MATLAB Report

Student ID: 10554466

Contents

Question 1- K-Means Clustering

1)

Code Used:

```
rows = size(x,1);
```

Output:

```
Rows:
        2028
```

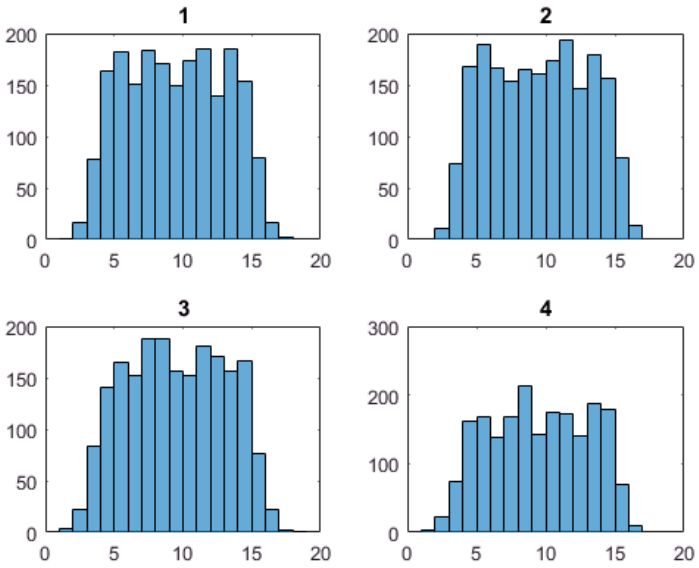
2)

Code Used:

```
columns = size(x,2);
figure();
for c = 1:columns
    temp = x(:,c);
    M = mean(temp);
    fprintf('Mean of column %d is %d\n',c,M);
    sd = std(temp);
    fprintf('Standard dev of column %d is %d\n',c,sd);
    subplot(2,2,c);
    histogram(temp);
    title(c);
end
```

Output:

```
Mean of column 1 is 9.467046e+00
Standard dev of column 1 is 3.532551e+00
Mean of column 2 is 9.491005e+00
Standard dev of column 2 is 3.511140e+00
Mean of column 3 is 9.514930e+00
Standard dev of column 3 is 3.560468e+00
Mean of column 4 is 9.496788e+00
Standard dev of column 4 is 3.532174e+00
```



3)

Code used:

```
covarianceMatrix = cov(x);
fprintf('Covariance matrix is ')
disp(covarianceMatrix);
correlationMatrix = corrcov(covarianceMatrix);
fprintf('Correlation matrix is ')
disp(correlationMatrix);
```

Output:

```
Covariance matrix is
   12.4789   11.3809   11.4285   11.3998
   11.3809   12.3281   11.2993   11.3381
   11.4285   11.2993   12.6769   11.4234
   11.3998   11.3381   11.4234   12.4763
Correlation matrix is
    1.0000    0.9176    0.9086    0.9136
    0.9176    1.0000    0.9039    0.9142
    0.9086    0.9039    1.0000    0.9083
    0.9136    0.9142    0.9083    1.0000
```

4)

The Mean and Standard deviation show that each feature has similar mean values with similar standard distribution, implying that the data has the same relationships. This is shown in the histogram as you can see each feature appears to be approximately normally distributed. The covariance and Correlation matrixes imply that each of the datapoints is strongly correlated to one another within the given column, implying that the data has similar relationships to each other.

K-means Algorithm

a)

Code Used:

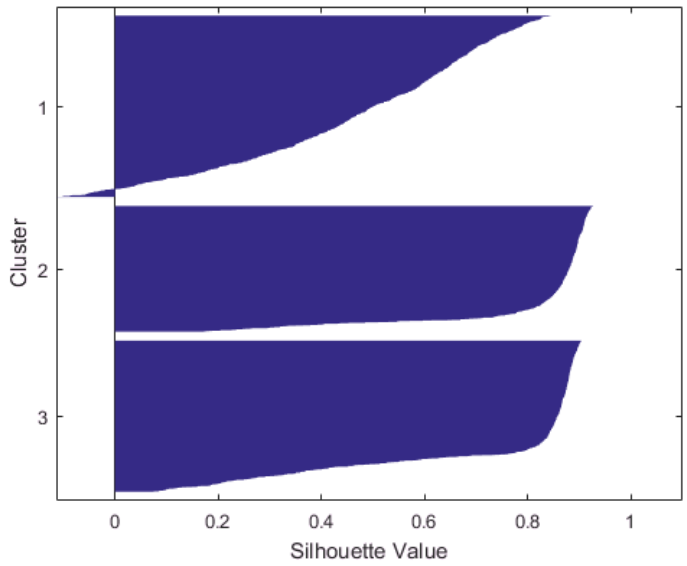
```
bestsil = 0;
 for k = 3:5
     figure();
     [idx,C] = kmeans(x,k);
     fprintf('co-ordinates of centroids for k = %d are\n',k)
     disp(C);
     silhouette(x,idx);
     meansil = mean(silhouette(x,idx));
     if meansil > bestsil
         bestsil = meansil;
         bestk = k;
         bestidx = idx;
         bestC = C;
     end
     fprintf('mean silhouette for k = %d is %d\n',k,meansil);
 end;
 fprintf('best clustering is %d\n', bestk);
```

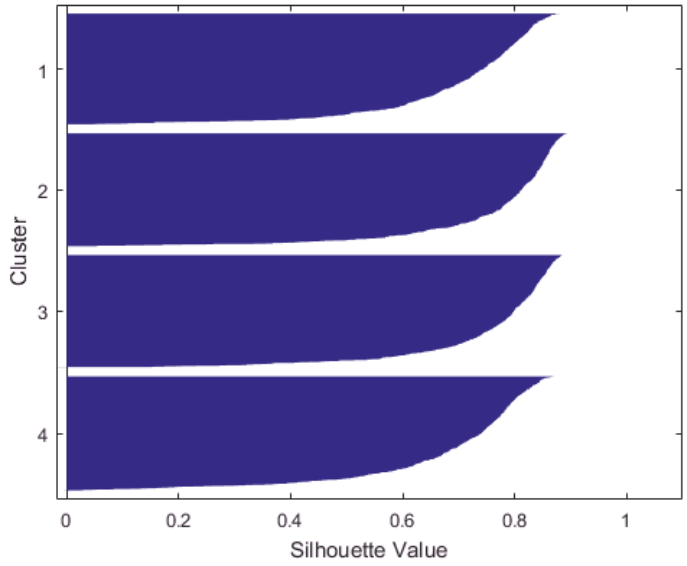
Output:

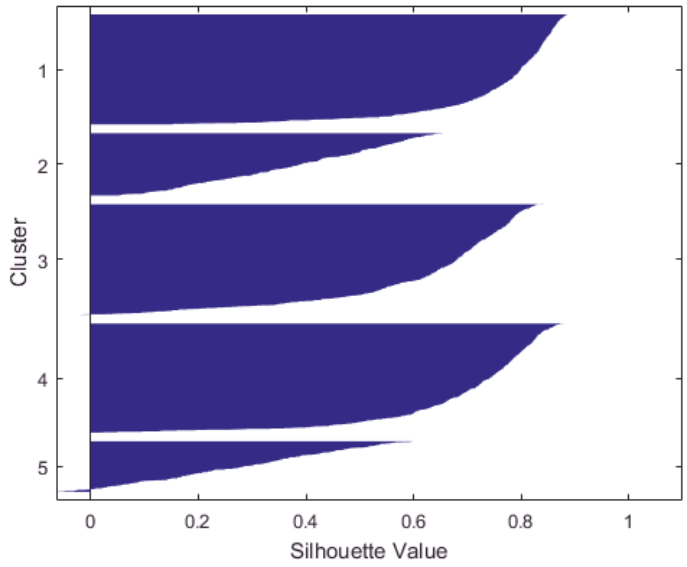
1)

```
 co-ordinates of centroids for k = 3 are
    9.7585    9.8132    9.7884    9.7910
   13.8104   13.7975   13.8639   13.8161
    5.5075    5.5252    5.5723    5.5538

co-ordinates of centroids for k = 4 are
    7.9145    7.9195    8.0114    8.0365
   13.9926   13.9677   14.0073   13.9914
    4.9724    5.0415    4.9748    4.9239
   10.9234   10.9696   11.0012   10.9712

co-ordinates of centroids for k = 5 are
    4.9708    5.0392    4.9718    4.9191
   14.3017   14.4530   14.6446   14.2328
   10.8931   10.9463   10.9851   10.9435
    7.9103    7.9162    8.0083    8.0351
   13.5598   13.3058   13.1431   13.6380
```

