به نام خدا

دانشگاه شیراز

دانشکده برق و کامپیوتر

شناسایی الگو

تمرین شماره ۲

سارا سیامک

۹۹۳۰۲۶۴

آذر ماه ۱۳۹۹

Contents

Abstract

In this exercise, 2 problems are investigated with the parametric classifications Maximum Likelihood (ML) estimation and Maximum a Posteriori (MAP) estimation.

In problem 1, the TinyMNIST dataset consists of a set of extracted handwriting features are available to us. There are 10 classes from 0 to 9.

At first, the parameters of Gaussian distributions for 10 classes are estimated using Maximum Likelihood Estimation. A mean vector and a covariance matrix are estimated for each class. But one of the covariance matrices is singular and cannot be used in the related distribution function.

To overcome this problem, the LDA method is used to obtain a full-rank covariance matrix and use it instead of the singular matrix.

Then, by using the known parameters of distributions for all classes, the test data are classified with Bayes optimal classifier.

NOTE: At the end of problem 1, MATLAB code is located.

In problem 2, the relation of the best estimation to obtain the one-dimensional mean in the MAP method is proved.

Problem #1

1.1. Design and implement a Bayes optimal classifier with a Gaussian parametric estimate of pdfs to minimize the probability of classification error using a reduced TinyMNIST dataset. You must state the equations which are used for the parameter estimation, and also explain how you choose the prior probabilities of the classes.

In this problem, the training dataset have 5000 data sample which have 62 features. According to the label of training data, the samples are placed in their classes. Considering the number of data in each class, the prior probability of each class can be obtained by the following:

$$P(\omega_i) = \frac{N_i}{N}$$

Where $N = 5000$ is the total number of dataset and N_i is the number of patterns per class. And $\sum_{i=1}^{10} P(\omega_i) = 1$.

TABLE 1. Prior probability of training data

Class Number	1	2	3	4	5	6	7	8	9	10
Abundance (N_i)	522	564	503	539	445	429	513	520	482	483
Prior Probability ($P(\omega_i)$)	0.1044	0.1128	0.1006	0.1078	0.0890	0.0858	0.1026	0.1040	0.0964	0.0966

For each class, the mean of the data should be obtained. For this problem, the Maximum Likelihood (ML) estimation is used to estimate the means. The means are 10 vectors which sizes are 62*1. With the following equation mean vectors are obtained:

$$\mu_i = \frac{1}{N_i} \sum_{k=1}^{N_i} x_k$$

Where μ_i is the mean vector for ith class. The training pattern x_k is a 62*1 vector. k varies from 1 to N_i and N_i is the number of data per each class.

According to the ML estimation, the covariance matrix is obtained by the following equation:

$$\Sigma_i = \frac{1}{N_i}\sum_{k=1}^{N_i}(x_k - \mu_i)(x_k - \mu_i)^T$$

Where Σ_i is the covariance matrix of ith class.

After computing the mean and covariance for each class, the Gaussian distribution functions can be obtained as follows:

$$P(x_k|\omega_i) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_i|^{\frac{1}{2}}} e^{\frac{-1}{2}(x_k-\mu_i)^T\Sigma_i^{-1}(x_k-\mu_i)}$$

In this case, d=62 is the dimensional of the features. The above equation is used for obtaining $P(x_k|\omega_i)$ but some problem has occurred in the one of covariance matrix that be investigated in section 1.2 and the problem is solved.

Now, we can go to the test dataset for the classification of the new data by Bayes classification method. The prior probabilities and the Gaussian distribution functions are prepared and the goal is to minimization of the error probability or maximization of the posterior probability of each sample data. According to the Bayes decision:

$$P(\omega_i|x_k) = \frac{P(x_k|\omega_i)P(\omega_i)}{P(x_k)}$$

$$\omega_{max} = arg\max_{\omega_i} P(\omega_i|x_k)$$

x_k is the kth sample. ω_{max} is the class of x_k. $P(x_k)$ is equal for all data so it can be ignored in the maximization. To design a Bayes optimal classifier, the above relationships are implemented. In the next section, the problems that arise, the solution and the final classification answer are given.

1.2. When estimating Gaussian distribution parameters, sometimes a singular matrix is obtained as the covariance of the data.

a) Why this situation is problematic?

In many pattern recognition, it happens that the number of samples in each class is less than the dimensionality of the features. For example, in the image data like face recognition application which the number of the pixels of the image is very high, the face images that are used for each class are so lower than the number of pixels. This means that the covariance matrix of such cases is singular, and there is no inversion for them.

In this problem, the images of the handwritten have 28*28 pixels initially, which means that the number of features is 784. But the number of data in each class is between 429 to 564. So the number of training patterns in each class is less than the dimension of the feature space.

In this problem, the covariance matrix determinant of class ω_2 is zero. The determinant of the covariance matrix is placed in the denominator of the distribution function, and there is a problem with the determinant of class ω_2. Also, in the exponential function of the Gaussian distribution function, the inverse of Σ_2 cannot be calculated. So we must use the methods which can help us to calculate a covariance matrix for class ω_2 which is reversible, therefore the use of a method that estimates a full-rank covariance matrix is useful.

b) This difficulty arises for the given dataset. Using the following hint, study the proposed methods, and apply one of them to your classifier. Evaluate your classifier.

According to the problem's given hint, the LDA method is used to improve the current situation concerning the class whose covariance matrix is in ill condition. In this method, all data of entire classes are pooled, and instead of the ill condition covariance matrix, an estimation of the covariance matrix is used. This estimation involves a mean of the weighted covariance matrix of all classes. The weight of each covariance matrix is a function of the number of data in each class. In the averaging operation, the number of classes is deduced from the considered total number of data because, in this method, each class's data number is deduced by one.

The pooled estimation of the covariance matrix for $g = 10$ classes and total data $N = 5000$ is as follows:

$$\Sigma_p = \frac{1}{N-g}\sum_{i=1}^{g}(N_i - 1)\Sigma_i = \frac{(N_1 - 1)\Sigma_1 + (N_2 - 1)\Sigma_2 + \cdots + (N_g - 1)\Sigma_g}{N-g}$$

In this problem, number of total train data is 5000 and the number of classes is 10. The number of data in each class is shown in the following table.

TABLE 2. The number of each class data

Class Number	1	2	3	4	5	6	7	8	9	10
Number of Class Data	522	564	503	539	445	429	513	520	482	483

After calculating of Σ_p, the covariance matrix of class 2, i.e. Σ_2 is replaced by Σ_p.

TABLE 3. The classifier accuracy for each class

Class Number	1	2	3	4	5	6	7	8	9	10
Recognition Percentage	96.804	94.425	89.492	87.795	89.091	85.520	86.222	79.767	85.124	83.197
Error Percentage	3.196	5.575	10.507	12.205	10.909	14.480	13.778	20.233	14.876	16.803

The total error percentage is 12.256%

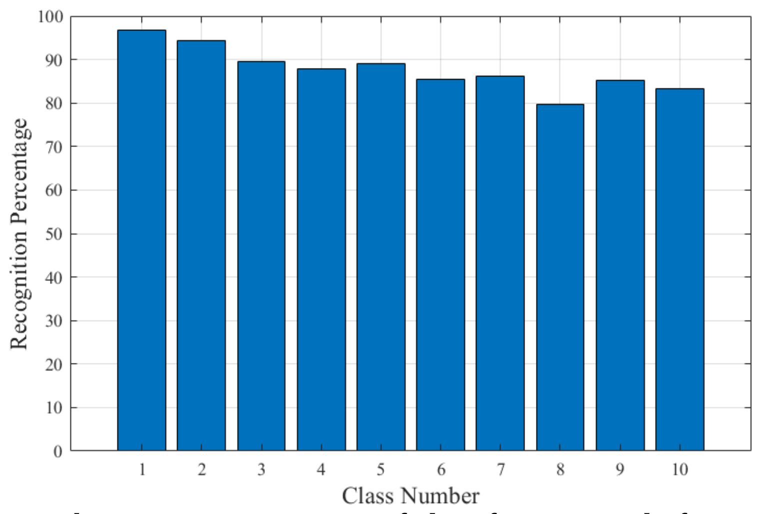

Figure 1. The accuracy percentage of classifier separately for each class

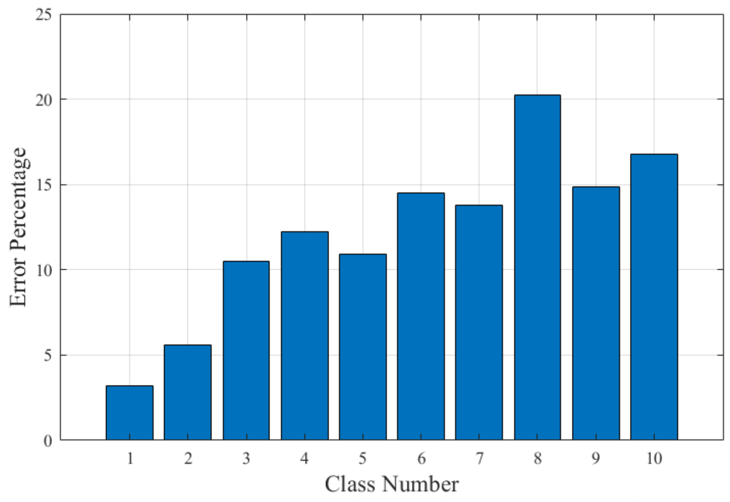

Figure 2. The error percentage of classifier separately for each class

TABLE 4. Confusion Matrix

		Actual Class										
		1	2	3	4	5	6	7	8	9	10	Total
Recognized Class	1	212	0	2	1	0	3	5	0	2	2	227
	2	6	271	14	12	9	10	18	29	30	19	418
	3	0	1	247	6	3	0	0	13	1	2	273
	4	0	0	2	223	2	11	0	2	3	2	245
	5	0	5	1	0	245	0	1	1	0	8	261
	6	0	0	0	2	0	189	4	1	0	0	196
	7	0	1	1	0	3	2	194	0	0	0	201
	8	0	0	1	1	2	0	0	205	0	2	211
	9	1	9	8	7	4	6	3	0	206	6	250
	10	0	0	0	2	7	0	0	6	0	203	218
	Total	219	287	276	254	275	221	225	257	242	244	2500

The last row is shown the abundance of the data per each class from the label data. In the bellow table, it is shown.

TABLE 1. Actual abundance of test data per each class from the label of data

Class Number	1	2	3	4	5	6	7	8	9	10
Abundance per each class	219	287	276	254	275	221	225	257	242	244

MATLAB Codes

In the following section the data is loaded:

```
addpath('Tiny_MNIST_Dataset')
load('MNIST_data')

Train_data=TrainData;
Train_label=Trainlabels;
Test_data=TestData;
Test_label=Testlabels;
```

Then, the class of training data is read from their labels and $P(\omega_i)$ are calculated by the following:

```
%% Class
% P(wj|D)
omega=zeros(10,1);
for i=1:size(Train_label,1)
    for j=1:10
       if Train_label(i)==j-1
           omega(j)=omega(j)+1;
       end
    end
end

P_w=omega/size(Train_label,1);
sum_P_w=sum(P_w);
```

The next work is to separating the data to each class:

```
% Separating the data
Di=zeros(10,size(Train_label,1));
for i=1:size(Train_label,1)
    for j=1:10
       if Train_label(i)==j-1
        Di(j,i)=i;
       end
    end
end
D0=[];D1=[];D2=[];D3=[];D4=[];
D5=[];D6=[];D7=[];D8=[];D9=[];
```

```
for i=1:size(Train_label,1)
   for j=1:10
    if Di(j,i)~=0
        if j==1
           D0=[D0;Train_data(i,:)];
        elseif j==2
           D1=[D1;Train_data(i,:)];
        elseif j==3
           D2=[D2;Train_data(i,:)];
        elseif j==4
           D3=[D3;Train_data(i,:)];
        elseif j==5
           D4=[D4;Train_data(i,:)];
        elseif j==6
           D5=[D5;Train_data(i,:)];
        elseif j==7
           D6=[D6;Train_data(i,:)];
        elseif j==8
           D7=[D7;Train_data(i,:)];
        elseif j==9
           D8=[D8;Train_data(i,:)];
        elseif j==10
           D9=[D9;Train_data(i,:)];
        end
    end
   end
end

Ttotal=size(D0,1)+size(D1,1)+size(D2,1)+size(D3,1)+size(D4,
1)+...
size(D5,1)+size(D6,1)+size(D7,1)+size(D8,1)+size(D9,1);

D{1}=D0;D{2}=D1;D{3}=D2;D{4}=D3;D{5}=D4;D{6}=D5;D{7}=D6;D{8
}=D7;D{9}=D8;D{10}=D9;
```

The mean vectors is computed by the following:

```
%% mean
for p=1:10
    mu0=zeros(62,1);
    for i=1:size(D{p},1)
            x=D{p}(i,:)';
            mu0=mu0+ x;
    end
    MU{p}=mu0/size(D{p},1);
end
```

The covariance matrices are calculated by the following:

```
%% Covariance
for p=1:10
    n=size(D{p},1);
    sum0=zeros(62,62);

    for i=1:n
         x= (D{p}(i,:))';
         sum0=sum0+(x-MU{p})*(x-MU{p})';
    end
COV{p}=(1/n)*sum0;
end
```

The determinant of covariance matrices are calculated by

```
for i=1:10
    det(COV{i})
end
```

because the `det(COV{2})` is zero, the LDA method is used for calculating `COV{2}`as follows:

```
sigma_p=zeros(62,62);
for i=1:10
   sigma_p=sigma_p+(size(D{i},1))*COV{i};
end
COV{2}=(1/(5000-10))*sigma_p;
```

The calculation of Gaussian distribution function of the test dataset and maximization of posterior probabilities for Bayes classification are accomplished from the following loop:

```
%% Test Data
d=size(Test_data,2);
for i=1:size(Test_data,1)

    x=(Test_data(i,:))';
    for j=1:10
        sigma=COV{j};
        mu=MU{j};
        Px_w(j)=Gaussian_Distribution(x,mu,sigma,d);

        a(j)=Px_w(j)*P_w(j);
    end

    b=max(a);
    class(i,1)=find(a==b)-1;
end
```

The distribution function of each sample is calculated by Gaussian_Distribution.m file as follows:

```
function Px_w=Gaussian_Distribution(x,mu,sigma,d)

 g=((2*pi)^(d/2))*((det(sigma))^(1/2));

 Px_w=(1/g)*exp((-1/2)*(x-mu)'*inv(sigma)*(x-mu));

end
```

The validation of the method and the classification is done by constructing the confusion matrix as the following:

```
Conf=zeros(10,10);
for i=1:size(Test_data,1)

    if Test_label(i)==class(i)

       j=Test_label(i)+1;
       Conf (j,j)= Conf (j,j)+1;
    else

       Conf (class(i)+1,Test_label(i)+1)= Conf
(class(i)+1,Test_label(i)+1)+1;

    end

end

for i=1:10
WW(i)=sum(Conf (i,:));
```

```
end
```

The accuracy and error of the classification are obtained by the following:

```
for i=1:10
    Accuracy(i)=( Conf  (i,i)/omega_test(i))*100;

    error(i)=100-Accuracy(i);
end
Total_error=sum(error)/10;
```

And at the end of the m-file, the plots of recognition accuracy and errors are done by:

```
%% plot
h1=figure('DefaultAxesFontName', font);
axes;
Xc=1:10;
bar(Xc,Accuracy)
grid on
xlabel('Class Number','fontsize',14)
ylabel('Recognition Percentage','fontsize',14)

h2=figure('DefaultAxesFontName', font);
axes;
Xc=1:10;
bar(Xc,error)
grid on
xlabel('Class Number','fontsize',14)
ylabel('Error Percentage','fontsize',14)
```

Problem #2

Apply MAP estimation to calculate p(θ|D) and p(X|D) Under the assumption that p(X|θ) follows a normal distribution $p(X|\theta) = p(X|\mu) \sim N(\mu, \sigma^2)$and $p(\mu) \sim N(\mu_0, \sigma_0^2)$ and Proof the equation below: Data : a set D of N (i.i.d) training samples X6, X0,…, X8.

$$p(\mu|D) = \frac{p(D|\mu)p(\mu)}{p(D)} = \prod_{k=1}^{N} p(X_k|\mu)p(\mu) = \alpha \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{[-\frac{1}{2}\frac{(\mu-\mu_N)^2}{\sigma_N^2}]}$$

* $\bar{\mu} = \frac{1}{N}\sum_{k=1}^{N} X_k$

* $\mu_N = \frac{N\sigma_0^2}{N\sigma_0^2+\sigma^2}\bar{\mu} + \frac{\sigma^2}{N\sigma_0^2+\sigma^2}\mu_0$

* $\sigma_N^2 = \frac{\sigma^2\sigma_0^2}{N\sigma_0^2+\sigma^2}$

$$P(\mu|D) = \prod_{k=1}^{N} p(x_k|\mu)\,p(\mu) = \prod_{k=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2}\left(\frac{x_k-\mu}{\sigma}\right)^2\right] \times \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{1}{2}\left(\frac{\mu-\mu_0}{\sigma_0}\right)^2\right)$$

$$= \underbrace{\frac{1}{(\sqrt{2\pi\sigma^2})^N\sqrt{2\pi\sigma_0^2}}}_{A} \exp -\frac{1}{2}\underbrace{\left[\sum_{k=1}^{N}\frac{(x_k-\mu)^2}{\sigma^2} + \frac{(\mu-\mu_0)^2}{\sigma_0^2}\right]}_{B}$$

