import pandas as pd

import numpy as np

from sklearn.datasets import load\_iris

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

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score

#visualise the decision tree

from sklearn.tree import export\_graphviz

from six import StringIO

from IPython.display import Image

import pydotplus

#load iris data

data = load\_iris()

print(data.data.shape)

print('classes to predict: ',data.target\_names)

print('Features: ',data.feature\_names)

X = data.data

y = data.target

print(display (X.shape, y.shape))

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y,random\_state = 50, test\_size = 0.25)

#default criterion is Gini

classifier = DecisionTreeClassifier()

classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test)

print('Accuracy on train data using Gini: ',accuracy\_score(y\_true = y\_train, y\_pred = classifier.predict(X\_train)))

print('Accuracy on test data using Gini: ',accuracy\_score(y\_true = y\_test, y\_pred = y\_pred))

#change criterion to entropy

classifier\_entropy = DecisionTreeClassifier(criterion='entropy')

classifier\_entropy.fit(X\_train, y\_train)

y\_pred\_entropy = classifier\_entropy.predict(X\_test)

print('Accuracy on train data using entropy', accuracy\_score(y\_true=y\_train, y\_pred = classifier\_entropy.predict(X\_train)))

print('Accuracy on test data using entropy', accuracy\_score(y\_true=y\_test, y\_pred = y\_pred\_entropy))

#change criterion to entropy with min\_samples\_split to 50. Default value is 2

classifier\_entropy1 = DecisionTreeClassifier(criterion='entropy', min\_samples\_split=50)

classifier\_entropy1.fit(X\_train, y\_train)

y\_pred\_entropy1 = classifier\_entropy1.predict(X\_test)

print('Accuracy on train data using entropy', accuracy\_score(y\_true=y\_train, y\_pred = classifier\_entropy1.predict(X\_train)))

print('Accuracy on test data using entropy', accuracy\_score(y\_true=y\_test, y\_pred = y\_pred\_entropy1))

#visualise the decision tree

#visualise the decision tree

dot\_data = StringIO()

#the students can try using classifier, classifier\_entropy and classifier\_entropy1

#as first parameter below.

export\_graphviz(classifier, out\_file = dot\_data,filled = True, rounded = True,special\_characters = True, feature\_names = data.feature\_names, class\_names = data.target\_names)

graph = pydotplus.graph\_from\_dot\_data(dot\_data.getvalue())

Image(graph.create\_png())



