

In [110]:

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

In [275]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C6_bmi.csv")
a
```

Out[275]:

	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 [276]:

```
from sklearn.linear_model import LogisticRegression
```

In [277]:

```
a=a.head(10)
a
```

Out[277]:

	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
5	Male	189	104	3
6	Male	147	92	5
7	Male	154	111	5
8	Male	174	90	3
9	Female	169	103	4

In [278]:

```
a.columns
```

Out[278]:

```
Index(['Gender', 'Height', 'Weight', 'Index'], dtype='object')
```

In [279]:

```
b=a[['Height', 'Weight', 'Index']]
b
```

Out[279]:

	Height	Weight	Index
0	174	96	4
1	189	87	2
2	185	110	4
3	195	104	3
4	149	61	3
5	189	104	3
6	147	92	5
7	154	111	5
8	174	90	3
9	169	103	4

In [280]:

```
c=b.iloc[:,0:11]  
d=b.iloc[:, -1]
```

In [281]:

```
c.shape
```

Out[281]:

```
(10, 3)
```

In [282]:

```
d.shape
```

Out[282]:

```
(10,)
```

In [283]:

```
from sklearn.preprocessing import StandardScaler
```

In [284]:

```
fs=StandardScaler().fit_transform(c)
```

In [285]:

```
logr=LogisticRegression()  
logr.fit(fs,d)
```

Out[285]:

```
LogisticRegression()
```

In [289]:

```
e=[[2,5,77]]
```

In [290]:

```
prediction=logr.predict(e)  
prediction
```

Out[290]:

```
array([5], dtype=int64)
```

In [291]:

```
logr.classes_
```

Out[291]:

```
array([2, 3, 4, 5], dtype=int64)
```

In [292]:

```
logr.predict_proba(e)[0][0]
```

Out[292]:

1.280970264553129e-62

In [293]:

```
logr.predict_proba(e)[0][1]
```

Out[293]:

1.8339879371722013e-48

In [294]:

```
import re
from sklearn.datasets import load_digits
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

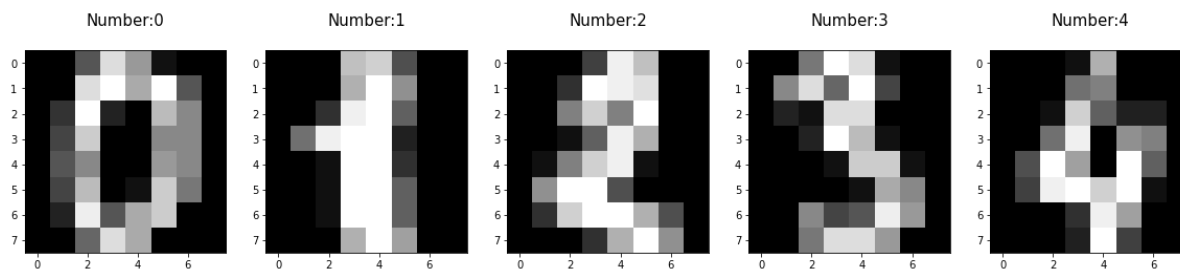
In [295]:

```
digits=load_digits()
digits
```

```
'pixel_0_4',
'pixel_0_5',
'pixel_0_6',
'pixel_0_7',
'pixel_1_0',
'pixel_1_1',
'pixel_1_2',
'pixel_1_3',
'pixel_1_4',
'pixel_1_5',
'pixel_1_6',
'pixel_1_7',
'pixel_2_0',
'pixel_2_1',
'pixel_2_2',
'pixel_2_3',
'pixel_2_4',
'pixel_2_5',
'pixel_2_6',
'pixel_2_7'
```

In [296]:

```
plt.figure(figsize=(20,4))
for index,(image,label)in enumerate(zip(digits.data[0:5],digits.target[0:5])):
    plt.subplot(1,5,index+1)
    plt.imshow(np.reshape(image,(8,8)),cmap=plt.cm.gray)
    plt.title('Number:%i\n'%label,fontsize=15)
```



In [297]:

```
x_train,x_test,y_train,y_test=train_test_split(digits.data,digits.target,test_size=0.30)
```

In [298]:

```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

(1257, 64)

(540, 64)

(1257,)

(540,)

In [299]:

```
logre=LogisticRegression(max_iter=10000)
logre.fit(x_train,y_train)
```

Out[299]:

LogisticRegression(max_iter=10000)

In [300]:

```
logre.predict(x_test)
```

Out[300]:

```
array([9, 8, 0, 2, 5, 7, 3, 4, 9, 3, 2, 8, 6, 1, 9, 9, 6, 8, 9, 0, 1, 9,
       4, 0, 7, 2, 7, 2, 9, 5, 3, 9, 1, 0, 8, 0, 6, 5, 3, 2, 8, 5, 1, 6,
       0, 4, 9, 8, 5, 0, 9, 5, 2, 4, 6, 1, 9, 7, 9, 2, 5, 8, 7, 1, 5, 6,
       2, 2, 6, 5, 7, 0, 2, 5, 3, 4, 5, 4, 1, 8, 7, 2, 7, 9, 3, 5, 7, 1,
       5, 4, 3, 7, 3, 7, 4, 9, 7, 4, 5, 2, 4, 8, 2, 2, 9, 4, 7, 6, 7, 9,
       1, 5, 0, 3, 8, 1, 3, 2, 2, 2, 9, 1, 6, 4, 6, 8, 1, 6, 7, 6, 6, 4,
       3, 9, 8, 1, 0, 9, 3, 2, 7, 6, 5, 7, 4, 1, 5, 6, 1, 7, 0, 5, 0, 7,
       7, 8, 2, 3, 2, 3, 7, 7, 2, 6, 6, 5, 5, 5, 3, 7, 5, 7, 6, 4, 9, 8,
       8, 4, 9, 1, 5, 5, 9, 4, 7, 0, 0, 7, 9, 0, 2, 4, 2, 4, 4, 8, 3, 7,
       5, 4, 4, 6, 4, 3, 8, 2, 7, 7, 1, 2, 9, 4, 5, 8, 5, 1, 3, 0, 3, 0,
       4, 6, 3, 8, 2, 1, 6, 6, 6, 8, 9, 5, 5, 2, 9, 5, 4, 8, 0, 8, 2, 7,
       4, 4, 9, 0, 8, 4, 1, 0, 5, 6, 7, 8, 6, 6, 9, 3, 4, 2, 7, 9, 9, 2,
       7, 0, 5, 9, 1, 4, 1, 1, 7, 6, 7, 0, 7, 8, 6, 8, 7, 6, 3, 8, 8, 5,
       6, 0, 1, 8, 1, 1, 1, 1, 5, 6, 6, 5, 9, 5, 2, 0, 3, 2, 6, 0, 0, 6,
       2, 8, 3, 1, 8, 0, 8, 5, 3, 8, 9, 7, 8, 6, 4, 7, 8, 8, 2, 1, 0, 7,
       3, 7, 3, 3, 5, 6, 8, 4, 4, 3, 9, 4, 2, 0, 7, 6, 8, 1, 2, 0, 2, 7,
       4, 9, 2, 9, 0, 7, 9, 8, 6, 8, 7, 2, 6, 1, 3, 4, 1, 2, 5, 4, 0, 0,
       3, 2, 7, 0, 9, 0, 9, 7, 1, 4, 3, 7, 3, 9, 6, 2, 8, 5, 9, 4, 6, 4,
       3, 7, 7, 9, 9, 6, 0, 3, 4, 4, 8, 6, 8, 8, 0, 4, 4, 0, 2, 6, 3, 8,
       6, 9, 0, 9, 0, 5, 5, 0, 7, 0, 5, 0, 3, 0, 6, 2, 6, 2, 7, 4, 3, 6,
       5, 1, 7, 9, 2, 4, 5, 8, 4, 7, 0, 4, 2, 5, 7, 8, 8, 1, 3, 6, 0, 1,
       9, 8, 8, 0, 8, 5, 5, 3, 9, 0, 8, 6, 4, 1, 5, 3, 4, 7, 4, 3, 5, 3,
       2, 9, 4, 9, 7, 3, 1, 1, 0, 9, 7, 5, 3, 8, 6, 1, 5, 8, 6, 4, 8, 0,
       2, 6, 9, 1, 8, 8, 9, 2, 4, 2, 0, 1, 0, 1, 9, 1, 1, 2, 1, 0, 6, 6,
       3, 8, 4, 5, 6, 1, 5, 8, 9, 4, 7, 6])
```

In [301]:

```
logre.score(x_test,y_test)
```

Out[301]:

```
0.9722222222222222
```

In [302]:

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

In [303]:

```
b=a[['Height', 'Weight', 'Index']]
b
```

Out[303]:

	Height	Weight	Index
0	174	96	4
1	189	87	2
2	185	110	4
3	195	104	3
4	149	61	3
5	189	104	3
6	147	92	5
7	154	111	5
8	174	90	3
9	169	103	4

In [305]:

```
b['Index'].value_counts()
```

Out[305]:

```
3    4
4    3
5    2
2    1
Name: Index, dtype: int64
```

In [306]:

```
x=b.drop('Index',axis=1)
y=b['Index']
```

In [307]:

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

	Height	Weight	Index
0	174	96	4
1	189	87	2
2	185	110	4
3	195	104	3
4	149	61	3
5	189	104	3
6	147	92	5
7	154	111	5
8	174	90	3
9	169	103	4

In [308]:

```
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 [309]:

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

In [310]:

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

Out[310]:

```
RandomForestClassifier()
```

In [311]:

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

In [312]:

```
from sklearn.model_selection import GridSearchCV
```

In [313]:

```
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:66
6: UserWarning: The least populated class in y has only 1 members, which is less
than n_splits=2.
  warnings.warn(("The least populated class in y has only %d"
```

Out[313]:

```
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 [314]:

```
grid_search.best_score_
```

Out[314]:

```
0.41666666666666663
```

In [315]:

```
rfc_best=grid_search.best_estimator_
```


In [316]:

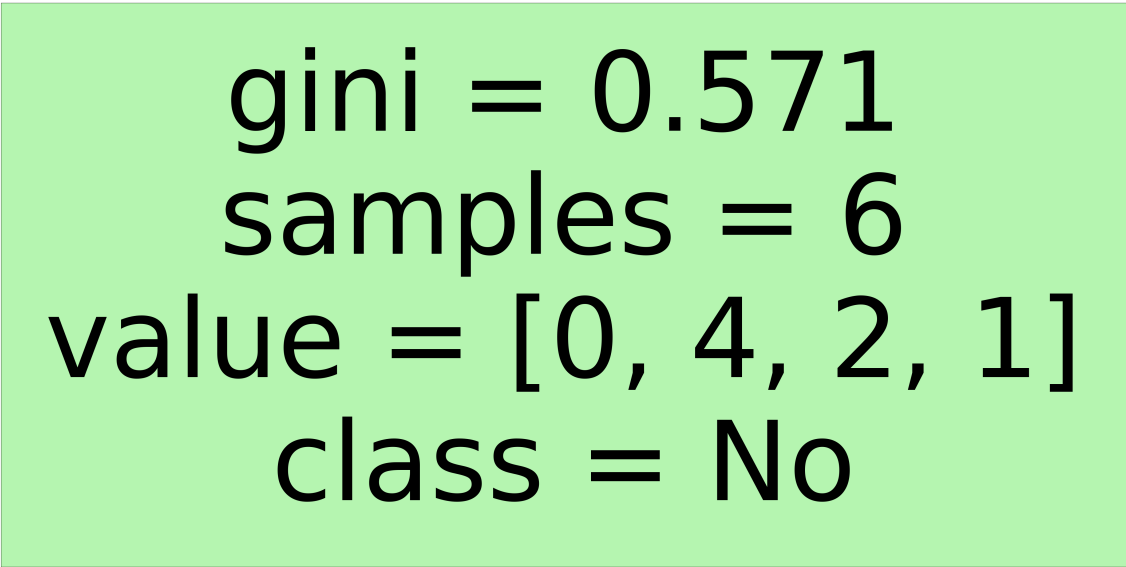
```
from sklearn.tree import plot_tree
```

In [317]:

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

Out[317]:

```
[Text(2232.0, 1087.2, 'gini = 0.571\nsamples = 6\nvalue = [0, 4, 2, 1]\nnclass =  
No')]
```



gini = 0.571
samples = 6
value = [0, 4, 2, 1]
class = No

In []: