Question 1

Modified code for Including all 4 features:

% Demonstrate classification/decision tree on 2d 3class iris data

% From http://www.mathworks.com/products/statistics/demos.html?file=/products/demos/shipping/stats/classdemo.html

requireStatsToolbox

% "Resubstitution error" is the training set error

load fisheriris

N = size(meas,1);

figure

gscatter(meas(:,1), meas(:,2), species,'rgb','osd');

xlabel('Sepal length');

ylabel('Sepal width');

printPmtkFigure('dtreeIrisData','pdf','D:\EE 660\HW 12')

s = RandStream('mt19937ar','seed',0);

RandStream.setGlobalStream(s);

cp = cvpartition(species,'k',10);

% fit tree

t = classregtree(meas(:,1:4), species,'names',{'SL' 'SW' 'PL' 'PW' });

% plot tree

view(t)

%printPmtkFigure('dtreeTreeUnpruned')

% Error rate

dtclass = t.eval(meas(:,1:4));

bad = ~strcmp(dtclass,species);

dtResubErr = sum(bad) / N

dtClassFun = @(xtrain,ytrain,xtest)(eval(classregtree(xtrain,ytrain),xtest));

dtCVErr = crossval('mcr',meas(:,1:4),species, ...

'predfun', dtClassFun,'partition',cp)

% Plot misclassified data

figure;

gscatter(meas(:,1), meas(:,2), species,'rgb','osd');

xlabel('Sepal length');

ylabel('Sepal width');

hold on;

plot(meas(bad,1), meas(bad,2), 'kx', 'markersize', 10, 'linewidth', 2);

title(sprintf('Unpruned, train error %5.3f, cv error %5.3f', dtResubErr, dtCVErr))

printPmtkFigure('dtreeDataUnpruned','pdf','D:\EE 660\HW 12')

% Error rate vs depth

figure;

resubcost = test(t,'resub');

[cost,secost,ntermnodes,bestlevel] = test(t,'cross',meas(:,1:4),species);

plot(ntermnodes,cost,'b-', ntermnodes,resubcost,'r--','linewidth',3)

figure(gcf);

xlabel('Number of terminal nodes');

ylabel('Cost (misclassification error)')

[mincost,minloc] = min(cost);

cutoff = mincost + secost(minloc);

hold on

plot([0 20], [cutoff cutoff], 'k:', 'linewidth', 3)

plot(ntermnodes(bestlevel+1), cost(bestlevel+1), 'mo', 'markersize', 12, 'linewidth', 2)

legend('Cross-validation','Training set','Min + 1 std. err.','Best choice')

printPmtkFigure('dtreeErrorVsDepth','pdf','D:\EE 660\HW 12')

Question 3

Code is in Python

# -\*- coding: utf-8 -\*-

"""

Created on Wed Nov 14 15:44:27 2018

@author: tchat

"""

import csv

import numpy as np

import pandas as pd

from scipy.spatial.distance import cdist

from sklearn.utils import resample

from sklearn.utils import shuffle

import math

from sklearn.decomposition import PCA

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

from sklearn.ensemble import RandomForestClassifier

from sklearn.datasets import make\_classification

from sklearn.metrics import accuracy\_score

import numpy as np

import matplotlib.pyplot as plt

x\_train = pd.read\_csv(r'D:\EE 660\HW 12\x\_train.csv')

y\_train = pd.read\_csv(r'D:\EE 660\HW 12\y\_train.csv',header=None)

y\_train = np.ravel(y\_train)

x\_test = pd.read\_csv(r'D:\EE 660\HW 12\x\_test.csv')

y\_test = pd.read\_csv(r'D:\EE 660\HW 12\y\_test.csv',header=None)

y\_test = np.ravel(y\_test)

acc\_test = np.zeros((10,30))

acc\_train = np.zeros((10,30))

for b in range(1,31):

for x in range(1,10):

X\_train\_bagged, X\_test\_bagged, y\_train\_bagged, y\_test\_bagged = train\_test\_split(x\_train, y\_train, train\_size=0.333)

clf = RandomForestClassifier(n\_estimators=b, max\_features=3, bootstrap=True)

clf.fit(X\_train\_bagged, y\_train\_bagged)

y\_pred\_train = clf.predict(X\_train\_bagged)

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

acc\_test[x-1][b-1] = accuracy\_score(y\_pred\_test,y\_test)

acc\_train[x-1][b-1] = accuracy\_score(y\_pred\_train,y\_train\_bagged)

mean\_accur\_test = acc\_test.mean(0)

mean\_accur\_train = acc\_train.mean(0)

std\_accur\_test = acc\_test.std(0)

std\_accur\_train = acc\_train.std(0)

print(mean\_accur\_test)

print(mean\_accur\_train)

print(acc\_test.std(0))

print(acc\_train.std(0))

plt.figure(1)

plt.plot(1-mean\_accur\_train, 'bo',label='mean\_train\_accuracy')

plt.legend(loc='best')

plt.title('mean\_train\_accuracy vs no of trees')

plt.xlabel('No. of trees')

plt.ylabel('Value of accuracy')

plt.figure(2)

plt.plot(1-mean\_accur\_test, 'ro',label='mean\_test\_accuracy')

plt.legend(loc='best')

plt.title('mean\_test\_accuracy vs no of trees')

plt.xlabel('No. of trees')

plt.ylabel('Value of accuracy')

plt.figure(3)

plt.plot(std\_accur\_test,'go',label='std\_test\_accuracy')

plt.legend(loc='best')

plt.title('std\_test\_accuracy vs no of trees')

plt.xlabel('No. of trees')

plt.ylabel('Value of accuracy')

plt.figure(4)

plt.plot(std\_accur\_train,'co',label='std train accuracy')

plt.legend(loc='best')

plt.title('std train accuracy vs no of trees')

plt.xlabel('No. of trees')

plt.ylabel('Value of accuracy')

plt.show()