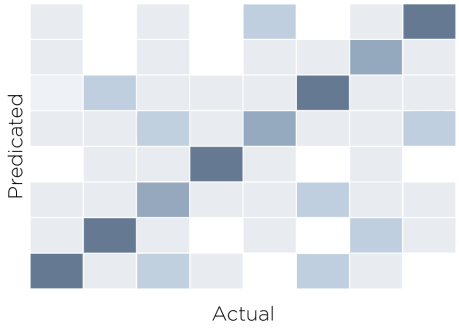
PRO-C118

WhiteHat Jr
Live Online Coding for Kids

Decision Tree



What is our GOAL for this MODULE?

The goal of this module is to explore the concept of Decision Tree.

What did we ACHIEVE in the class TODAY?

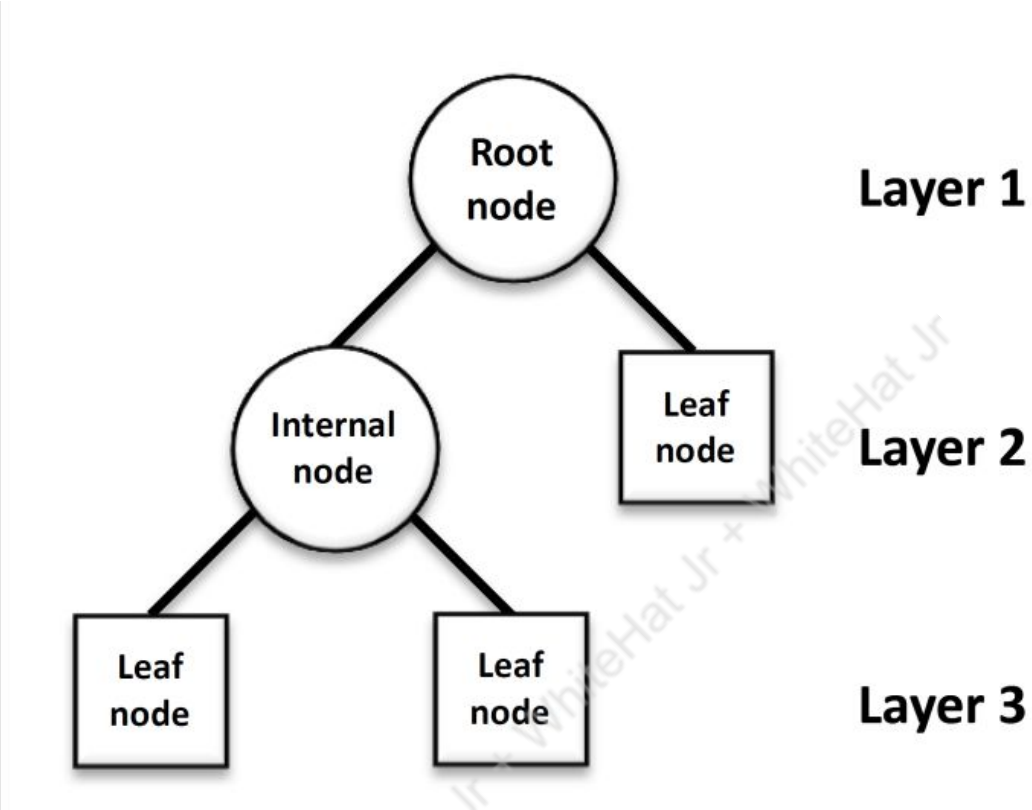
We explored the concept of Decision Tree , wrote our own Decision Tree algorithm and created the charts for same..

Which CONCEPTS/CODING BLOCKS did we cover today?

- Decision Tree algorithm
- Usage of **pydotplus** module to turn the data into chart images.

How did we DO the activities?

1. We explored the concept of Decision Tree .
    - We saw the different nodes of the Decision Tree.



2. We used the data of diabetes patients for our algorithm.
3. We uploaded the data and then created the data frames of it.

```
] #Uploading the csv
  from google.colab import files
  data_to_load = files.upload()
```

```
import pandas as pd

#Column Name
col_names = ['pregnant', 'glucose', 'bp', 'skin', 'insulin', 'bmi', 'pedigree', 'age', 'label']

df = pd.read_csv("diabetes.csv", names=col_names).iloc[1:]

print(df.head())
```

```
  pregnant glucose  bp skin insulin   bmi pedigree age label
1        6     148  72   35       0  33.6    0.627  50     1
2        1      85  66   29       0  26.6    0.351  31     0
3        8     183  64    0       0  23.3    0.672  32     1
4        1      89  66   23      94  28.1    0.167  21     0
5        0     137  40   35     168  43.1    2.288  33     1
```

```
features = ['pregnant', 'insulin', 'bmi', 'age','glucose','bp','pedigree']
X = df[features]
y = df.label
```

4. Then we split the data to train , test and then fit it into the model.

```
[ ] from sklearn.tree import DecisionTreeClassifier
    from sklearn.model_selection import train_test_split
    from sklearn import metrics

    #splitting data in training and testing
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)

    #Initialising the Decision Tree Model
    clf = DecisionTreeClassifier()

    #Fitting the data into the model
    clf = clf.fit(X_train,y_train)

    #Calculating the accuracy of the model
    y_pred = clf.predict(X_test)
    print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.6666666666666666
```

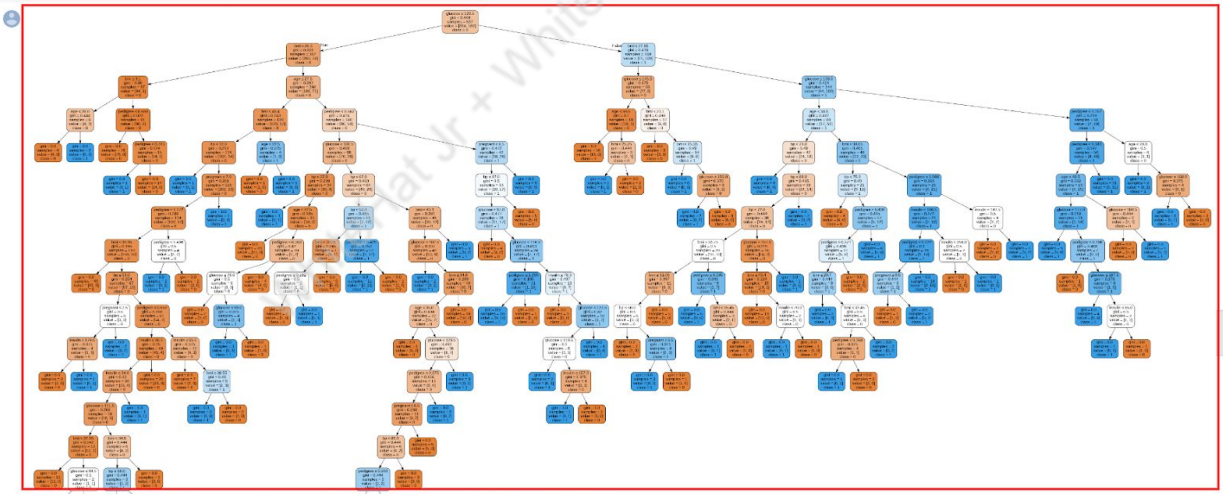
5. Using the graph_viz function we created the representation of the data.

```
[ ]
from sklearn.tree import export_graphviz
from sklearn.externals.six import StringIO
from IPython.display import Image
import pydotplus

dot_data = StringIO() #Where we will store the data from our decision tree classifier as text.

export_graphviz(clf, out_file=dot_data, filled=True, rounded=True, special_characters=True, feature_names=features, class_names=['0','1'])

print(dot_data.getvalue())
```

```
digraph Tree {
node [shape=box, style="filled, rounded", color="black", fontname=helvetica] ;
edge [fontname=helvetica] ;
0 [label=<glucose &le; 129.5<br/>gini = 0.449<br/>samples = 537<br/>value = [354, 183]<br/>class = 0>, fillcolor="#f2c29f"] ;
1 [label=<bmi &le; 26.3<br/>gini = 0.329<br/>samples = 357<br/>value = [283, 74]<br/>class = 0>, fillcolor="#eca26d"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label=<bmi &le; 9.1<br/>gini = 0.06<br/>samples = 97<br/>value = [94, 3]<br/>class = 0>, fillcolor="#e6853f"] ;
1 -> 2 ;
3 [label=<age &le; 28.0<br/>gini = 0.444<br/>samples = 6<br/>value = [4, 2]<br/>class = 0>, fillcolor="#f2c09c"] ;
2 -> 3 ;
4 [label=<gini = 0.0<br/>samples = 4<br/>value = [4, 0]<br/>class = 0>, fillcolor="#e58139"] ;
3 -> 4 ;
5 [label=<gini = 0.0<br/>samples = 2<br/>value = [0, 2]<br/>class = 1>, fillcolor="#399de5"] ;
3 -> 5 ;
6 [label=<pedigree &le; 0.669<br/>gini = 0.022<br/>samples = 91<br/>value = [90, 1]<br/>class = 0>, fillcolor="#e5823b"] ;
2 -> 6 ;
7 [label=<gini = 0.0<br/>samples = 76<br/>value = [76, 0]<br/>class = 0>, fillcolor="#e58139"] ;
6 -> 7 ;
8 [label=<pedigree &le; 0.705<br/>gini = 0.124<br/>samples = 15<br/>value = [14, 1]<br/>class = 0>, fillcolor="#e78a47"] ;
6 -> 8 ;
9 [label=<gini = 0.0<br/>samples = 1<br/>value = [0, 1]<br/>class = 1>, fillcolor="#399de5"] ;
8 -> 9 ;
10 [label=<gini = 0.0<br/>samples = 14<br/>value = [14, 0]<br/>class = 0>, fillcolor="#e58139"] ;
8 -> 10 ;
11 [label=<age &le; 27.5<br/>gini = 0.397<br/>samples = 260<br/>value = [189, 71]<br/>class = 0>, fillcolor="#efb083"] ;
1 -> 11 ;
12 [label=<bmi &le; 45.4<br/>gini = 0.243<br/>samples = 120<br/>value = [103, 17]<br/>class = 0>, fillcolor="#e9965a"] ;
11 -> 12 ;
13 [label=<bp &le; 12.0<br/>gini = 0.212<br/>samples = 116<br/>value = [102, 14]<br/>class = 0>, fillcolor="#e99254"] ;
12 -> 13 ;
14 [label=<gini = 0.0<br/>samples = 1<br/>value = [0, 1]<br/>class = 1>, fillcolor="#399de5"] ;
```

