**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

**from** sklearn.preprocessing **import** StandardScaler

In [3]:

df **=** pd**.**read\_csv(r"C:\Users\user\Desktop\C5\_health care diabetes (1).csv")

df

Out[3]:

|  | **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 [4]:

df**.**head()

Out[4]:

|  | **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 |

In [5]:

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 [6]:

df**.**describe()

Out[6]:

|  | **Pregnancies** | **Glucose** | **BloodPressure** | **SkinThickness** | **Insulin** | **BMI** | **DiabetesPedigreeFunction** | **Age** | **Outcome** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **count** | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 |
| **mean** | 3.845052 | 120.894531 | 69.105469 | 20.536458 | 79.799479 | 31.992578 | 0.471876 | 33.240885 | 0.348958 |
| **std** | 3.369578 | 31.972618 | 19.355807 | 15.952218 | 115.244002 | 7.884160 | 0.331329 | 11.760232 | 0.476951 |
| **min** | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.078000 | 21.000000 | 0.000000 |
| **25%** | 1.000000 | 99.000000 | 62.000000 | 0.000000 | 0.000000 | 27.300000 | 0.243750 | 24.000000 | 0.000000 |
| **50%** | 3.000000 | 117.000000 | 72.000000 | 23.000000 | 30.500000 | 32.000000 | 0.372500 | 29.000000 | 0.000000 |
| **75%** | 6.000000 | 140.250000 | 80.000000 | 32.000000 | 127.250000 | 36.600000 | 0.626250 | 41.000000 | 1.000000 |
| **max** | 17.000000 | 199.000000 | 122.000000 | 99.000000 | 846.000000 | 67.100000 | 2.420000 | 81.000000 | 1.000000 |

In [7]:

df**.**columns

Out[7]:

Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',

'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],

dtype='object')

In [8]:

feature\_matrix **=** df**.**iloc[:,0:34]

target\_vector **=** df**.**iloc[:,**-**1]

In [9]:

fs **=** StandardScaler()**.**fit\_transform(feature\_matrix)

logr **=** LogisticRegression()

logr**.**fit(fs,target\_vector)

Out[9]:

LogisticRegression()

In [14]:

observation**=**[[1,2,3,4,5,6,7,8,9]]

prediction **=** logr**.**predict(observation)

print(prediction)

[1]

In [15]:

logr**.**classes\_

Out[15]:

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

In [16]:

logr**.**predict\_proba(observation)

Out[16]:

array([[0., 1.]])

In [18]:

x **=** df**.**iloc[:,0:34]

y**=** df**.**iloc[:,**-**1]

In [19]:

**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 [20]:

**from** sklearn.ensemble **import** RandomForestClassifier

rfc **=** RandomForestClassifier()

rfc**.**fit(x\_train,y\_train)

Out[20]:

RandomForestClassifier()

In [21]:

parameters **=** {'max\_depth':[1,2,3,4,5],'min\_samples\_leaf':[5,10,15,20,25],

'n\_estimators': [10,20,30,40,50]

}

In [22]:

**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[22]:

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 [23]:

grid\_search**.**best\_score\_

Out[23]:

1.0

In [24]:

rfc\_best **=** grid\_search**.**best\_estimator\_

In [25]:

**from** sklearn.tree **import** plot\_tree

plt**.**figure(figsize**=**(89,40))

plot\_tree(rfc\_best**.**estimators\_[5], feature\_names**=**x**.**columns, class\_names**=**['Yes','No'], filled**=True**)

Out[25]:

[Text(2483.1, 1630.8000000000002, 'BMI <= 31.05\ngini = 0.466\nsamples = 334\nvalue = [339, 198]\nclass = Yes'),

Text(1241.55, 543.5999999999999, 'gini = 0.27\nsamples = 150\nvalue = [209, 40]\nclass = Yes'),

Text(3724.6499999999996, 543.5999999999999, 'gini = 0.495\nsamples = 184\nvalue = [130, 158]\nclass = No')]

In [ ]:

