HW7

Xinhao du

A13.3

a. $A = \begin{bmatrix} x_M & x_{M-1} & \cdots & x_1 \\ x_{M+1} & x_M & \cdots & x_2 \\ \vdots & \vdots & \ddots & \vdots \\ x_{N-1} & x_{N-2} & \cdots & x_{N-M} \end{bmatrix}, b = \begin{bmatrix} x_{M+1} \\ x_{M+2} \\ \vdots \\ x_N \end{bmatrix}$

The dimension of matrix A is (N-M)*M, and b is (N-M)*1

b. First, we run the code which is showed in the picture.

mis7.jl

```
include("time_series_data.jl")
using LinearAlgebra
using Distances
for M in 2:12
    T1=toeplitz(x_train,M)
    T=zeros(100-M,M)
    for i in 1:100-M
        for j in 1:M
            T[i,j]=T1[i+M-1,j]
        end
    end
    b=x_train[M+1:100]
    btrim=T\b
    display(btrim)
    println("J-train")
    dist=sqrt(euclidean(T*btrim,b))
    J=dist/(100-M)
    println(J)
    T2=toeplitz(x_test,M)
    Test=zeros(100-M,M)
    for i in 1:100-M
        for j in 1:M
            Test[i,j]=T2[i+M-1,j]
        end
    end
    b_test=x_test[M+1:100]
    println("J-test")
    distest=sqrt(euclidean(Test*btrim,b_test))
    Jtest=distest/(100-M)
    println(Jtest)
end
```

Then we can get the minimum value of J_test [0.013831634116043443]

when M is 6

```
data given is x_train and x_test
M is 2
J-train
0.013416408219730211
J-test
0.014244010095376045
M is 3
J-train
0.013497572936886979
J-test
0.014231032891108095
M is 4
J-train
0.012899498071768802
J-test
0.013986379516551862
M is 5
J-train
0.012987694733071563
J-test
0.01386513322995077
M is 6
J-train
0.012899375009750016
J-test
0.013831634116043443
M is 7
J-train
0.012532541184970677
J-test
0.014468016638241325
M is 8
J-train
0.012520701976085062
J-test
0.014518488186580377
M is 9
J-train
0.012614566978130943
J-test
0.014890746176639548
M is 10
J-train
0.012705846209248223
J-test
0.015078694335989634
M is 11
J-train
0.012846576262633079
J-test
0.015252635692332218
M is 12
J-train
0.012907567707791846
J-test
0.01559439701969854
xinhaodu@xinhaodeMacBookPro julia %
```

A13.5

The matrix of A and vector of b can be expressed as follows:

$$A = \begin{bmatrix} z_2 & z_1 & z_2 z_1 \\ z_3 & z_2 & z_3 z_2 \\ \vdots & \vdots & \vdots \\ z_{T-1} & z_{T-2} & z_{T-1} z_{T-2} \end{bmatrix}, b = \begin{bmatrix} z_3 \\ z_4 \\ \vdots \\ z_T \end{bmatrix}$$

A13.8

For the given data set, the function $\hat{y}$ have the following form.

$\hat{y} = \theta_1 + \theta_2(-2) + \theta_3(0)$ for $x \le -2$

$\hat{y} = \theta_1 + \theta_2(-1) + \theta_3(0)$ for $-2 < x \le -1$

$\hat{y} = \theta_1 + \theta_2(0) + \theta_3(0)$ for $-1 < x \le 0$

$\hat{y} = \theta_1 + \theta_2(0) + \theta_3(1)$ for $0 < x \le 1$

$\hat{y} = \theta_1 + \theta_2(0) + \theta_3(1)$ for $x > 1$

Then we have,

$$A = \begin{bmatrix} 1 & -2 & 0 \\ 1 & -1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 3 \\ 1 \\ 1 \\ 2 \\ 1 \end{bmatrix}$$

$$\|A\theta - b\|^2 = \theta^T (A^T A)\theta - 2b^T A\theta + b^T b$$

A13.9

For the given sum square error formula $E(\theta) = \sum_{t=3}^{N-2} (\hat{z}_t - z_t)^2$, we can have

$$E(\theta) = (\theta_1 z_1 + \theta_2 z_2 + \theta_3 z_4 + \theta_4 z_5 - z_3)^2 + (\theta_1 z_2 + \theta_2 z_3 + \theta_3 z_5 + \theta_4 z_6 - z_4)^2 + \cdots + (\theta_1 z_{N-4} + \theta_2 z_{N-3} + \theta_3 z_{N-1} + \theta_4 z_N - z_{N-2})^2$$

