Machine Learning

**Ex:10 Decision Tree**

**Name:** Athithraja. R

**Reg.no:** 2022503702

**Aim:**

Create a Decision tree using ID3 algorithm using sklearn, and print the decision tree and confusion matrix for PIMA indian diabeties dataset

**Code:**

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier, plot\_tree

import matplotlib.pyplot as plt

from sklearn.metrics import confusion\_matrix, classification\_report

import numpy as np

import pandas as pd

data=pd.read\_excel('diabetes\_0.xlsx')

#print(data)

x=data.drop('Outcome',axis=1)

y=data['Outcome']

#print(x)

#print(y)

x\_train, x\_test, y\_train, y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=42)

model=DecisionTreeClassifier(criterion='entropy',random\_state=42,max\_depth=3)

model.fit(x\_train,y\_train)

y\_pred=model.predict(x\_test)

plt.figure(figsize=(15,10))

plot\_tree(model, filled=True,

feature\_names=x.columns.tolist(),

class\_names=['0','1'],rounded=True,

fontsize=10,impurity=False,

proportion=False,

)

plt.title("Decision Tree for PIMA Indian Diabetes Dataset")

plt.show()

cm=confusion\_matrix(y\_test,y\_pred)

print("Confusion Matrix: \n"+str(cm))

print("\nClassification Report:\n"+str(classification\_report(y\_test,y\_pred)))

def entropy(y):

class\_counts=np.bincount(y)

probabilities=class\_counts/len(y)

return -np.sum(probabilities\*np.log2(probabilities+1e-9))

def information\_gain(x,y,feature):

total\_entropy=entropy(y)

unique\_values=np.unique(x[feature])

weighted\_entropy=0

for value in unique\_values:

subset=y[x[feature]==value]

weighted\_entropy+=len(subset)/len(y)\*entropy(subset)

return total\_entropy-weighted\_entropy

info\_gains={feature:information\_gain(x,y,feature)for feature in x.columns}

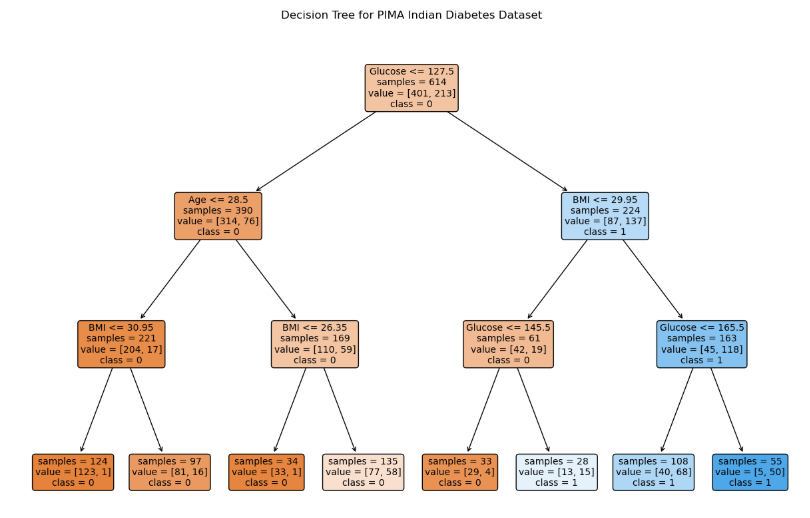
print("Information Gain: \n")

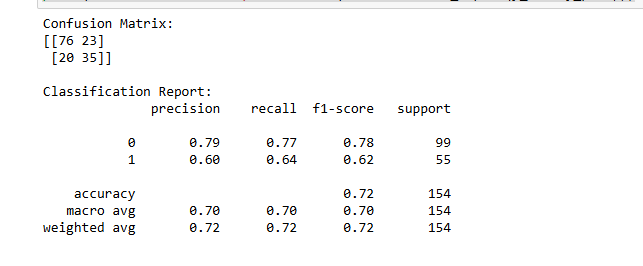
for feature,gain in info\_gains.items():

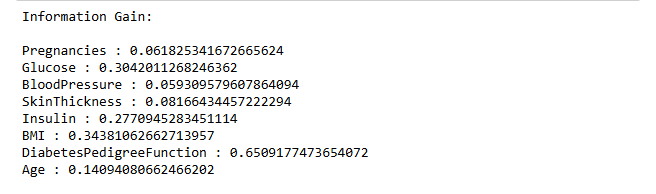
print(str(feature)+" : "+str(gain))

**Output:**









2) Play Tennis or Not

**Code:**

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier, plot\_tree

from sklearn.preprocessing import LabelEncoder

import matplotlib.pyplot as plt

from sklearn.metrics import confusion\_matrix, classification\_report

import numpy as np

import pandas as pd

data=pd.read\_excel('tennis.xlsx')

label\_encoder = LabelEncoder()

data['Outlook'] = label\_encoder.fit\_transform(data['Outlook'])

data['Temperature'] = label\_encoder.fit\_transform(data['Temperature'])

data['Humidity'] = label\_encoder.fit\_transform(data['Humidity'])

data['Wind'] = label\_encoder.fit\_transform(data['Wind'])

data['Play Tennis'] = label\_encoder.fit\_transform(data['Play Tennis'])

#print(data)

x=data.drop('Play Tennis',axis=1)

y=data['Play Tennis']

x\_train, x\_test, y\_train, y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=42)

model=DecisionTreeClassifier(criterion='entropy',random\_state=42,max\_depth=3)

model.fit(x\_train,y\_train)

y\_pred=model.predict(x\_test)

plt.figure(figsize=(15,10))

plot\_tree(model, filled=True,

feature\_names=x.columns.tolist(),

class\_names=['0','1'],rounded=True,

fontsize=10,impurity=False,

proportion=False,

)

plt.title("Decision Tree for Play Tennis Dataset")

plt.show()

cm=confusion\_matrix(y\_test,y\_pred)

print("Confusion Matrix: \n"+str(cm))

print("\nClassification Report:\n"+str(classification\_report(y\_test,y\_pred)))

def entropy(y):

class\_counts=np.bincount(y)

probabilities=class\_counts/len(y)

return -np.sum(probabilities\*np.log2(probabilities+1e-9))

def information\_gain(x,y,feature):

total\_entropy=entropy(y)

unique\_values=np.unique(x[feature])

weighted\_entropy=0

for value in unique\_values:

subset=y[x[feature]==value]

weighted\_entropy+=len(subset)/len(y)\*entropy(subset)

return total\_entropy-weighted\_entropy

info\_gains={feature:information\_gain(x,y,feature)for feature in x.columns}

print("Information Gain: \n")

for feature,gain in info\_gains.items():

print(str(feature)+" : "+str(gain))

**Output:**

