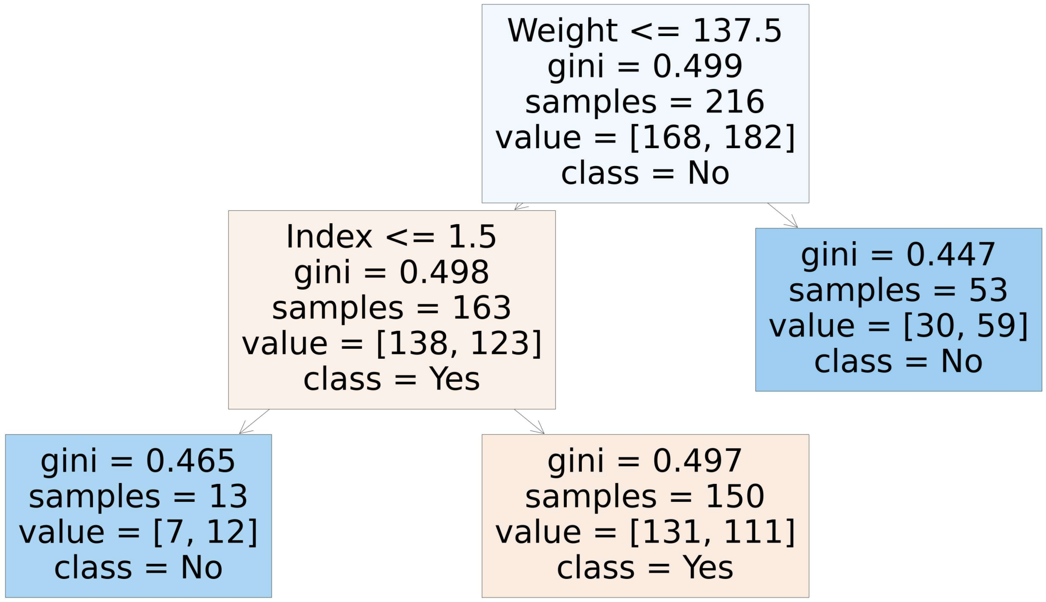
In [36]:
```
grid_search.best_score_
```

Out[36]:
```
0.5057142857142858
```

In [37]:
```
rfc_best = grid_search.best_estimator_
```

In [38]:
```
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'],fi
```

Out[38]:
```
[Text(2678.3999999999996, 1812.0, 'Weight <= 137.5\ngini = 0.499\nsamples = 216\nvalue =
[168, 182]\nclass = No'),
 Text(1785.6, 1087.2, 'Index <= 1.5\ngini = 0.498\nsamples = 163\nvalue = [138, 123]\ncl
ass = Yes'),
 Text(892.8, 362.39999999999986, 'gini = 0.465\nsamples = 13\nvalue = [7, 12]\nclass = N
o'),
 Text(2678.3999999999996, 362.39999999999986, 'gini = 0.497\nsamples = 150\nvalue = [13
1, 111]\nclass = Yes'),
 Text(3571.2, 1087.2, 'gini = 0.447\nsamples = 53\nvalue = [30, 59]\nclass = No')]
```


Weight <= 137.5
gini = 0.499
samples = 216
value = [168, 182]
class = No
Index <= 1.5
gini = 0.498
samples = 163
value = [138, 123]
class = Yes
gini = 0.447
samples = 53
value = [30, 59]
class = No
gini = 0.465
samples = 13
value = [7, 12]
class = No
gini = 0.497
samples = 150
value = [131, 111]
class = Yes