CSE 5524 – Homework #11

11/07/2013

Manjari Akella

1). Write an implementation of the k-Nearest Neighbors (kNN) algorithm to classify data points. Use the points in file train.txt as training data (this file contains 1 row for each data point where the first two columns are x-y coordinates and the third column is the ground truth classification). Classify all the test data points in the file test.txt (formatted in the same way). Calculate the accuracy of your algorithm (compared to the third column of the test data). Plot the test data points, color coded by the class your algorithm gives each (use plot() options 'r.' and 'b.'). On the same figure (use hold on/off), (re)plot the points which are misclassified (use plot() option 'ko'). Repeat this for K=1, 5, 11, and 15.

Compare the results for different values of K. Do you notice a pattern in the misclassified points? [3 pts] (The Matlab function knnsearch() can be used for part of the problem.)

- From Table 1, we can see that the accuracy remains nearly same for all values of k
- Maximum Accuracy was observed when k=15
- In general, having a higher k will give more robustness to noise but making it too large will defeat the purpose of the knn algorithm
- As k increases, knn tends to become a majority classifier. That is to say that in this case, if k=1000, this would basically assign the new test point the label of the majority class of the training set
- The misclassified points in all cases lie along the decision boundary between classes. This makes sense because the chance of misclassification of points decreases as we go deeper into a class region (farther away from the decision boundary)

Output

K	Accuracy
1	96.766667%
5	96.333333%
11	96.266667%
15	96.900000%

Table 1: Accuracy Percentage for different k

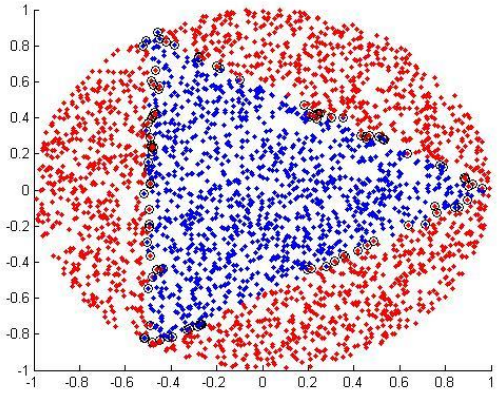

Figure 1: k=1

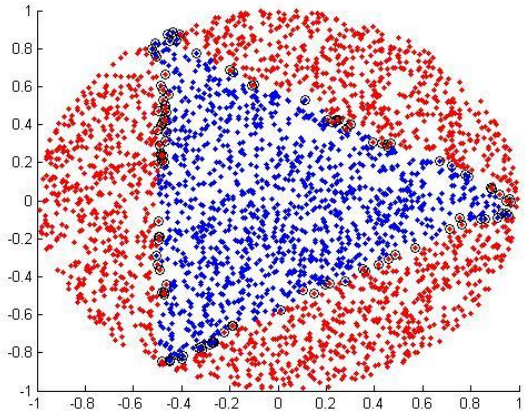

Figure 2: k=5

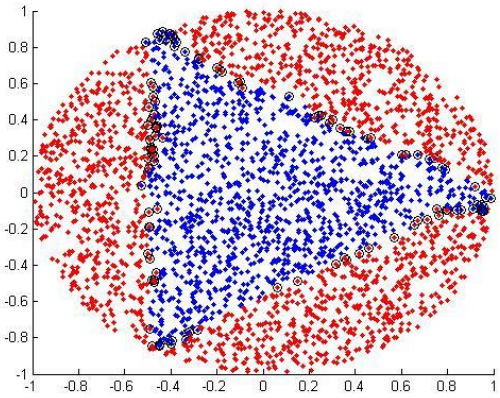

Figure 3: k=11

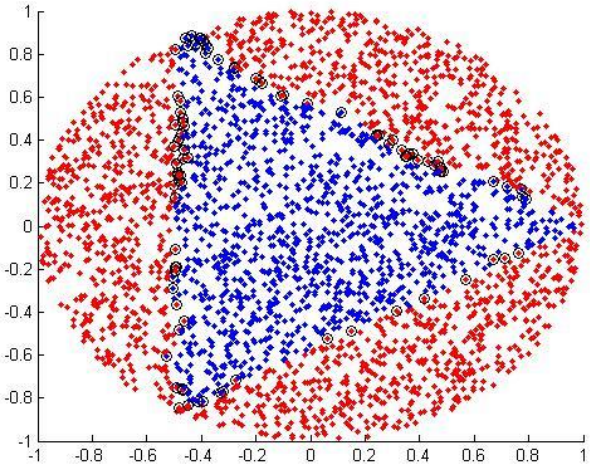

Figure 4: k=15

2). Repeat Problem 1, but using a decision tree classifier. Use functions built into Matlab. Try different pruning levels, L=0,2,9,10, and 11.

- The knn classifier gives better results for all values of k which were tested
- The accuracy of classification by the d-tree decreases significantly as L in increased (except for L=0 to L=2 which *marginally* increases)
- Further, a d-tree separates classes by iteratively drawing a horizontal or vertical decision boundary at each level. This is the reason for the triangular misclassification artifacts in Figures 5-9(most prominent in Figure 5)

Output

L	Accuracy
0	95.5000%
2	95.6000%
9	88.8667%
10	85.6333%
11	70.3333%

Table 2: Accuracy Percentage for different L

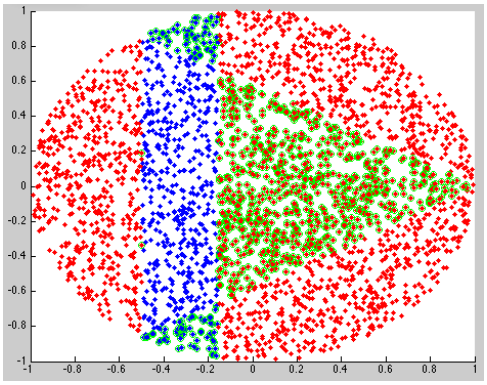

Figure 5: L=11

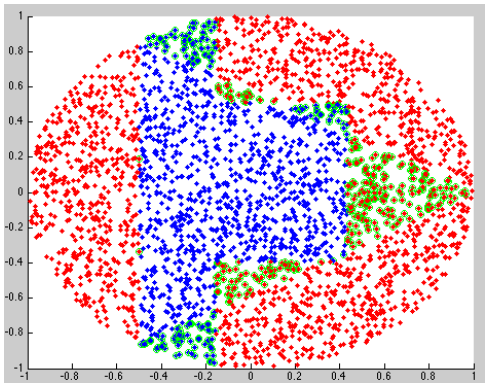

Figure 6: L=10

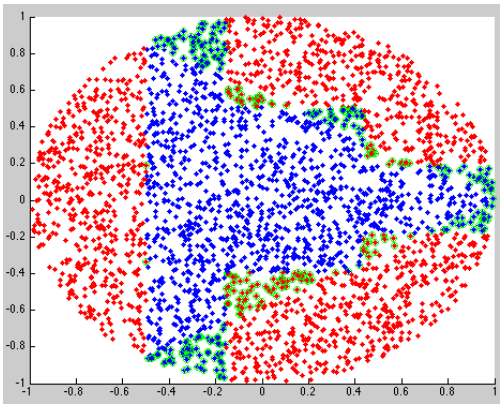

Figure 7: L=9

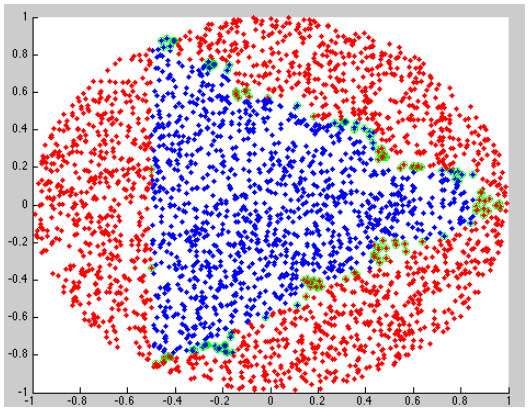

Figure 8: L=2

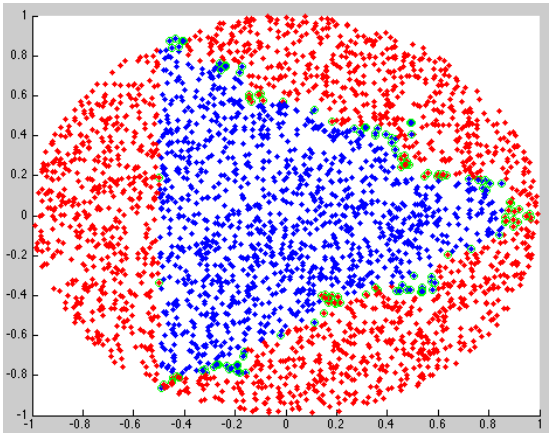

Figure 9: L=0

CODE

1).HW11.m

```
% Manjari Akella
% CSE5524 - HW11
% 11/07/2013

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 1
clear all;
close all;
clc;
% Define k
k=[1,5,11,15];
% Load train and test files
tr = load('given_data/train.txt');
te = load('given_data/test.txt');
% Define number of training and testing instances
Ntr = size(tr,1);
Nte = size(te,1);
fprintf('KNN\n');
% Loop for each k
for itr=1:size(k,2)
    % For classes, to count no. of misclassifications, distance
    count = 0;
    dist = [];
    class = [];
    for i=1:Nte
        for j=1:Ntr
            % Euclidian Distance
            dist(1,j) = sqrt(sum(((te(i,1:2)-tr(j,1:2)).^2),2));
        end
        % Sort in ascending order
        [sdist,idx] = sort(dist,'ascend');
        % Nearest k neighbours indices
        idx = idx(1:k(itr));
        % Get classification label of nearest neighbours
        nearest = tr(idx,3);
        % Assign mode of labels of nearest neighbours to the test point
        class(i,1) = mode(nearest);
    end
    figure('Name',strcat('Q1: k= ',num2str(k(itr))),'NumberTitle','off');
    hold on;
    for i=1:Nte
            % Plot the points according to class label
            if(class(i,1)==1)
                plot(te(i,1),te(i,2),'b.');
            else
                plot(te(i,1),te(i,2),'r.');
            end

            % (Re)-plot misclassified points
            if(class(i,1)~=te(i,3))
                plot(te(i,1),te(i,2),'ko');
                count = count+1;
```

```
            end
    end
    hold off;
    accuracy = (Nte-count)/Nte;
    fprintf('k=%d,accuracy=%f\n',k(itr),accuracy);
end
pause;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 2
clear all;
close all;
clc;
% Load train and test files and define test data size
tr = load('given_data/train.txt');
te = load('given_data/test.txt');
Nte = size(te,1);
% Level of pruning
L = [0,2,9,10,11];
fprintf('D-TREE\n');
% Loop for each L
for itr=1:size(L,2)
    tree = ClassificationTree.fit(tr(:,1:2),tr(:,3));
    tprune = prune(tree,'level',L(itr));
    class = predict(tprune,te(:,1:2));
    view(tprune,'mode','graph');
    figure('Name',strcat('Q2: L= ',num2str(L(itr))),'NumberTitle','off');
    hold on;
    % To count number of misclassified instances
    count=0;
    for i=1:Nte
            % Plot the points according to class label
            if(class(i,1)==1)
                plot(te(i,1),te(i,2),'b.');
            else
                plot(te(i,1),te(i,2),'r.');
            end

            % (Re)-plot misclassified points
            if(class(i,1)~=te(i,3))
                plot(te(i,1),te(i,2),'go');
                count = count+1;
            end
    end
    hold off;
    accuracy = (Nte-count)/Nte;
    fprintf('L=%d,accuracy=%f\n',L(itr),accuracy);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%% END %%%%%%%%%%%%%%%%%%%%%%
```