EE583 Pattern Recognition HW2

Kutay Uğurlu 2232841

November 17, 2021

1 Question 1

Maximum Likelihood Estimate of Multivariate Gaussian random variable are calculated as:

- $\hat{\mu} = \frac{1}{m}\sum_{i=1}^{m} X^{(i)}$
- $\hat{\Sigma} = \frac{1}{m}\sum_{i=1}^{m}(X^{(i)} - \hat{\mu})(X^{(i)} - \hat{\mu})^T$

where $X^{(i)}$ is the i^{th} observation.

The given MATLAB code in Section 6.1 produces the following results:

$m = 10$ $\quad \hat{\mu} = \begin{bmatrix} -0.4519 & 1.0984 \end{bmatrix}$ $\quad \hat{\Sigma} = \begin{bmatrix} 0.3191 & 0.1799 \\ 0.1799 & 0.6506 \end{bmatrix}$

$m = 1000$ $\quad \hat{\mu} = \begin{bmatrix} -0.7367 & 0.5176 \end{bmatrix}$ $\quad \hat{\Sigma} = \begin{bmatrix} 0.4730 & 0.2925 \\ 0.2925 & 0.8079 \end{bmatrix}$

It is observed that the estimations get more accurate, *i.e.* the calculated metrics get closer to the original ones, with increasing number of samples.

2 Question 2

The ML estimate is calculated using mean of the generated data. On the other hand, to calculate the MAP estimate of μ, following formula is utilized:

$$\hat{\mu}_n = \frac{n{\sigma_0}^2}{n{\sigma_0}^2 + \sigma^2}\frac{1}{n}\sum_{i=1}^{n} x_i + \frac{\sigma^2\mu_0}{n{\sigma_0}^2 + \sigma^2} \tag{1}$$

The formula given is a normalized weighted average of mean of the training data and mean assumed previously. The mean estimate in Eq. 1 converges to sample mean when $n \to \infty$. For $n = 0$, the estimate is just the prior mean, and it gets closer to the sample mean when n increases. Therefore, one would expect to see an estimate around μ_0 for "small" n. However, even for small values of n, such as 25, the weight of the sample mean approaches to 0.95. Still, for small n, the sample mean does not converge to the actual sample mean, and in one experiment, I have made the following observations.

2.1 25 Samples

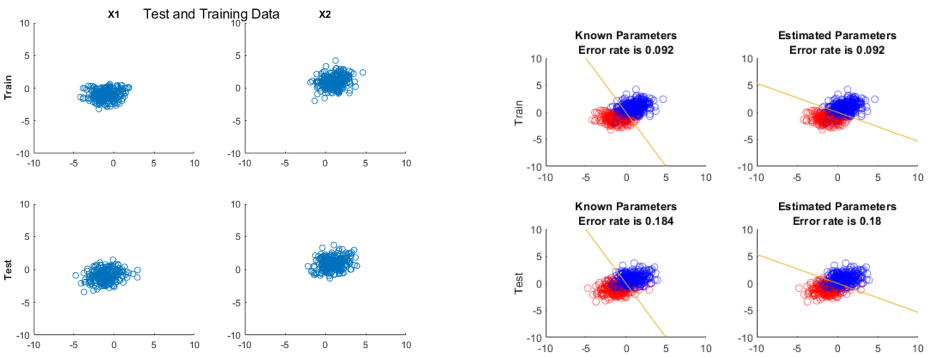
$\hat{x}_{ML} = 2.8995$ $\quad \hat{x}_{MAP} = 2.9019$

2.2 1000 samples

$\hat{x}_{ML} = 2.9983$ $\quad \hat{x}_{MAP} = 2.9984$

With increasing n, the sample mean dominates the estimate for sure. But for small n around 20, the estimate is highly dependent on the sample mean. It is also observed that, the estimates gets closer with increasing number of training samples.

3 Question 3



(a) Training and Test data (b) Decision Boundaries and Error Rates

Figure 1: Data distributions and Decision Boundaries

As expected, training accuracy is higher than the test accuracy. Furthermore, decision boundaries obtained with estimated parameters resulted in lower accuracy, since the minimum error rate classifier boundaries are calculated analytically with already known distribution parameters.

4 Question 4

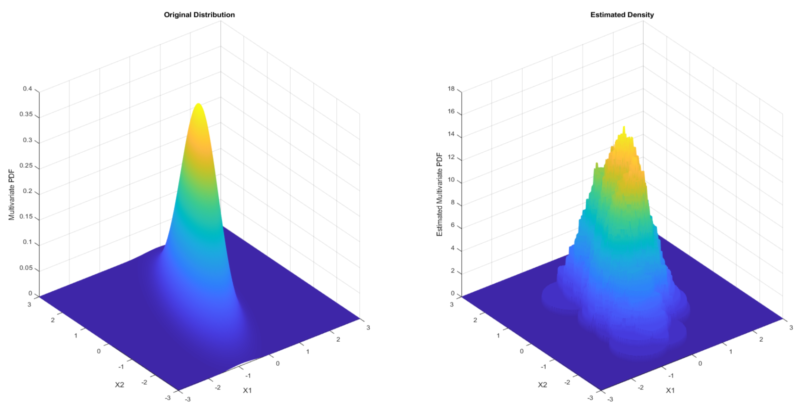
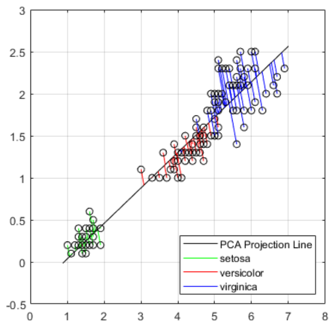


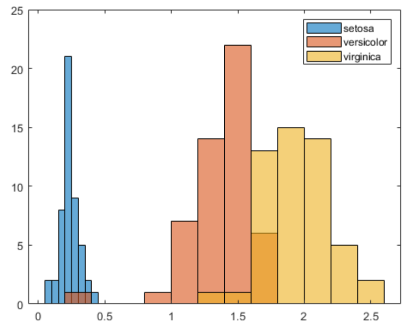
Figure 2: Parzen window estimation results

The selection of initial volume and volume shrinking formula are presented in subsection 6.4.

5 Question 5



(a) PCA Projection onto 1D



(b) Histograms

Figure 3: PCA Projection and histogram

6 APPENDIX

The code given in this section is shared @.

6.1 Q1

```
% Q1
m = 1000;
sigma_1 = [0.5  0.3;0.3  0.8];
mu_1 = [-0.75;0.5];
X = mvnrnd(mu_1,sigma_1,m);
mu_1_hat = mean(X);

sum = zeros(size(sigma_1));
for i = 1:m
    sum = sum + (X(i,:)-mu_1_hat)'*(X(i,:)-mu_1_hat);
end
sigma_1_hat = sum/m;
mu_1_hat
sigma_1_hat
```

6.2 Q2

```
% Part i - ML Estimate of mean
n = 25;
sigma_sq = 0.49;
X = sqrt(sigma_sq) * randn(n,1) + 3;
mu_x_ML = mean(X);
% Part ii - MAP Estimate of mean
mu_0 = 2.8;
sigma_0_sq = 0.8;
mu_n = n*sigma_0_sq/(n*sigma_0_sq+sigma_sq) * mean(X) + sigma_sq
    /(n*sigma_0_sq+sigma_sq) * mu_0;
```

