In [31]:

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

In [123]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C5_health care diabetes.csv")
a
```

Out[123]:

	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
...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171
764	2	122	70	27	0	36.8	0.340
765	5	121	72	23	112	26.2	0.245
766	1	126	60	0	0	30.1	0.349
767	1	93	70	31	0	30.4	0.315

768 rows × 9 columns

In [124]:

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

In [125]:

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

Out[125]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Ag
0	6	148	72	35	0	33.6	0.627	5
1	1	85	66	29	0	26.6	0.351	3
2	8	183	64	0	0	23.3	0.672	3
3	1	89	66	23	94	28.1	0.167	2
4	0	137	40	35	168	43.1	2.288	3
5	5	116	74	0	0	25.6	0.201	3
6	3	78	50	32	88	31.0	0.248	2
7	10	115	0	0	0	35.3	0.134	2
8	2	197	70	45	543	30.5	0.158	5
9	8	125	96	0	0	0.0	0.232	5

In [137]:

```
c=a.iloc[:,0:9]
d=a.iloc[:,-1]
```

In [138]:

```
c.shape
```

Out[138]:

```
(10, 9)
```

In [139]:

```
d.shape
```

Out[139]:

```
(10,)
```

In [140]:

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

In [141]:

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

In [142]:

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

Out[142]:

```
LogisticRegression()
```

In [144]:

```
e=[[2,5,77,8,6,5,4,66,88]]
```

In [145]:

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

Out[145]:

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

In [146]:

```
logr.classes_
```

Out[146]:

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

In [147]:

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

Out[147]:

```
0.0
```

In [148]:

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

Out[148]:

```
1.0
```

In [149]:

```
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 [150]:

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

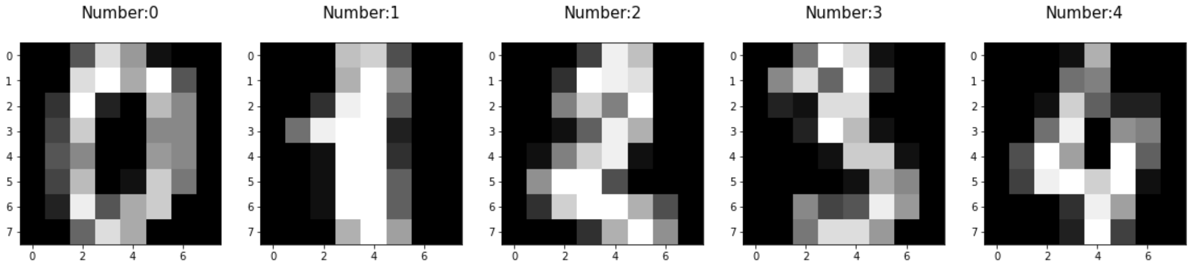
```
         [ 0.,  4., 11., ..., 12.,  7.,  0.],
         [ 0.,  2., 14., ..., 12.,  0.,  0.],
         [ 0.,  0.,  6., ...,  0.,  0.,  0.]],

        [[ 0.,  0.,  0., ...,  5.,  0.,  0.],
         [ 0.,  0.,  0., ...,  9.,  0.,  0.],
         [ 0.,  0.,  3., ...,  6.,  0.,  0.],
         ...,
         [ 0.,  0.,  1., ...,  6.,  0.,  0.],
         [ 0.,  0.,  1., ...,  6.,  0.,  0.],
         [ 0.,  0.,  0., ..., 10.,  0.,  0.]],

        [[ 0.,  0.,  0., ..., 12.,  0.,  0.],
         [ 0.,  0.,  3., ..., 14.,  0.,  0.],
         [ 0.,  0.,  8., ..., 16.,  0.,  0.],
         ...,
         [ 0.,  9., 16., ...,  0.,  0.,  0.],
         [ 0.,  3., 13., ..., 11.,  5.,  0.],
         [ 0.,  0.,  0., ..., 16.,  9.,  0.]],
```

In [151]:

```
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 [152]:

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

In [153]:

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

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

In [154]:

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

Out[154]:

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

In [155]:

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

Out[155]:

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

In [156]:

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

Out[156]:

```
0.9481481481481482
```

In [157]:

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

In [158]:

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

Out[158]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Ag
0	6	148	72	35	0	33.6	0.627	5
1	1	85	66	29	0	26.6	0.351	3
2	8	183	64	0	0	23.3	0.672	3
3	1	89	66	23	94	28.1	0.167	2
4	0	137	40	35	168	43.1	2.288	3
5	5	116	74	0	0	25.6	0.201	3
6	3	78	50	32	88	31.0	0.248	2
7	10	115	0	0	0	35.3	0.134	2
8	2	197	70	45	543	30.5	0.158	5
9	8	125	96	0	0	0.0	0.232	5

In [172]:

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

Out[172]:

```
1    6
0    4
Name: Outcome, dtype: int64
```

In [173]:

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

In [161]:

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

```
   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   
5            5      116             74              0        0  25.6   
6            3       78             50             32       88  31.0   
7           10      115              0              0        0  35.3   
8            2      197             70             45      543  30.5   
9            8      125             96              0        0   0.0   

   DiabetesPedigreeFunction  Age  Outcome  
0                     0.627   50        1  
1                     0.351   31        0  
2                     0.672   32        1  
3                     0.167   21        0  
4                     2.288   33        1  
5                     0.201   30        0  
6                     0.248   26        1  
7                     0.134   29        0  
8                     0.158   53        1  
9                     0.232   54        1  
```

In [162]:

```
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 [163]:

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

In [164]:

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

Out[164]:

```
RandomForestClassifier()
```

In [165]:

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

In [166]:

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

In [167]:

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

Out[167]:

```
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 [168]:

```
grid_search.best_score_
```

Out[168]:

```
0.7083333333333333
```

In [169]:

```
rfc_best=grid_search.best_estimator_
```

In [170]:

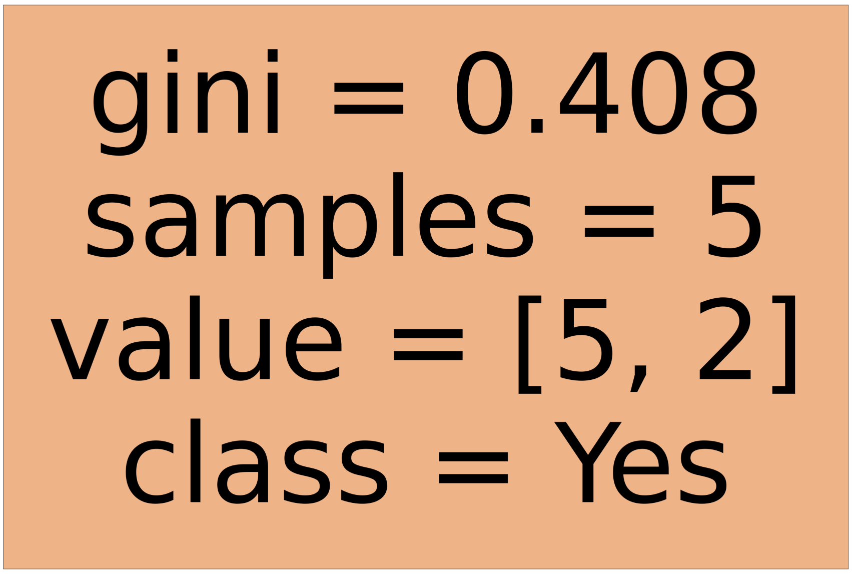
```
from sklearn.tree import plot_tree
```

In [171]:

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

Out[171]:

```
[Text(2232.0, 1087.2, 'gini = 0.408\nsamples = 5\nvalue = [5, 2]\nclass = Yes')]
```



In [ ]:

```

```