$$B = \frac{\sum x_k^2}{\sigma^2} + \frac{\sum \mu^2}{\sigma^2} - 2\frac{\sum x_k \mu}{\sigma^2} + \frac{\mu^2}{\sigma_0^2} + \frac{\mu_0^2}{\sigma_0^2} - 2\frac{\mu\mu_0}{\sigma_0^2} \xRightarrow{\bar{\mu} = \frac{1}{N}\sum x_k}$$

$$B = \left(\frac{\sum x_k^2}{\sigma^2} + \frac{\mu_0^2}{\sigma_0^2}\right) + \frac{N\mu^2}{\sigma^2} + \frac{\mu^2}{\sigma_0^2} - \frac{2N\bar{\mu}\mu}{\sigma^2} - 2\frac{\mu\mu_0}{\sigma_0^2}$$

$$\Rightarrow \left\{\begin{array}{l} \mu_N = \frac{N\sigma_0^2}{N\sigma_0^2+\sigma^2}\bar{\mu} + \frac{\sigma^2}{N\sigma_0^2+\sigma^2}\mu_0 = \frac{n}{\sigma^2}\sigma_N^2\bar{\mu} + \frac{1}{\sigma_0^2}\sigma_N^2\mu_0 \\ \sigma_N^2 = \frac{\sigma^2\sigma_0^2}{N\sigma_0^2+\sigma^2} \end{array}\right\} \Rightarrow$$

$$\frac{\mu_N}{\sigma_N} = \frac{N}{\sigma^2}\bar{\mu} + \frac{1}{\sigma_0^2}\mu_0$$

$$\Rightarrow \quad B = \left(\frac{\Sigma x_k^2}{\sigma^2} + \frac{\mu_0^2}{\sigma_0^2} \right) + \frac{\mu^2}{\sigma_N^2} - 2\frac{\mu\,\mu_N}{\sigma_N^2} + \frac{\mu_N^2}{\sigma_N^2} - \frac{\mu_N^2}{\sigma_N^2}$$

$$\Rightarrow B = \left(\frac{\Sigma x_k^2}{\sigma^2} + \frac{\mu_0^2}{\sigma_0^2} - \frac{\mu_N^2}{\sigma_N^2} \right) + \left(\frac{\mu - \mu_N}{\sigma_N} \right)^2$$

$$A = \frac{1}{(\sqrt{2\pi})^N \, (\sqrt{\sigma^2})^N \sqrt{\sigma_0^2} \times \sqrt{2\pi}} \Rightarrow \frac{1}{(\sqrt{\sigma^2})^N \sqrt{\sigma_0^2}} = Q \frac{1}{\sqrt{\sigma_N^2}}$$

$$\Rightarrow \quad Q = \frac{\sqrt{\sigma_N^2}}{(\sqrt{\sigma^2})^N \sqrt{\sigma_0^2}} \Rightarrow p(\mu|D) = \alpha \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{\left[-\frac{1}{2}\frac{(\mu-\mu_N)^2}{\sigma_N^2} \right]}$$

$$\alpha = \frac{1}{(\sqrt{2\pi})^N} \frac{\sqrt{\sigma_N^2}}{(\sqrt{\sigma^2})^N \sqrt{\sigma_0^2}} \times \exp\left(\frac{\Sigma x_k^2}{\sigma^2} + \frac{\mu_0^2}{\sigma_0^2} - \frac{\mu_N^2}{\sigma_N^2} \right) \times \left(\frac{-1}{2}\right)$$

$$\Rightarrow \quad \alpha = \frac{1}{(\sqrt{2\pi})^N} \frac{\sigma^2\sigma_0^2}{(\sqrt{\sigma_0^2})^N \sqrt{\sigma_0^2 (N\sigma_0^2 + \sigma^2)}} \exp\left(-\frac{1}{2}\left(\frac{\Sigma x_k^2}{\sigma^2} + \frac{\mu_0^2}{\sigma_0^2} - \frac{N\sigma_0^2 + \sigma^2}{\sigma^2\sigma_0^2}\mu_N^2 \right)\right)$$

Appendix

All MATLAB codes are located in folder Codes.

The MATLAB files of Problem 1 are Project2_main.m and Gaussian_Distribution.m.

The main file is Project2_main.m. The function of Gaussian distribution is Gaussian_Distribution.m.