In [1]:
```
from IPython.display import Image
Image(filename='logo.PNG', height=340, width=900)
```

Out[1]:

In [2]:
```
# Load libraries
import numpy as np
import pandas as pd
```

In [3]:
```
pima = pd.read_csv("diabetes.csv")
```

In [4]:
```
pima.head()
```

Out[4]:

	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

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction
4	0	137	40	35	168	43.1	2.288

In [5]:

```
pima.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
Pregnancies                 768 non-null int64
Glucose                     768 non-null int64
BloodPressure               768 non-null int64
SkinThickness               768 non-null int64
Insulin                     768 non-null int64
BMI                         768 non-null float64
DiabetesPedigreeFunction    768 non-null float64
Age                         768 non-null int64
Outcome                     768 non-null int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [6]:

```
pima.describe()
```

Out[6]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesI
count	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	
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	

In [7]:

```
X = pima.iloc[:,:-1].values
y = pima.iloc[:,-1].values
```

In [8]:

```
from sklearn.model_selection import train_test_split # Import train_tes
t_split function
```

In [9]:

```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3
, random_state=1) # 70% training and 30% test
```

In [10]:

```
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree
 Classifier
```

In [11]:

```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)
y_pred
```

Out[11]:

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

In [12]:

```
from sklearn import metrics #Import scikit-learn metrics module for acc
uracy calculation
```

In [13]:

```
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",round(metrics.accuracy_score(y_test, y_pred)*100,2),
"%")
```

```
Accuracy: 71.0 %
```

Well, you got a classification rate of 71.0%, considered as good accuracy.

In [14]:

```
from sklearn.metrics import confusion_matrix
```

In [15]:

```
cm=confusion_matrix(y_test,y_pred)
```

In [16]:

```
cm
```

Out[16]:

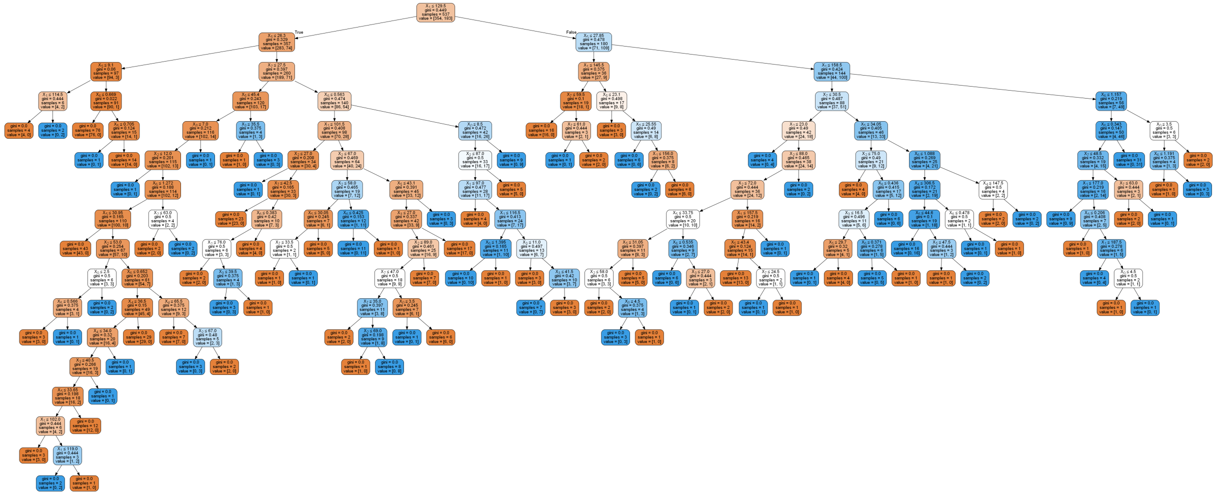
```
array([[120,  26],
       [ 41,  44]], dtype=int64)
```

In [19]:

```
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
                filled=True, rounded=True,
                        special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png(), width=12000, height=12000)
```

```
C:\Users\Admin\Anaconda3\lib\site-packages\sklearn\externals\six.py:31:
DeprecationWarning: The module is deprecated in version 0.21 and will b
e removed in version 0.23 since we've dropped support for Python 2.7. P
lease rely on the official version of six (https://pypi.org/project/si
x/).
  "(https://pypi.org/project/six/).", DeprecationWarning)
```

Out[19]:



In [ ]: