In [31]:

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

In [70]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C4_framingham.csv")
a
```

Out[70]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	dia
0	1	39	4.0	0	0.0	0.0	0	0	
1	0	46	2.0	0	0.0	0.0	0	0	
2	1	48	1.0	1	20.0	0.0	0	0	
3	0	61	3.0	1	30.0	0.0	0	1	
4	0	46	3.0	1	23.0	0.0	0	0	
...	...	...	...	...	...	...	...	...	
4233	1	50	1.0	1	1.0	0.0	0	1	
4234	1	51	3.0	1	43.0	0.0	0	0	
4235	0	48	2.0	1	20.0	NaN	0	0	
4236	0	44	1.0	1	15.0	0.0	0	0	
4237	0	52	2.0	0	0.0	0.0	0	0	

4238 rows × 16 columns

In [71]:

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

In [77]:

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

Out[77]:

e	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totCl
l	39	4.0	0	0.0	0.0	0	0	0	19
)	46	2.0	0	0.0	0.0	0	0	0	25
l	48	1.0	1	20.0	0.0	0	0	0	24
)	61	3.0	1	30.0	0.0	0	1	0	22
)	46	3.0	1	23.0	0.0	0	0	0	28
)	43	2.0	0	0.0	0.0	0	1	0	22
)	63	1.0	0	0.0	0.0	0	0	0	20
)	45	2.0	1	20.0	0.0	0	0	0	31
l	52	1.0	0	0.0	0.0	0	1	0	26
l	43	1.0	1	30.0	0.0	0	1	0	22

In [78]:

```
c=a.iloc[:,0:16]
d=a.iloc[:,-1]
```

In [79]:

```
c.shape
```

Out[79]:

```
(10, 16)
```

In [80]:

```
d.shape
```

Out[80]:

```
(10,)
```

In [81]:

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

In [82]:

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

In [83]:

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

Out[83]:

```
LogisticRegression()
```

In [84]:

```
e=[[2,5,77,8,6,5,4,66,88,46,65,76,87,45,92,44]]
```

In [85]:

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

Out[85]:

```
array([1], dtype=int64)
```

In [86]:

```
logr.classes_
```

Out[86]:

```
array([0, 1], dtype=int64)
```

In [87]:

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

Out[87]:

```
0.0
```

In [88]:

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

Out[88]:

```
1.0
```

In [89]:

```
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
```

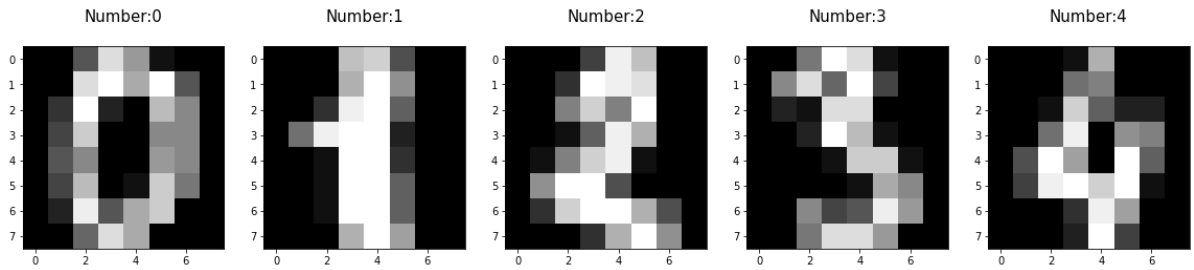
In [90]:

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

```
 'pixel_3_7',
 'pixel_4_0',
 'pixel_4_1',
 'pixel_4_2',
 'pixel_4_3',
 'pixel_4_4',
 'pixel_4_5',
 'pixel_4_6',
 'pixel_4_7',
 'pixel_5_0',
 'pixel_5_1',
 'pixel_5_2',
 'pixel_5_3',
 'pixel_5_4',
 'pixel_5_5',
 'pixel_5_6',
 'pixel_5_7',
 'pixel_6_0',
 'pixel_6_1',
```

In [91]:

```
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 [92]:

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

In [93]:

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

```
(1257, 64)
(540, 64)
(1257,)
(540,)
```

In [94]:

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

Out[94]:

```
LogisticRegression(max_iter=10000)
```

In [95]:

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

Out[95]:

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

In [96]:

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

Out[96]:

```
0.9703703703703703
```

In [97]:

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

In [109]:

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

Out[109]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabet
0	1	39	4.0	0	0.0	0.0	0	0	
1	0	46	2.0	0	0.0	0.0	0	0	
2	1	48	1.0	1	20.0	0.0	0	0	
3	0	61	3.0	1	30.0	0.0	0	1	
4	0	46	3.0	1	23.0	0.0	0	0	
5	0	43	2.0	0	0.0	0.0	0	1	
6	0	63	1.0	0	0.0	0.0	0	0	
7	0	45	2.0	1	20.0	0.0	0	0	
8	1	52	1.0	0	0.0	0.0	0	1	
9	1	43	1.0	1	30.0	0.0	0	1	

In [110]:

```
a['TenYearCHD'].value_counts()
```

Out[110]:

```
0    8
1    2
Name: TenYearCHD, dtype: int64
```

In [111]:

```
x=a.drop('TenYearCHD',axis=1)
y=a['TenYearCHD']
```

In [112]:

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

```
   male  age  education  currentSmoker  cigsPerDay  BPMeds  prevalentStroke  \
0     1   39        4.0              0         0.0     0.0                0
1     0   46        2.0              0         0.0     0.0                0
2     1   48        1.0              1        20.0     0.0                0
3     0   61        3.0              1        30.0     0.0                0
4     0   46        3.0              1        23.0     0.0                0
5     0   43        2.0              0         0.0     0.0                0
6     0   63        1.0              0         0.0     0.0                0
7     0   45        2.0              1        20.0     0.0                0
8     1   52        1.0              0         0.0     0.0                0
9     1   43        1.0              1        30.0     0.0                0

   prevalentHyp  diabetes  totChol  sysBP  diaBP    BMI  heartRate  glucose  \
0             0         0    195.0  106.0   70.0  26.97       80.0     77.0
1             0         0    250.0  121.0   81.0  28.73       95.0     76.0
2             0         0    245.0  127.5   80.0  25.34       75.0     70.0
3             1         0    225.0  150.0   95.0  28.58       65.0    103.0
4             0         0    285.0  130.0   84.0  23.10       85.0     85.0
5             1         0    228.0  180.0  110.0  30.30       77.0     99.0
6             0         0    205.0  138.0   71.0  33.11       60.0     85.0
7             0         0    313.0  100.0   71.0  21.68       79.0     78.0
8             1         0    260.0  141.5   89.0  26.36       76.0     79.0
9             1         0    225.0  162.0  107.0  23.61       93.0     88.0

   TenYearCHD
0           0
1           0
2           0
3           1
4           0
5           0
6           1
7           0
8           0
9           0
```

In [113]:

```
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 [114]:

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

In [115]:

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

Out[115]:

```
RandomForestClassifier()
```

In [116]:

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

In [117]:

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

In [118]:

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

Out[118]:

```
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 [119]:

```
grid_search.best_score_
```

Out[119]:

```
0.7083333333333333
```

In [120]:

```
rfc_best=grid_search.best_estimator_
```

In [121]:

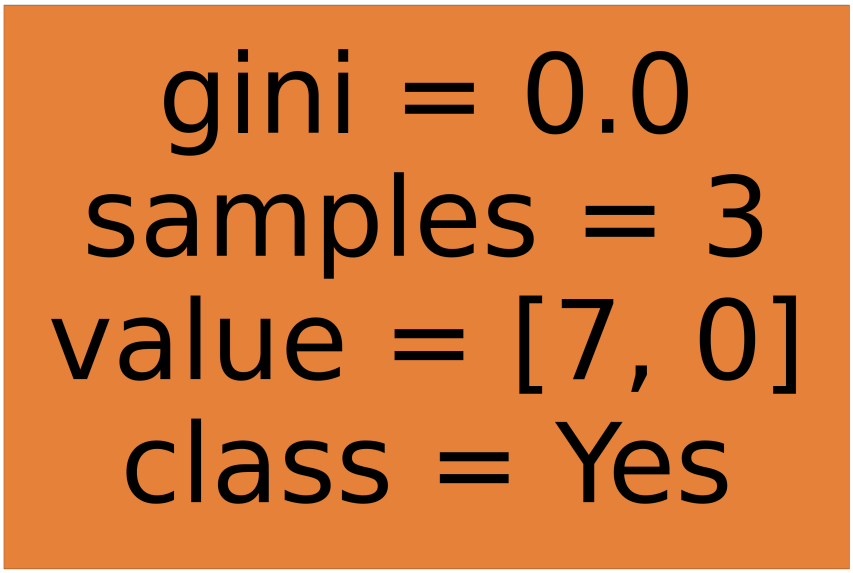
```
from sklearn.tree import plot_tree
```

In [122]:

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

Out[122]:

```
[Text(2232.0, 1087.2, 'gini = 0.0\nsamples = 3\nvalue = [7, 0]\nclass = Yes')]
```



In [ ]: