RS322: Pattern Recognition
Programming Assignment 2

Atma Anand
SC12B156

Date: April 10, 2015 Department: Physical Sciences

Aim: To project the given data using Principal Component Analysis (PCA) and classify it using k-Nearest Neighbour Estimator (in MATLAB).

Data Set Used: *Iris Data* – 3 classes, 4 parameters

Theory and Procedure:

(Also refer to the comments written alongside the code)

1. Input the given data and append the original class of the samples to the data.
2. Find the eigenvectors and eigenvalues of the mean subtracted data. This is done easily by the svd command.
3. Project the data along 'n' (1<=n<=4) directions is descending order of eigenvalues, using the eigenvectors corresponding to those eigenvalues. 'n' is input by user.
4. For generality, randomly create two subsets of the data: first the training set and second the test set, for the algorithm. The sets should be non-overlapping (disjoint) and contain equal representation from all classes.
5. Find the 'k' nearest neighbours of each class from the training data for each test sample. This is done by maintaining an ascending order sorted array and comparing the distance of the test sample from each training data set. We assume each parameter to have equal weightage to the distance.
6. As the total no. of samples and 'k' of each class is the same, the classification is determined by the farthest point of each class (V). The data is classified to the class having the smallest value of distance for its furthest point in the k-NN. In case of a tie the data may be classified to any of the tied class. 'k' is input by the user.
7. The confusion matrix is prepared and the accuracy of the classification is found out. Confusion matrix states the classified and misclassified test samples in a matrix representation.

The overall algorithm is very simple to implement and execute.

Code: Classifying Iris Data Using PCA and k-NN Estimator

```
% Reading the Iris data
data1 = xlsread('D:\Workspace\Pattern_Recognition\Iris3\irisdata.xls');
class = ones(150,1);
class(51:100) = 2;
class(101:150) = 3;
data = [data1 class];

% Applying PCA on the given data
md = data(:,1:4);
M = mean(md);
for i=1:150
    md(i,:) = md(i,:) - M;
end
[U,S,V] = svd(md);
% V is the matrix having eigenvectors as columns
n = input('Enter the Dimension of projection (1-4): ');
e = V(1:4,1:n);
a = md*e;
x = mean(1:n) + a*e(1:n,:);

% Randomly selecting 25 Training data points from each of the 3 Classes
train = zeros(1,75);
for i=1:3
    k=1;
    while k <= 25
        r = (i-1)*50 + randi(50,1);
        f=0;
        for l=(i-1)*25+1:i*25
            if(train(l)==r)
                f=1;
                break;
            end
        end
        if f==0
            train((i-1)*25+k)=r;
            k=k+1;
        end
    end
end
% Randomly selecting 15 Test samples from each of the 3 Classes
% (non-overlapping with the Training Set)
test = zeros(1,45);
for i=1:3
    k=1;
    while k <= 15
        r = (i-1)*50 + randi(50,1);
        f=0;
        for l=(i-1)*15+1:i*15
            if(train(l)==r || test(l)==r)
                f=1;
                break;
            end
        end
        for l=(i-1)*25+16:i*25
            if(train(l)==r)
```

```
                f=1;
                break;
            end
        end
        if f==0
            test((i-1)*15+k)=r;
            k=k+1;
        end
    end
end

k = input('Enter value of k for k-NN Classifier: ');
res = [zeros(1,45);zeros(1,45)];
for i=1:45
    res(1,i)=data(test(i),5);
    dist = 100*ones(1,3*k);
    pos = ones(3*k);
    % Computing the k Nearest Neighbours of each Class
    for z=1:3
        for l=(z-1)*25+1:z*25
            d = sum((x(train(l),1:n)- x(test(i),1:n)).^2); %distance
            for m=(z-1)*k+1:z*k
                if d<dist(m)
                    for q=z*k:-1:(z-1)*25+m+1
                        dist(q)=dist(q-1);
                        pos(q)=pos(q-1);
                    end
                    dist(m)=d;
                    pos(m)=train(l);
                    break;
                end
            end
        end
    end
    % Classification
    if dist(k)<dist(2*k) && dist(k)<dist(3*k)
        res(2,i)=1;
    else if dist(2*k)<dist(k) && dist(2*k)<dist(3*k)
            res(2,i)=2;
        else
            res(2,i)=3;
        end
    end
end

% Making the Confusion Matrix
conf = zeros(3,3);
for i=1:45
    conf(res(1,i),res(2,i)) = conf(res(1,i),res(2,i))+1;
end
display('The Confusion Matrix is:');
display(conf);
acc = trace(conf)/45*100; %accuracy
display(sprintf('Classification Accuracy = %g', acc));
```

Output:

Sample Run: (Different for each Execution)

1. Enter the Dimension of projection (1-4): 2
   Enter value of k for k-NN Classifier: 3
   The Confusion Matrix is:
   conf =
   15 0 0
   0 15 0
   0 1 14
   Classification Accuracy = 97.7778

2. Enter the Dimension of projection (1-4): 2
   Enter value of k for k-NN Classifier: 2
   The Confusion Matrix is:
   conf =
   15 0 0
   0 14 1
   0 1 14

   Classification Accuracy = 95.5556

3. Enter the Dimension of projection (1-4): 3
   Enter value of k for k-NN Classifier: 4
   The Confusion Matrix is:
   conf =
   15 0 0
   0 15 0
   0 0 15
   Classification Accuracy = 100

Result:

The given data set was classified into the 3 classes using all four parameters with an accuracy of above 95% using PCA and k-NN Estimator.