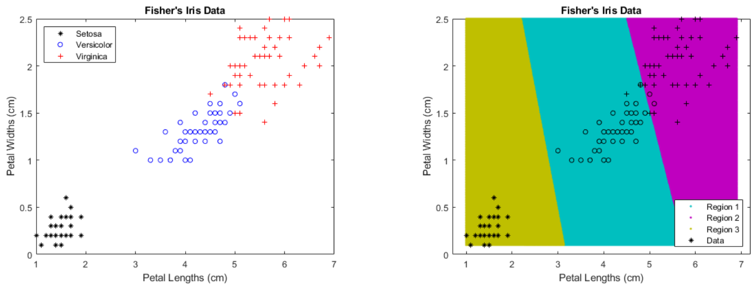
EE583 Pattern Recognition HW4

Kutay Uğurlu 2232841

December 5, 2021

1 Question 1



(a) (b)

Figure 1: The data distribution and partitioning

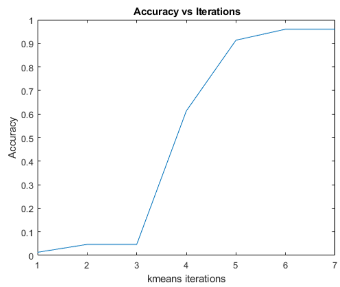
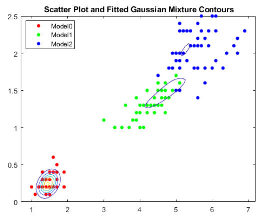


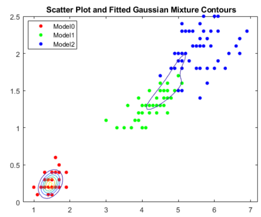
Figure 2: classification success for different centroids

The centroids are manually set according to the approximate mean deduced from the data distribution plot. Initial centroids can be found in 5.1. The accuracy is calculated in two steps. First, the confusion matrix is calculated, then the diagonal entries of it are added to obtain the true positive classifications. Finally, $Accuracy \triangleq \frac{\#TP}{\#Samples}$.
Figure 2 shows the accuracies of the iterations initialized with different centroids. Through careful centroid selection, the accuracy can be increased from 0.1 to 0.9647.

2 Question 2



(a) Case 1



(b) Case 2

Figure 3: Gaussian Fits

I have included 2 different versions of Gaussian fit to the feature vectors, due to the randomness of the algorithm. In Figure 3a, I have observed 3 different Gaussian fits. However, the second trial resulted in algorithm fitting 2 different features to the same Gaussian distribution in Figure 3b.

3 Question 3

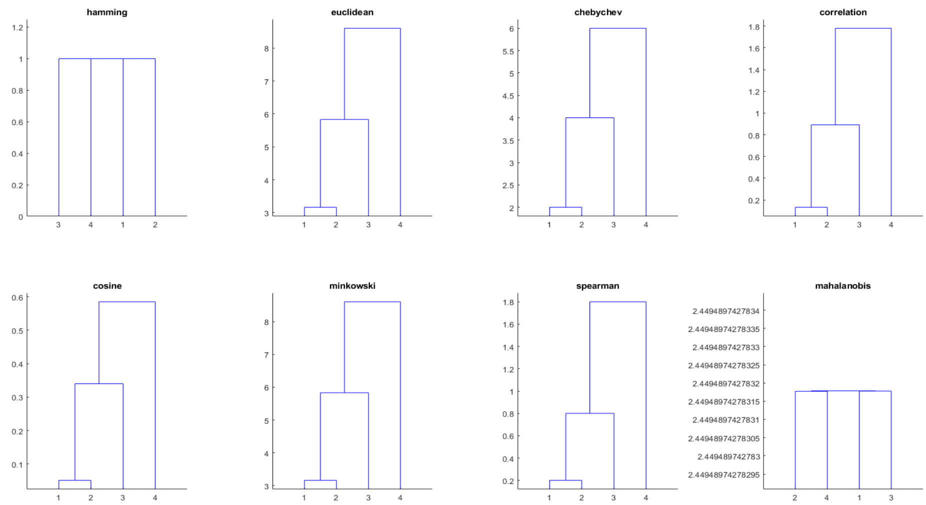
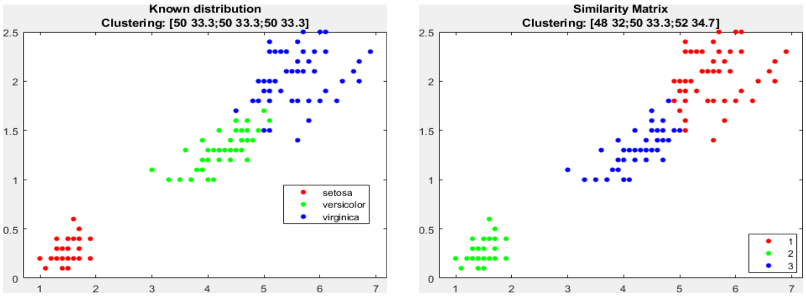


Figure 4: Dendrograms for different metrics

All the metrics beside Hamming and Mahalanobis, resulted in similar dendrograms with easily noticeable difference in the distances in the relevant scales. However, distance calculated using Mahalanobis resulted in close numbers, hence even in order of 10^{-8} the dendrogram presentation is not clear. Hamming distance calculate the percentage of different coordinates in the data matrix X. Since all rows of X carries unique elements, Hamming metric resulted in 100% for all off the feature vectors.

4 Question 4

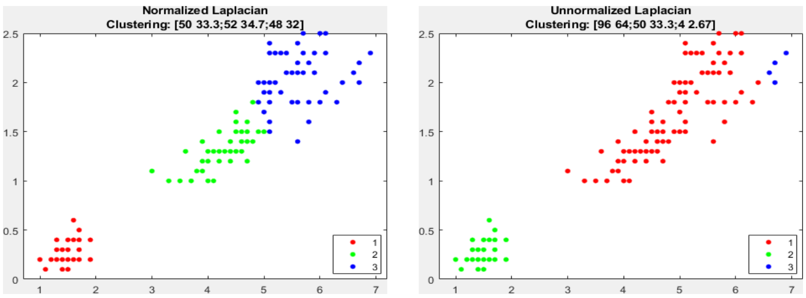
4.1 Precomputed clustering



(a) (b)

Figure 5: Clustering for different cases

4.2 Laplacian Matrix Normalization

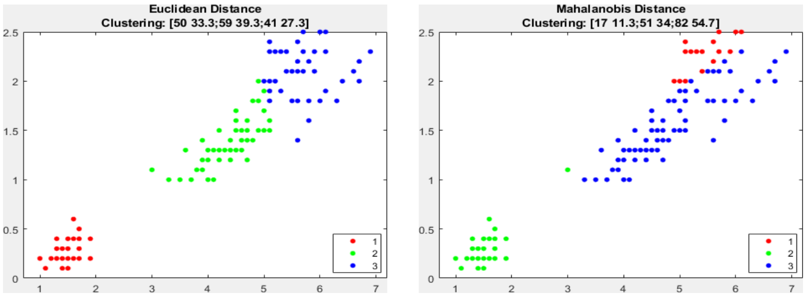


(a) (b)

Figure 6: Clustering for different cases

Using unnormalized Laplacian matrix increased the number of misclassifications.

4.3 Distance Metrics

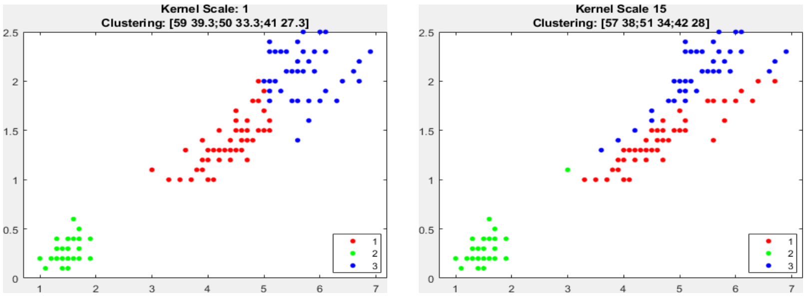


(a) (b)

Figure 7: Clustering for different cases

When the clustering is based on Mahalanobis distance metric, the accuracy gets bad. Since, it takes into distances to the mean of normalized distribution. On the other hand, Euclidean metrics finds the neighborhood relationship simply calculating the distance between data point on 2 dimensional feature space. Hence, the result is closer to the real distribution.

4.4 Kernel Scale



(a) (b)

Figure 8: Clustering for different cases

Kernel scale parameter is used to scale the distance in the pairwise similarity definition:

$$S_{i,j} = e^{-(\frac{D_{i,j}}{\sigma})^2} \tag{1}$$

According to Eqn. 1, the similarity of 2 data points is exponentially proportional to the similarity. Hence, using a larger kernel scale will result in increased similarity in further data points. This is exactly what is observed in Figure 8. Kernel scale of 15 resulted in further points to be classified in the same cluster, hence caused misclassifications.

5 APPENDIX

The code given in this section is shared @.

5.1 Q1

```
load fisheriris
X = meas(:,3:4);
setosa_idx = strcmp(species,"setosa");
versicolor_idx = strcmp(species,"versicolor");
virginica_idx = strcmp(species,"virginica");

SETOSA = X(setosa_idx,:);
VERSICOLOR = X(versicolor_idx,:);
VIRGINICA = X(virginica_idx,:);

figure;
plot(SETOSA(:,1),SETOSA(:,2),'k*','MarkerSize',5);
hold on
plot(VERSICOLOR(:,1),VERSICOLOR(:,2),'bo','MarkerSize',5);
hold on
plot(VIRGINICA(:,1),VIRGINICA(:,2),'r+','MarkerSize',5);
title 'Fisher''s Iris Data';
xlabel 'Petal Lengths (cm)';
ylabel 'Petal Widths (cm)';
legend('Setosa','Versicolor','Virginica','Location','northwest')

%%
% rng(1); % For reproducibility

Cs = {};
Cs{1} = [6 2 ; 1.5 0.3 ; 4 1];
Cs{2} = [5.2 2.5; 4.4 1.25; 3.8 5.5];
Cs{3} = [5 2.4; 4.6 1.45; 3.3 3.5];
Cs{4} = [4.5 2.1; 4.8 1.55; 3.2 3.2];
Cs{5} = [4.2 1.8; 5.2 1.78; 2.5 2.1];
Cs{6} = [4.1 1.4; 5.5 1.89; 1.8 1.05];
Cs{7} = [4 1 ; 6 2 ; 1.5 0.3];
gnd_truth = repelem([3;1;2],50,1);
accs = [];
CONFMATS = {};
for i = 1:7
[idx,C_new] = kmeans(X,3,'MaxIter',1,'Start',Cs{i});
confusion_matrix = confusionmat(gnd_truth,idx);
accuracy = sum(diag(confusion_matrix)) / size(X,1);
C_old = C_new;
accs(end+1) = accuracy;
CONFMATS{end+1} = confusion_matrix;
end
figure
plot(accs)
```

```
title({'\bf Accuracy vs Iterations'})
ylabel('Accuracy')
xlabel('kmeans iterations')
%%
x1 = min(X(:,1)):0.01:max(X(:,1));
x2 = min(X(:,2)):0.01:max(X(:,2));
[x1G,x2G] = meshgrid(x1,x2);
XGrid = [x1G(:),x2G(:)]; % Defines a fine grid on the plot

idx2Region = kmeans(XGrid,3,'MaxIter',1,'Start',C_new);
    % Assigns each node in the grid to the closest centroid

figure;
gscatter(XGrid(:,1),XGrid(:,2),idx2Region,...
    [0,0.75,0.75;0.75,0,0.75;0.75,0.75,0],'..');
hold on;
plot(X(setosa_idx,1),X(setosa_idx,2),'k*','MarkerSize',5);
hold on
plot(X(versicolor_idx,1),X(versicolor_idx,2),'ko','MarkerSize',5);
hold on
plot(X(virginica_idx,1),X(virginica_idx,2),'k+','MarkerSize',5); title 'Fisher''s Iris Data';
xlabel 'Petal Lengths (cm)';
ylabel 'Petal Widths (cm)';
legend('Region 1','Region 2','Region 3','Data','Location','SouthEast');
hold off;
```

5.2 Q2

```
load fisheriris.mat
X = meas(:,3:4);
GMModel = fitgmdist(X,3);
figure
y = [zeros(50,1);ones(50,1);2*ones(50,1)];
h = gscatter(X(:,1),X(:,2),y);
hold on
gmPDF = @(x,y) arrayfun(@(x0,y0) pdf(GMModel,[x0 y0]),x,y);
g = gca;
fcontour(gmPDF,[g.XLim g.YLim])
title('{\bf Scatter Plot and Fitted Gaussian Mixture Contours}')
legend(h,'Model0','Model1','Model2')
hold off
```

5.3 Q3

```
X = [0 1 2 3; 1 0 4 5; 2 4 0 6; 3 5 6 0];
y = squareform(X);

distances = ["hamming","euclidean","chebychev","correlation",...
    "cosine","minkowski","spearman","mahalanobis"];
figure('units','normalized','outerposition',[0 0 1 1])
for distance = distances
    idx = find(distances == distance);
    subplot(2,4,idx)
    Z = linkage(X,'complete',distance);
    dendrogram(Z)
    title(distance)
end
```

5.4 Q4

```
%%
chdir('..')
addpath('export_fig')
chdir('HW4')
%%
load fisheriris
X = meas(:,3:4);
gscatter(X(:,1),X(:,2),species);
dist_temp = pdist(X);
dist = squareform(dist_temp);
S = exp(-dist.^2);
k = 3; % Number of clusters
rng('default') % For reproducibility
mat = tabulate(species);
title({'Known distribution',['Clustering: ',mat2str(repmat([50,33.3],3,1),3)]})
%% Use similarity mat(:,2:end)rix
figure
idx = spectralcluster(S,k,'Distance','precomputed','LaplacianNormalization','symmetric');
gscatter(X(:,1),X(:,2),idx);
mat = tabulate(idx);
title({'Similarity Matrix',['Clustering: ',mat2str(mat(:,2:end),3)]})
%% Laplacian normalized
figure
idx2 = spectralcluster(X,k,'NumNeighbors',size(X,1),'LaplacianNormalization','symmetric');
gscatter(X(:,1),X(:,2),idx2);
mat = tabulate(idx2);
title({'Normalized Laplacian',['Clustering: ',mat2str(mat(:,2:end),3)]})
%% Laplacian unnormalized
figure
idx2 = spectralcluster(X,k,'NumNeighbors',size(X,1),'LaplacianNormalization','none');
gscatter(X(:,1),X(:,2),idx2);
mat = tabulate(idx2);
title({'Unnormalized Laplacian',['Clustering: ',mat2str(mat(:,2:end),3)]})

%% Distance Euclidean
figure
idx2 = spectralcluster(X,k,'NumNeighbors',size(X,1),'Distance','euclidean');
gscatter(X(:,1),X(:,2),idx2);
mat = tabulate(idx2);
title({'Euclidean Distance',['Clustering: ',mat2str(mat(:,2:end),3)]})
```

```
%% Distance mahalanobis
figure
idx2 = spectralcluster(X,k,'NumNeighbors',size(X,1),'Distance','mahalanobis');
gscatter(X(:,1),X(:,2),idx2);
mat = tabulate(idx2);
title({'Mahalanobis Distance',['Clustering: ',mat2str(mat(:,2:end),3)]})

%% Kernel Scale 1
figure
idx2 = spectralcluster(X,k,'NumNeighbors',size(X,1),'KernelScale',1);
gscatter(X(:,1),X(:,2),idx2);
mat = tabulate(idx2);
title({'Kernel Scale: 1',['Clustering: ',mat2str(mat(:,2:end),3)]})

%% Kernel Scale 15
figure
idx2 = spectralcluster(X,k,'NumNeighbors',size(X,1),'KernelScale',15);
gscatter(X(:,1),X(:,2),idx2);
mat = tabulate(idx2);
title({'Kernel Scale 15',['Clustering: ',mat2str(mat(:,2:end),3)]})


%%
figHandles = findall(0,'Type','figure');

for i = 1:numel(figHandles)
    export_fig(['Q4_',num2str(i)], '-png', figHandles(i), '-append')
end
```