MACHINE LEARNING WITH NETWORKS

ECEN – 765

SAMBANDH BHUSAN DHAL

UIN 726006080

PROGRAMMING ASSIGNMENT - 1

Question 1:
MATLAB Code for visualizing the data.

```
for kvar=0:9
    filename = strcat(int2str(kvar),'_22.txt')
    text = fileread(filename); // reads the filename
    A=zeros(32,34);
    for i=1:32
        for j=1:34
            A(i,j)=str2double(text(((i-1)*34)+j));
        end
    end
    B=zeros(32,32);
    for j=1:32
        B(:,j)=A(:,j);
    end

    imwrite(B,strcat(int2str(kvar),'_22.png'))
end
```

Outputs of the code:

MATLAB Code to convert data to vectors

This function converts all the training data and testing data to vectors whenever called in the program.

```
function [data,number] = converttovector(foldername)
List=dir(fullfile(strcat(pwd,'\',foldername,'\*.txt')));// makes a list of
data=zeros(length(List),1024);                          all the files
number=zeros(length(List),1);
for i=1:length(List)
 tempo=List(i).name;
 io_contents = fullfile(pwd,foldername,tempo);
 text = fileread(io_contents);

 A=zeros(32,34);
 for j=1:32
 for k=1:34
 A(j,k)=str2double(text(((j-1)*34)+k));
 end
end
```

```
data1=zeros(32,32);
for l=1:32
data1(:,l)=A(:,l);
end

data2=reshape(data1',[1,1024]);


for m=1:1024
data(i,m)=data2(m);
end

tempname=List(i).name(1);
number(i,:)=str2double(tempname);
end
```

Question 2 and 3:

In this question, we use MATLAB to implement our model. The training data is used to train both the Naïve Bayes and KNN classifiers where the value of K for the KNN classifier varies from 1 to 10.

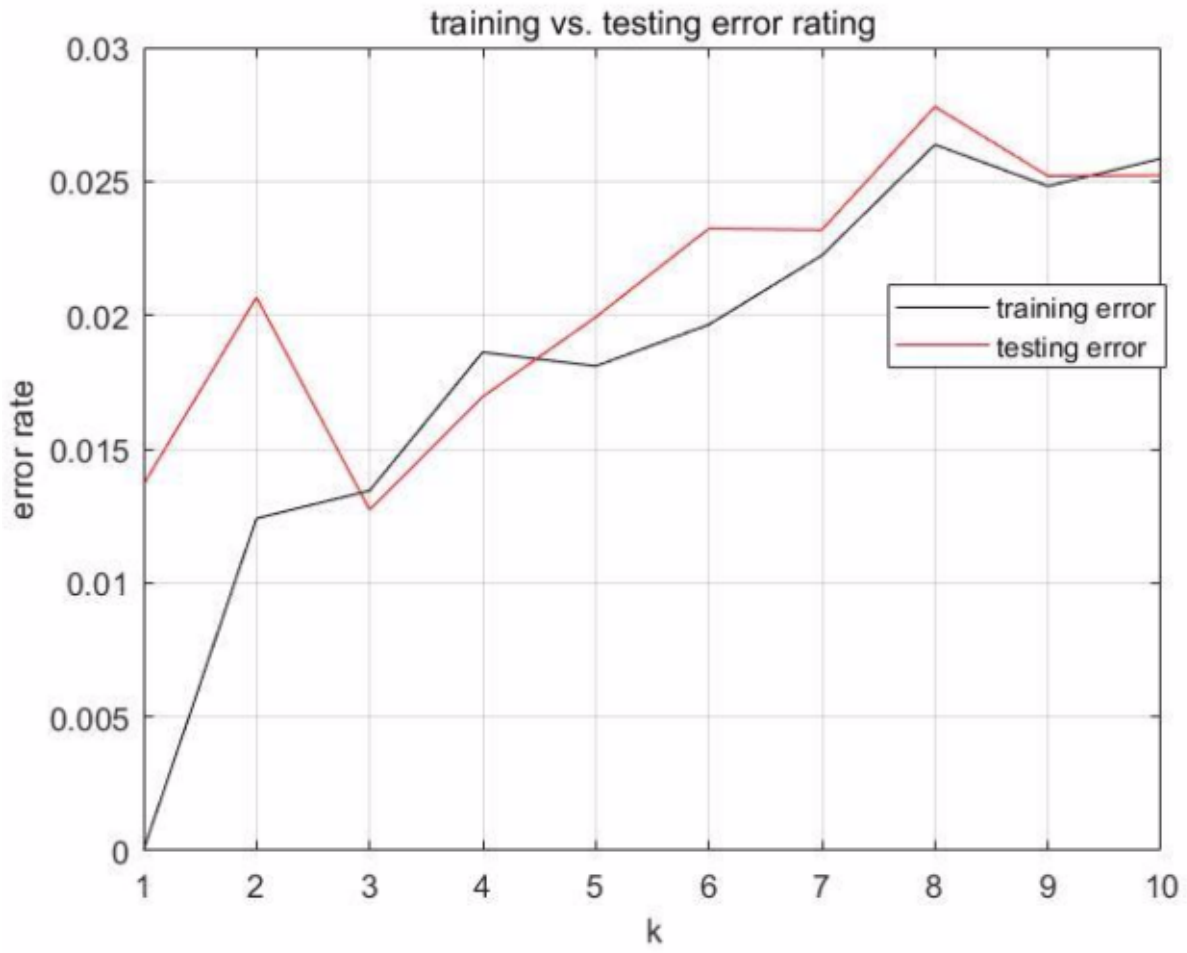
The code outputs the no of errors and the error ratio for each classifier along with the plot for the KNN classifier.

MATLAB Code to run KNN and Naïve Bayes classifier without PCA

```
clc;
clear all;
close all;
[training_data,training_number] = converttovector('trainingDigits');
X=training_data;
Y=training_number;
[test_data,test_number] = converttovector('testDigits');
Z=test_data;
N=test_number;
%KNN Classifier for k=1 to 10.
%Varying k for both testing data and training data and finding the number of
errors.
for l=1:10
 Mdl1 = fitcknn(X,Y,'NumNeighbors',l);

 label1 = predict(Mdl1,Z);
 [number1(l),ratio1(l)] = biterr(N,label1);
```

```
 label2 = predict(Mdl1,X);
 [number2(l),ratio2(l)] = biterr(Y,label2);
end
%Plotting KNN-classifier results from k=1 to 10.
k=1:10;
figure
grid on
plot(k,ratio1);
hold on
plot(k,ratio2);
legend('Training error','Testing error','location','best');
grid;
xlabel('K');
ylabel('Error Rate');
title('training vs testing error rating');
%Naive Bayes Classifier implementation and calculation of error percentage
Mdl2 = fitcnb(X,Y,'Distribution','kernel');
label3 = predict(Mdl2,Z);
[number3,ratio3] = biterr(N,label3);
label4 = predict(Mdl2,X);
[number4,ratio4] = biterr(Y,label4);
```



For Naïve Bayes Classifier the following table gives the error rates.

Data Used	Error Rate
On Testing Data	0.0523
On Training Data	0.0518

KNN ERROR RATE for K = 1 to 10

Testing Data -> number of errors and error rate

K=1
the total number of errors is: 12
the total error rate is: 0.014685
K=2
the total number of errors is: 13
the total error rate is: 0.013742
K=3
the total number of errors is: 13
the total error rate is: 0.013742
K=4
the total number of errors is: 14
the total error rate is: 0.014799
K=5
the total number of errors is: 17
the total error rate is: 0.017970
K=6
the total number of errors is: 18
the total error rate is: 0.019027
K=7
the total number of errors is: 20
the total error rate is: 0.021142
K=8
the total number of errors is: 24
the total error rate is: 0.025370
K=9
the total number of errors is: 23
the total error rate is: 0.024313
K=10
the total number of errors is: 20
the total error rate is: 0.021142

Training Data -> number of errors and error rate

K=1
the total number of errors is: 0
the total error rate is: 0.000000
K=2

```
the total number of errors is: 24
the total error rate is: 0.012410
K=3
the total number of errors is: 25
the total error rate is: 0.012927
K=4
the total number of errors is: 28
the total error rate is: 0.014478
K=5
the total number of errors is: 32
the total error rate is: 0.016546
K=6
the total number of errors is: 33
the total error rate is: 0.017063
K=7
the total number of errors is: 37
the total error rate is: 0.019131
K=8
the total number of errors is: 41
the total error rate is: 0.021200
K=9
the total number of errors is: 46
the total error rate is: 0.023785
K=10
the total number of errors is: 47
the total error rate is: 0.024302
```

Observations:

For KNN algorithm:

a) In general, when KNN algorithm is used, it can be observed from the plots that as the value of 'k' increases from 1 to 10 the value of error rate increases from approximately 1% to 10% for both testing and training data sets.
b) For testing data set, a minimal error is obtained approximately for both k=1 and 3.
c) For training data set, a minimal error is obtained for k=1.
d) For both training and test data sets, error rates increase as one moves from k=1 to 10.

For Naïve Bayes Classifier:

a) For Naïve Bayes classifier, an error rate of approximately 5% is obtained for both training and testing data sets.
b) Naive Bayes classifier assumes that each class is distributed according to a simple distribution, independent on feature basis. For continuous case - It will fit a radial Normal distribution to your whole class and then make a decision through it.
KNN on the other hand is not a probabilistic model. It assumes that we are referring to simply a "smooth" version of the original idea, where we return ratio of each class in the nearest neighbors set. This assumes nothing about data distribution besides being locally smooth. In particular - it is a nonparametric model which, given enough training samples, will fit perfectly to any dataset. Naive Bayes will fit perfectly only to K-Gaussians where K is number of classes.

c) The performance of Naive Bayes depends on the conditional probability and independence. It also depends on the type of distribution used (Gaussian by default). Its performance can be improved by choosing the distributions that best characterizes our data and prediction problem.

Question 4:

In this question, we apply Principal Component Analysis on the input data using a PCA sub-function and we feed the output of the PCA sub-function to the Naïve Bayes and KNN classifiers to calculate the error rate.

MATLAB Code to run KNN and Naïve Bayes Classifier with Principal Component Analysis

```
clc;
clear all;
close all;
%%Converting to Training data into Vector format.
[training_data,training_number] = converttovector('trainingDigits');
X=training_data;
Y=training_number;
%% Converting to Test data into vector format.
[test_data,test_number] = converttovector('testDigits');
Z=test_data;
N=test_number;
%Applying Principal Component Analysis on training and testing data.
[Xmodified] = pcadataarranger(X);
[Zmodified] = pcadataarranger(Z);
%Varying k for testing data
%KNN Classifier for k=1 to 10.
%Varying k for both testing data and training data and finding the number of
errors.
for l=1:10
 Mdl1 = fitcknn(Xmodified,Y,'NumNeighbors',l);

 label1 = predict(Mdl1,Z);
 [number1(l),ratio1(l)] = biterr(N,label1);

 label2 = predict(Mdl1,X);
 [number2(l),ratio2(l)] = biterr(Y,label2);
end

%Plotting KNN-classifier results from k=1 to 10.
k=1:10;
figure
grid on
plot(k,ratio1);
```

```
hold on
plot(k,ratio2);
legend('KNN on Test Data','KNN on Training Data','location','best');
grid;
xlabel('K');
ylabel('Error Rate');
title('training vs. testing error rating after pca process');
%Naive Bayes Classifier implementation and calculation of error percentage
Mdl2 = fitcnb(Xmodified,Y,'Distribution','kernel');
label3 = predict(Mdl2,Z);
[number3,ratio3] = biterr(test_number,label3);
label4 = predict(Mdl2,X);
[number4,ratio4] = biterr(Y,label4);
```

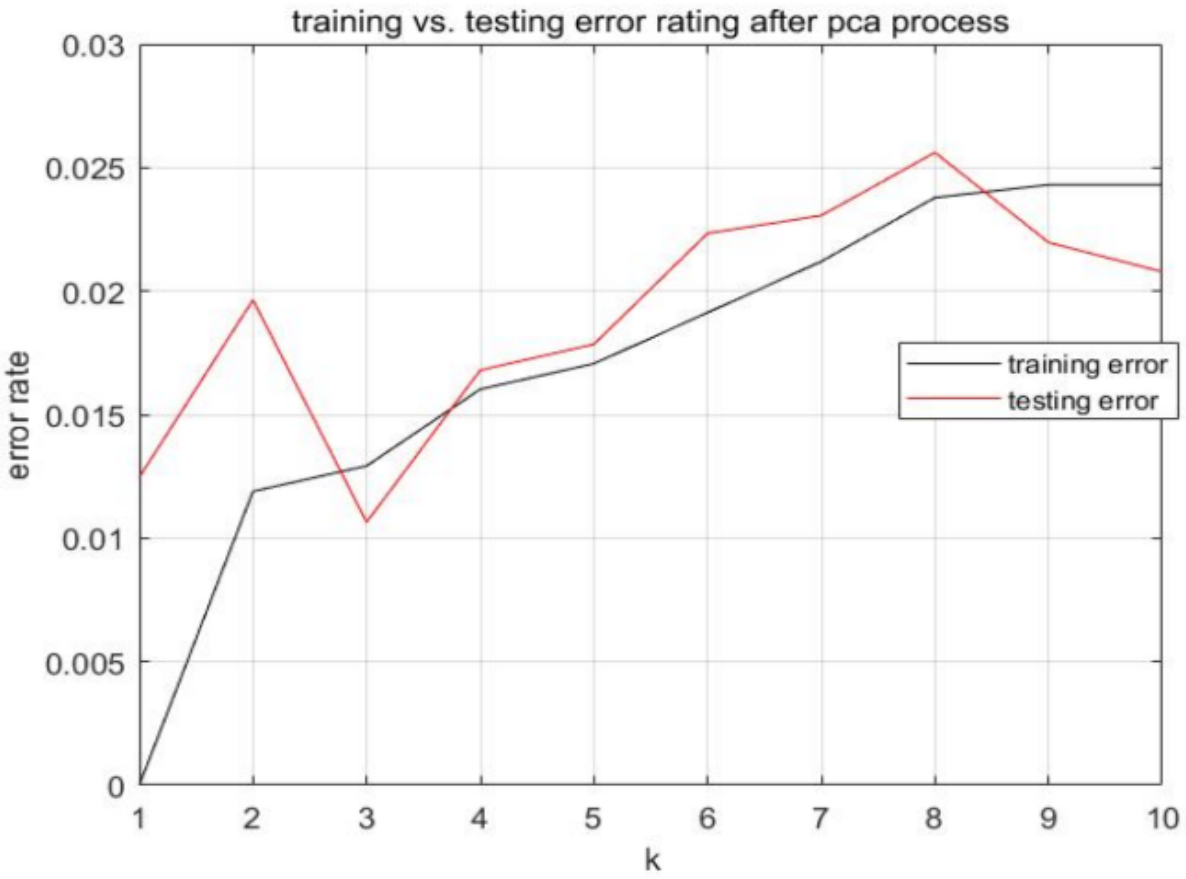
%PCA Data arranger Sub-function

This code applies PCA to the training/testing digits which are then applied on the Naïve Bayes and KNN algorithms and then the corresponding error rates are calculated.

```
function [Xmodified] = pcadataarranger(X)
[coeff,score,latent,tsquared,explained] = pca(X);
comp=cumsum(latent)./sum(latent);
threshold=0.999;
comp2=[];
for i=1:length(comp)
 if comp(i)>threshold
 comp2=[comp2 comp(i)];
 else
 end
end
comp2=comp2';
[Ybrml,Ebrml,Lbrml,mbrml,Xmodified]=pca(X,length(comp2));
```

```
function [Y,E,L,m,Xmodified]=pca(X,varargin)
%PCA Principal Components Analysis
% [Y,E,L,m,Xmodified]=pca(X,<M>,<usemean>,<return only principal solution>)
% Inputs:
% X : each column of matrix X contains a datapoint
% M : the number of principal components to retain (if missing M=size(X,1))
% usemean : set this to 1 to use a mean for the reconstruction
% return only principal solution : set to 1 to return only the M leading eigenvectors and eigenvalues
% Outputs:
% Y : coefficients
% E : eigenvectors
% L : eigenvalues
% m : mean of the data
% Xmodified : reconstructions using M components
if isempty(varargin)
   M=size(X,1);
else
   M=varargin{1};
end
if nargin>=3; usemean=varargin{2}; else usemean=1; end
N=size(X,2);
```

```
m = usemean*mean(X,2);
X0 = X - repmat(m,1,N); % centre the data
[E D]=svd(X0,0);
L = (diag(D).^2)/(N-1);
if M<N
   Y = E(:,1:M)'*X0;
   Xmodified = E(:,1:M)*Y + repmat(m,1,N);
else
   Y = E'*X0;
   Xmodified = E*Y + repmat(m,1,N);
end
if nargin==4; if varargin{3}; E=E(:,1:M); L=L(1:M); end; end
```



When PCA Algorithm is used, for Naïve Bayes Classifier the following table gives the error rates.

Data Used	Error Rate
On Testing Data	0.0887
On Training Data	0.0049

KNN ERROR RATE for K = 1 to 10

Testing Data -> number of errors and error rate

K=1
the total error rate is: 0.0123
K=2
the total error rate is: 0.0197
K=3
the total error rate is: 0.0112
K=4
the total error rate is: 0.0169
K=5
the total error rate is: 0.0179
K=6
the total error rate is: 0.0223
K=7
the total error rate is: 0.0231
K=8
the total error rate is: 0.0251
K=9
the total error rate is: 0.0221
K=10
the total error rate is: 0.0208

Training Data -> number of errors and error rate

K=1
the total error rate is: 0.000000
K=2
the total error rate is: 0.0119
K=3
the total error rate is: 0.0128
K=4
the total error rate is: 0.0162
K=5
the total error rate is: 0.0171
K=6
the total error rate is: 0.0193
K=7
the total error rate is: 0.0215
K=8
the total error rate is: 0.0241
K=9
the total error rate is: 0.0243
K=10
the total error rate is: 0.0244

Observations (For KNN and Naïve Bayes algorithms with PCA):

a) The error rates of KNN and NB classifiers with PCA are plotted above. For KNN, the error rate increases from approximately 1% to 10% as one moves k from 1 to 10. For NB, an error rate of approximately 5 to 10% is obtained.

b) PCA is a way of identifying patterns in data, and expressing the data in such a way as to highlight their similarities and differences. Since patterns in data can be hard to find in data of high dimension, where the luxury of graphical representation is not available, PCA is a powerful tool for analyzing data.

   The other main advantage of PCA is that once we have found the patterns in the data, we can compress the data, i.e. by reducing the number of dimensions, without much loss of information.

   However, PCA assumes that the dimensionality of data can be efficiently reduced by linear transformation and most information is contained in those directions where input data variance is maximum. However, these conditions are by no means always met. The second disadvantage of the PCA consists in the fact that the directions maximizing variance do not always maximize information.