This can be expressed in matrix form as $A\theta - b = 0$, and we also know $N=9$

$$A = \begin{bmatrix} z_1 & z_2 & z_4 & z_5 \\ z_2 & z_3 & z_5 & z_6 \\ z_3 & z_4 & z_6 & z_7 \\ z_4 & z_5 & z_7 & z_8 \\ z_5 & z_6 & z_8 & z_9 \end{bmatrix}$$

$$b = \begin{bmatrix} z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \end{bmatrix}$$

$$\|A\theta - b\|^2$$

$$= (A\theta - b)^T (A\theta - b)$$

$$= \theta^T (A^T A)\theta - 2\theta^T (A^T b) + b^T b$$

A14.3

```
using LinearAlgebra
include("iris_flower_data.jl")
include("iris_multiclass_helpers.jl")
function get_regression_classifiers(X,y,k)
    S=size(X)[2]
    one=ones(S)
    A=[X' one]
    y1=2(y .==k) .-1
    theta=pinv(A)*y1
    return theta,errorRate(X,y,k,theta,S)
end
function errorRate(X,y,k,theta,S)
    one=ones(S)
    A=[X' one]
    y_hat=sign.(A*theta)
    y1=2(y .==k) .-1
    correct=sum(y_hat.==y1)
    return 1-correct/S
end
X_train = X[:,1:100];
Y_train = y[1:100];
X_test = X[:,101:150];
Y_test = y[101:150];


theta1, train1=get_regression_classifiers(X_train,Y_train,1);
theta2, train2=get_regression_classifiers(X_train,Y_train,2);
theta3, train3=get_regression_classifiers(X_train,Y_train,3);
test1=errorRate(X_test,Y_test,1,theta1,50);
test2=errorRate(X_test,Y_test,2,theta2,50);
test3=errorRate(X_test,Y_test,3,theta3,50);

println("error rate of train1:",train1)
println("error rate of train2:",train2)
println("error rate of train3:",train3)
println("error rate of test1:",test1)
println("error rate of test2:",test2)
println("error rate of test3:",test3)

one_train=ones(size(X_train)[2]);
one_test=ones(size(X_test)[2]);
theta=[theta1 theta2 theta3];
A_train=[X_train' one_train];
A_test=[X_test' one_test];
twoy_train=argmax_by_row(A_train*theta);
twoy_test=argmax_by_row(A_test*theta);
println("We can get the confusion matrix C on train set is :")
C_train=confusion_matrix(twoy_train,Y_train)
display(C_train)
println("We can get the confusion matrix C on test set is :")
C_test=confusion_matrix(twoy_test,Y_test)
display(C_test)
```

After we run the code, we can know that the error rate of train dataset is (0, 0.28, 0.099) and test dataset is (0, 0.24, 0.02) respectively. Also, the confusion matrix C on test set and train set are shown in the picture.

```
error rate of train1:0.0
error rate of train2:0.28
error rate of train3:0.09999999999999998
error rate of test1:0.0
error rate of test2:0.24
error rate of test3:0.020000000000000018
We can get the confusion matrix C on train set is :
3×3 Matrix{Float64}:
 29.0   0.0   0.0
  0.0  24.0   5.0
  0.0  11.0  31.0
We can get the confusion matrix C on test set is :
3×3 Matrix{Float64}:
 20.0  0.0   0.0
  1.0  9.0   0.0
  0.0  6.0  14.0
```

A14.4

a. False, the effect on the false positive rate will depend on the distribution of the feature vectors and the relationship between x and y.

b. Ture, if replacing β with zero will pass through the point (0,v). This means that all data points with y = -1 that have $x^T \beta < 0$ will be correctly classified as negative, which will reduce the false positive rate.

c. False. Halving v will shift the hyperplane up by v/2 units along the y-axis. This means that some data points with y = -1 that were below the hyperplane may now be above it, which will increase the false positive rate.

d. True. Halving β will make the hyperplane less steep and closer to horizontal. This means that all data points with y = -1 that have $x^T\beta < 0$ will still be below the hyperplane and correctly classified as negative, which will not increase the false positive rate.

A14.5

To reduce the false negative rate, we should increase the offset v so that the function moves up and crosses more data points that are above the x-axis, which will increase the number of positive predictions.