6.3 Q3

```
m = 250;
sigma = [1.4 0.2;0.2 0.8];
mu_1 = [-1;-1];
X1 = mvnrnd(mu_1,sigma,m);
mu_2 = [1;1];
X2 = mvnrnd(mu_2,sigma,m);
edgealpha = 0.5;
%% Decision boundary
% Since class priors are equal
figure
subplot(2,2,1)
x0 = 0.5 * (mu_1+mu_2); % = 0
A = (mu_1-mu_2)'*(inv(sigma))'; % =   -1.1111    -2.2222
% Then decision boundary -> -x1-2x2 = 0 -> x1 = -2x2
scatter(X1(:,1),X1(:,2),'red','MarkerEdgeAlpha',edgealpha)
hold on
scatter(X2(:,1),X2(:,2),'blue','MarkerEdgeAlpha',edgealpha)
hold on
x = linspace(-5,5,100);
plot(x,-2*x)
ylabel('Train')
xlim([-10,10])
ylim([-10,10])
conf_mat_1 = zeros(2,2);

for i = 1:m
    x = X1(i,:);
    eval_at_line = A * transpose(x);
    if eval_at_line > 0
        conf_mat_1(1,1) = conf_mat_1(1,1) + 1;
    else
        conf_mat_1(2,1) = conf_mat_1(2,1) + 1;
    end

    x = X2(i,:);
    eval_at_line = A * transpose(x);
    if eval_at_line > 0
        conf_mat_1(2,2) = conf_mat_1(1,2) + 1;
    else
        conf_mat_1(1,2) = conf_mat_1(2,2) + 1;
    end
end
error_rate_1 = (conf_mat_1(1,2) + conf_mat_1(2,1)) / (2*m);
```

```
title({'Known Parameters',['Error rate is ',num2str(error_rate_1)
    ]})

%% ML Estimation
subplot(2,2,2)
[mu_1_hat,Sigma_1_hat] = estimate_ML(X1);
[mu_2_hat,Sigma_2_hat] = estimate_ML(X2);
Sigma = 0.5 * (Sigma_1_hat+Sigma_2_hat);
x0 = 0.5 * (mu_1_hat+mu_2_hat); % = 0
A_ML = (mu_1_hat-mu_2_hat)'*(inv(Sigma))'; %
syms f(x);
scatter(X1(:,1),X1(:,2),'red','MarkerEdgeAlpha',edgealpha)
hold on
scatter(X2(:,1),X2(:,2),'blue','MarkerEdgeAlpha',edgealpha)
hold on
f(x) = x0(2) - A_ML(1)/A_ML(2) * (x-x0(1));
fplot(f(x))
xlim([-10,10])
ylim([-10,10])

conf_mat_2 = zeros(2,2);

for i = 1:m
    x = X1(i,:);
    eval_at_line = A * transpose(x);
    if eval_at_line > 0
        conf_mat_2(1,1) = conf_mat_2(1,1) + 1;
    else
        conf_mat_2(2,1) = conf_mat_2(2,1) + 1;
    end

    x = X2(i,:);
    eval_at_line = A * transpose(x);
    if eval_at_line > 0
        conf_mat_2(2,2) = conf_mat_2(1,2) + 1;
    else
        conf_mat_2(1,2) = conf_mat_2(2,2) + 1;
    end
end
error_rate_2 = (conf_mat_2(1,2) + conf_mat_2(2,1)) / (2*m);
title({'Estimated Parameters',['Error rate is ',num2str(
    error_rate_2)]})
```

```
%% Test Data

X1_test = mvnrnd(mu_1,sigma,m);
X2_test = mvnrnd(mu_2,sigma,m);

%% Test Data Known
subplot(2,2,3)

x0 = 0.5 * (mu_1+mu_2); % = 0
A = (mu_1-mu_2)'*(inv(sigma))'; % = -1.1111 -2.2222
% Then decision boundary -> -x1-2x2 = 0 -> x1 = -2x2
scatter(X1_test(:,1),X1_test(:,2),'red','MarkerEdgeAlpha',
    edgealpha)
hold on
scatter(X2_test(:,1),X2_test(:,2),'blue','MarkerEdgeAlpha',
    edgealpha)
hold on
x = linspace(-5,5,100);
plot(x,-2*x)
ylabel('Test')
xlim([-10,10])
ylim([-10,10])
conf_mat_1 = zeros(2,2);

for i = 1:m
    x = X1_test(i,:);
    eval_at_line = A * transpose(x);
    if eval_at_line > 0
        conf_mat_1(1,1) = conf_mat_1(1,1) + 1;
    else
        conf_mat_1(2,1) = conf_mat_1(2,1) + 1;
    end

    x = X2_test(i,:);
    eval_at_line = A * transpose(x);
    if eval_at_line > 0
        conf_mat_1(2,2) = conf_mat_1(1,2) + 1;
    else
        conf_mat_1(1,2) = conf_mat_1(2,2) + 1;
    end
end
error_rate_1 = (conf_mat_1(1,2) + conf_mat_1(2,1)) / (2*m);

title({'Known Parameters',['Error rate is ',num2str(error_rate_1)
    ]})
```

```
%% ML Estimation Test data
subplot(2,2,4)
scatter(X1_test(:,1),X1_test(:,2),'red','MarkerEdgeAlpha',
    edgealpha)
hold on
scatter(X2_test(:,1),X2_test(:,2),'blue','MarkerEdgeAlpha',
    edgealpha)
hold on
syms f(x)
f(x) = x0(2) - A_ML(1)/A_ML(2) * (x-x0(1));
fplot(f(x))
xlim([-10,10])
ylim([-10,10])

conf_mat_2 = zeros(2,2);

for i = 1:m
    x = X1_test(i,:);
    eval_at_line = A_ML * transpose(x);
    if eval_at_line > 0
        conf_mat_2(1,1) = conf_mat_2(1,1) + 1;
    else
        conf_mat_2(2,1) = conf_mat_2(2,1) + 1;
    end

    x = X2_test(i,:);
    eval_at_line = A_ML * transpose(x);
    if eval_at_line > 0
        conf_mat_2(2,2) = conf_mat_2(1,2) + 1;
    else
        conf_mat_2(1,2) = conf_mat_2(2,2) + 1;
    end
end
error_rate_2 = (conf_mat_2(1,2) + conf_mat_2(2,1)) / (2*m);
title({'Estimated Parameters',['Error rate is ',num2str(
    error_rate_2)]})

figure
suptitle('Test and Training Data')
subplot(2,2,1)
scatter(X1(:,1),X1(:,2))
title('X1')
xlim([-10,10])
ylim([-10,10])
```

```
ylabel('Train','FontWeight','bold')
subplot(2,2,2)
scatter(X2(:,1),X2(:,2))
title('X2')
xlim([-10,10])
ylim([-10,10])
subplot(2,2,3)
scatter(X1_test(:,1),X1_test(:,2))
xlim([-10,10])
ylim([-10,10])
ylabel('Test','FontWeight','bold')
subplot(2,2,4)
scatter(X2_test(:,1),X2_test(:,2))
xlim([-10,10])
ylim([-10,10])
```

6.4 Q4

```
m = 50;
mu = [0 0];
Sigma = [0.25 0.3; 0.3 1];
x1 = -3:0.02:3;
x2 = -3:0.02:3;
[X1,X2] = meshgrid(x1,x2);
X = [X1(:) X2(:)];
y = mvnpdf(X,mu,Sigma);
y = reshape(y,length(x2),length(x1));
subplot(1,2,1)
surf(x1,x2,y,'EdgeColor','interp','EdgeAlpha',1)
title('Original Distribution')
xlabel('X1')
ylabel('X2')
zlabel('Multivariate PDF')

%% Parzen
mu = [0 0];
Sigma = [0.25 0.3; 0.3 1];
n = 50;
X_sampled = mvnrnd(mu, Sigma, n);
h = 1.55;
V = h/n;
pn = zeros(size(X,1),1);
Estimated_density = zeros(size(X,1),1);
for i = 1:size(X,1)
    x = X(i,:);
    sum = 0;
    for j = 1:n
        sum = sum + 1/V * parzen_win((x-X_sampled(j,:))/h);
    end
    pn = 1/n * sum;
    Estimated_density(i) = pn;
end
subplot(1,2,2)
Estimated_density = reshape(Estimated_density,length(x2),length(x1));
surf(x1,x2,Estimated_density,'EdgeColor','interp','EdgeAlpha',1)
title('Estimated Density')
xlabel('X1')
ylabel('X2')
zlabel('Estimated Multivariate PDF')
```

6.5 Q5

```
X = load('fisheriris');
features = X.meas;
classes = X.species;
X = features(:,[3 4]);
figure
scatter(X(:,1),X(:,2),'bo');
grid on;
maxlim = max(abs(X(:)))*1.1;
axis([-maxlim maxlim -maxlim maxlim]);

%% PCA
[coeff,score,roots] = pca(X);
basis = coeff(:,1);
normal = coeff(:,2);
[n,p] = size(X);
meanX = mean(X,1);
Xfit = repmat(meanX,n,1) + score(:,1)*coeff(:,1)';
residuals = X - Xfit;

%% Visualize Line Fit
dirVect = coeff(:,1);
Xfit1 = repmat(meanX,n,1) + score(:,1)*coeff(:,1)';
t = [min(score(:,1))-.2, max(score(:,1))+.2];
endpts = [meanX + t(1)*dirVect'; meanX + t(2)*dirVect'];
plot(endpts(:,1),endpts(:,2),'k-');
X1 = [X(:,1) Xfit1(:,1)];
X2 = [X(:,2) Xfit1(:,2)];
hold on
plot(X1(1:50,:)',X2(1:50,:)','g-', X(1:50,1),X(1:50,2),'ko');
hold on
plot(X1(50:100,:)',X2(50:100,:)','r-', X(50:100,1),X(50:100,2),'
    ko');
hold on
plot(X1(100:150,:)',X2(100:150,:)','b-', X(100:150,1),X
    (100:150,2),'ko');

L(1) = plot(nan, nan, 'k-');
L(2) = plot(nan, nan, 'g-');
L(3) = plot(nan, nan, 'r-');
L(4) = plot(nan, nan, 'b-');

legend_cell = cell(4,1);
legend_cell{1} = 'PCA Projection Line';
```

```
legend_cell{2} = 'setosa';
legend_cell{3} = 'versicolor';
legend_cell{4} = 'virginica';

legend(L,legend_cell,'Location','southeast');

hold off
maxlim = max(abs(X(:)))*1.1;
axis([0 8 -0.5 3]);
axis square
grid on
setosa = Xfit1(1:50,2);
versicolor = Xfit1(50:100,2);
virginica = Xfit1(100:150,2);
figure
histogram(setosa)
hold on
histogram(versicolor)
hold on
histogram(virginica)
legend('setosa','versicolor','virginica')
```