2)


Cluster
1
2
3
0
0.2
0.4
0.6
0.8
1
Silhouette Value


Cluster
1
2
3
4
0
0.2
0.4
0.6
0.8
1
Silhouette Value


Cluster
1
2
3
4
5
0
0.2
0.4
0.6
0.8
1
Silhouette Value

3)

```
mean silhouette for k = 3 is 6.559021e-01
mean silhouette for k = 4 is 7.267338e-01
mean silhouette for k = 5 is 6.001589e-01
```

B)

1)

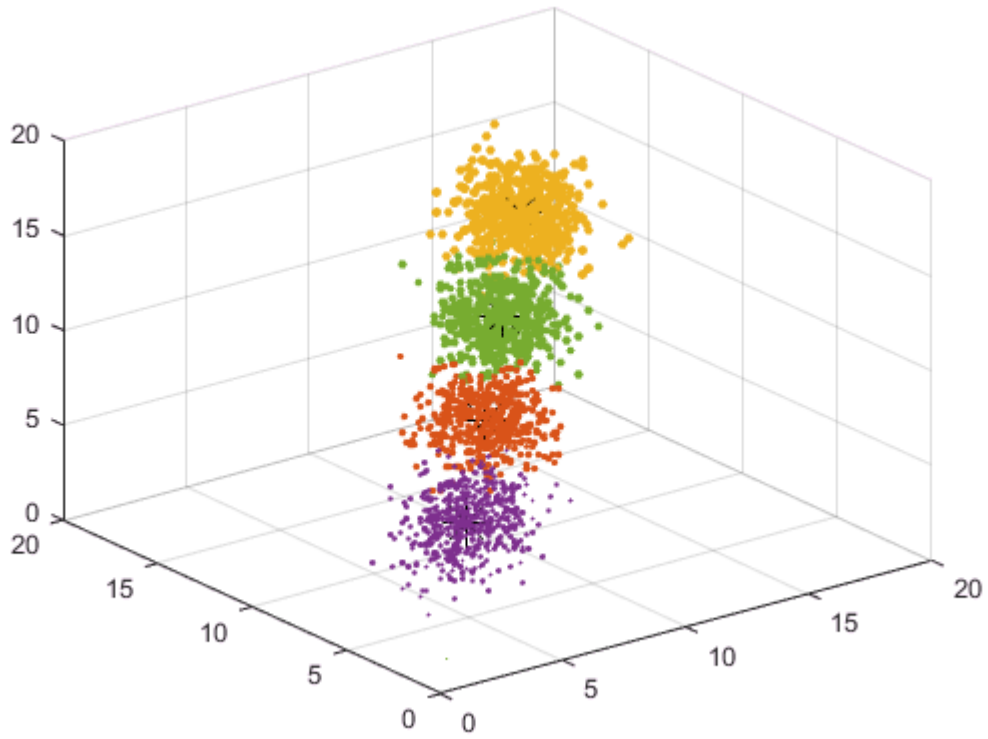
Optimal K is 4 as when k=4 mean silhouette is biggest

2)

Code Used:

```
figure();
 scatter3(bestC(:,1),bestC(:,2),bestC(:,3),200,'black','*');
 hold on;
 for j = 1:bestk;
     tempScat = ones(1,4);
     for i = 1:size(bestidx);
         if bestidx(i) == j
             tempScat = cat(1,tempScat,x(i,:));
         end
     end;
     scatter3(tempScat(:,1),tempScat(:,2),tempScat(:,3),tempScat(:,4),'filled');
 end;
```

Output:



Each colour is a cluster, centroids are Black '*'

Question 2 – K Nearest Neighbour Classifier

1)

Code Used:

```
rows = size(Y,1);

fprintf('rows');

disp(rows);
```

Output:

```
rows

                3042
```

2)

Code Used:

```
columns = size(Y,2);

for c = 1:columns-1

   temp = Y(:,c);

   M = mean(temp);

   fprintf('Mean of column %d is %f\n',c,M);

   sd = std(temp);

   fprintf('standard dev of column %d is %f\n',c,sd);

end
```

Output:

```
Mean of column 1 is 8.912831

standard dev of column 1 is 3.049497

Mean of column 2 is 8.922234

standard dev of column 2 is 3.012221

Mean of column 3 is 8.939463

standard dev of column 3 is 3.064609

Mean of column 4 is 8.865053

standard dev of column 4 is 3.055271

Mean of column 5 is 8.886396

standard dev of column 5 is 3.029849
```

3)

Code Used:

```
covarianceMatrix = cov(Y);
fprintf('Covariance matrix is ')
disp(covarianceMatrix);
correlationMatrix = corrcov(covarianceMatrix);
fprintf('Correlation matrix is ')
disp(correlationMatrix);
```

Output:

```
Covariance matrix is
    9.2994    7.9629    8.4358    7.9619    8.0060    4.7930
    7.9629    9.0735    7.9454    8.0113    7.8904    4.7249
    8.4358    7.9454    9.3918    8.0275    8.0326    4.8046
    7.9619    8.0113    8.0275    9.3347    8.0075    4.7906
    8.0060    7.8904    8.0326    8.0075    9.1800    4.7847
    4.7930    4.7249    4.8046    4.7906    4.7847    2.9176
Correlation matrix is
    1.0000    0.8669    0.9027    0.8545    0.8665    0.9202
    0.8669    1.0000    0.8607    0.8705    0.8646    0.9183
    0.9027    0.8607    1.0000    0.8573    0.8651    0.9178
    0.8545    0.8705    0.8573    1.0000    0.8650    0.9180
    0.8665    0.8646    0.8651    0.8650    1.0000    0.9245
    0.9202    0.9183    0.9178    0.9180    0.9245    1.0000
```

4)

Code Used:

```
NoC = max(Y(:,6));
fprintf('Number of classes is %i\n',NoC);
```

Output:

```
Number of classes is 6
```

5)

The mean and Standard deviation of each feature is very similar implying that the dataset consists of similar features, This is further supported by the covariance and correlation matrixes as these imply that there is a strong correlation between the dataset.

Data Pre-Processing:

Code Used:

```
%get 60% value

ltr = round(rows*0.6);

%get 40% value

lte = rows - ltr;

%make sure there is less training then total

assert(ltr < rows);

%randomly sort array

rRows = randperm(rows);

%preallocate for efficiency

TrainingSet = zeros(ltr,columns);

TestingSet = zeros(lte,columns);

%fill training set with first 60

for i=1:rows

   if i <= ltr

      TrainingSet(i,:) = Y(rRows(i),:);

   else

      TestingSet((i-ltr),:)= Y(rRows(i),:);

   end

end
```

KNN Classifier

A & B)

Code Used

```
k=5;
for a=1:2
   fitknn = fitcknn(TrainingSet(:,1:5),TrainingSet(:,6),'NumNeighbors',k);
   for i=1:size(TestingSet(:,1:5),1)
      Pred_KNN(i) = predict(fitknn,TestingSet(i,1:5));
   end
      %confusion matrix
   for i=1:NoC %max of this is number of classes
      in1=find(TestingSet(:,6)==i);
      nor=length(in1); %number of datas classified as in1
      for j=1:NoC
         Classification=length(find(Pred_KNN(in1)==j));
         Con_Matrix(j,i)=Classification/nor*100;
       end
   end
   %percentage correct
   percentCorrect = length(find((Pred_KNN-TestingSet(:,6)')==0))/length(TestingSet(:,5))*100;
   fprintf('For k = %d \n',k);
   fprintf('Confusion Matrix\n');
   disp(Con_Matrix);
   fprintf('Percentage Correct\n');
   disp(percentCorrect)
   k= k + 2;
End
```

Output

```
For k = 5
Confusion Matrix
   86.1244    8.9552         0         0         0         0
   13.3971   79.6020   10.7623    0.5464         0         0
    0.4785   11.4428   77.5785   10.9290         0         0
         0         0   11.6592   79.2350   11.3861         0
         0         0         0    9.2896   86.6337    3.5176
         0         0         0         0    1.9802   96.4824

Percentage Correct
   84.2235

For k = 7
Confusion Matrix
   88.9952    8.4577         0         0         0         0
   10.5263   81.0945    9.8655    0.5464         0         0
    0.4785   10.4478   78.0269   11.4754         0         0
         0         0   12.1076   81.4208   10.8911         0
         0         0         0    6.5574   87.1287    3.5176
         0         0         0         0    1.9802   96.4824

Percentage Correct
   85.4560
```

C)

When K=5 The KNN classifier is 84.2235% correct, When K=7 the KNN classifier is 85.4560% Correct, giving the conclusion that when k=7 The classifier is correct 1.3% more of the time implying that in this case K=7 is better.

All code for question 1:

```
clear all;
close all;
%data analysis
x = gen_kmeansdata(10554466);
rows = size(x,1);
disp('Rows: ')
disp(rows);


columns = size(x,2);
figure();
for c = 1:columns
    temp = x(:,c);
    M = mean(temp);
    fprintf('Mean of column %d is %d\n',c,M);
    sd = std(temp);
    fprintf('standard dev of column %d is %d\n',c,sd);
    subplot(2,2,c);
    histogram(temp);
    title(c);
end


covarianceMatrix = cov(x);
fprintf('Covariance matrix is ')
disp(covarianceMatrix);
correlationMatrix = corrcov(covarianceMatrix);
fprintf('Correlation matrix is ')
disp(correlationMatrix);


%kmeans stuff
bestsil = 0;
 for k = 3:5
     figure();
     [idx,C] = kmeans(x,k);
     fprintf('co-ordinates of centroids for k = %d are\n',k)
     disp(C);
     silhouette(x,idx);
     meansil = mean(silhouette(x,idx));
     if meansil > bestsil
         bestsil = meansil;
         bestk = k;
         bestidx = idx;
         bestC = C;
     end
     fprintf('mean silhouette for k = %d is %d\n',k,meansil);
 end;
 fprintf('best clustering is %d\n', bestk);
 %scatter best clustering
 figure();
 scatter3(bestC(:,1),bestC(:,2),bestC(:,3),200,'black','*');
 hold on;
 for j = 1:bestk;
```

```
    tempScat = ones(1,4);
    for i = 1:size(bestidx);
        if bestidx(i) == j
            tempScat = cat(1,tempScat,x(i,:));
        end
    end;
    scatter3(tempScat(:,1),tempScat(:,2),tempScat(:,3),tempScat(:,4),'filled');
end;
```

All Code for question 2:

```
clear all;
close all;
rng shuffle;
Y = gen_superdata(10554466);


rows = size(Y,1);
fprintf('rows');
disp(rows);


columns = size(Y,2);


for c = 1:columns-1
    temp = Y(:,c);
    M = mean(temp);
    fprintf('Mean of column %d is %f\n',c,M);
    sd = std(temp);
    fprintf('standard dev of column %d is %f\n',c,sd);
end
NoC = max(Y(:,6));
fprintf('Number of classes is %i\n',NoC);


covarianceMatrix = cov(Y);
fprintf('Covariance matrix is ')
disp(covarianceMatrix);
correlationMatrix = corrcov(covarianceMatrix);
fprintf('Correlation matrix is ')
disp(correlationMatrix);


%get 60% value
ltr = round(rows*0.6);
%get 40% value
lte = rows - ltr;
%make sure there is less training then total
assert(ltr < rows);
%randomly sort array
rRows = randperm(rows);
%preallocate for efficiency
TrainingSet = zeros(ltr,columns);
TestingSet = zeros(lte,columns);
%fill training set with first 60
for i=1:rows
    if i <= ltr
        TrainingSet(i,:) = Y(rRows(i),:);
    else
        TestingSet((i-ltr),:)= Y(rRows(i),:);
    end
```

```
end




k=5;


for a=1:2

    fitknn = fitcknn(TrainingSet(:,1:5),TrainingSet(:,6),'NumNeighbors',k);
    for i=1:size(TestingSet(:,1:5),1)
        Pred_KNN(i) = predict(fitknn,TestingSet(i,1:5));
    end

    %confusion matrix
    for i=1:NoC %max of this is number of classes
        in1=find(TestingSet(:,6)==i);
        nor=length(in1); %number of datas classified as in1
        for j=1:NoC
            Classification=length(find(Pred_KNN(in1)==j));
            Con_Matrix(j,i)=Classification/nor*100;
         end
    end



    %percentage correct
    percentCorrect = length(find((Pred_KNN-
TestingSet(:,6)')==0))/length(TestingSet(:,5))*100;

    fprintf('For k = %d',k);
    fprintf('Confusion Matrix');
    disp(Con_Matrix);
    fprintf('Percentage Correct');
    disp(percentCorrect)


    k= k + 2;



end
```