6. Using pydotplus we created the chart image of the data.

```
[ ]
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
Image(graph.create_png())
```

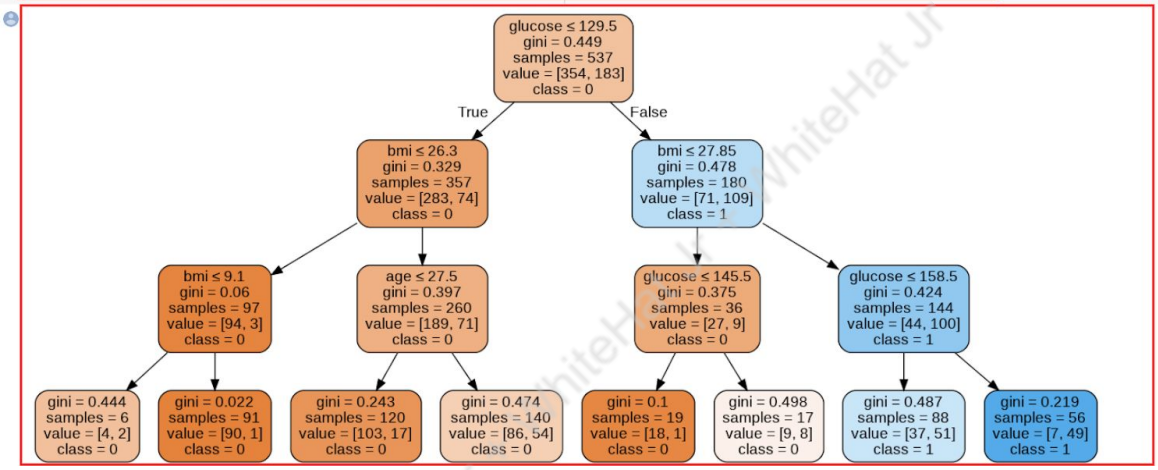


7. We trimmed the data and then again created the image chart of that data.

```
[ ]  clf = DecisionTreeClassifier(max_depth=3)

     clf = clf.fit(X_train,y_train)

     y_pred = clf.predict(X_test)
     print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.7575757575757576
```

```
dot_data = StringIO() #Where we will store the data from our decision tree classifier as text.

export_graphviz(clf, out_file=dot_data, filled=True, rounded=True, special_characters=True, feature_names=features, class_names=['0','1'])

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
Image(graph.create_png())
```



WE concluded that almost 75% accuracy that a person who's

Glucose is greater than 129.5 and,
BMI is greater than 27.85
Is more prone to be a Diabetes Patient.

What's NEXT?

In the next class, we will explore more of machine learning.

EXTEND YOUR KNOWLEDGE:

Learn more about the decision tree from the following link:
https://scikit-learn.org/stable/modules/tree.html