**Final Project**

Ying-Yu Huang (yingyu010365@gmail.com)

**Question1:**

The original data has 10000 cases, and I choose 1200 cases from the data, and the data has 12 predictive attributes: tau1, tau2, tau3, tau4, p1, p2, p3, p4, g1, g2, g3, g4,

tau[] is the reaction time of participant

p[] is nominal power consumed(negative)/produced(positive)

g[] is the coefficient (gamma) proportional to price elasticity

They are all continuous.

We use these features to predict the stability of the Electrical Grid. If it is negative, then the electrical grid is unstable. If it’s positive, then the electrical grid is stable.

**The practical impact**:

This data is to predict Electrical Grid Stability, and make sure that the ability of the electric grid to deliver electricity to customers without degradation or failure.

The mean of the features:



The standard deviation of the features:



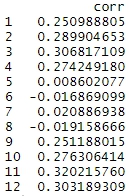
The mean of Y:

0.01560539

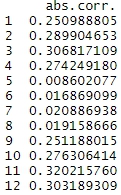
The standard deviation of Y:

0.0363883

the empirical correlations cor(X1, Y) ... cor(Xp,Y)



their absolute values C1 ... Cp

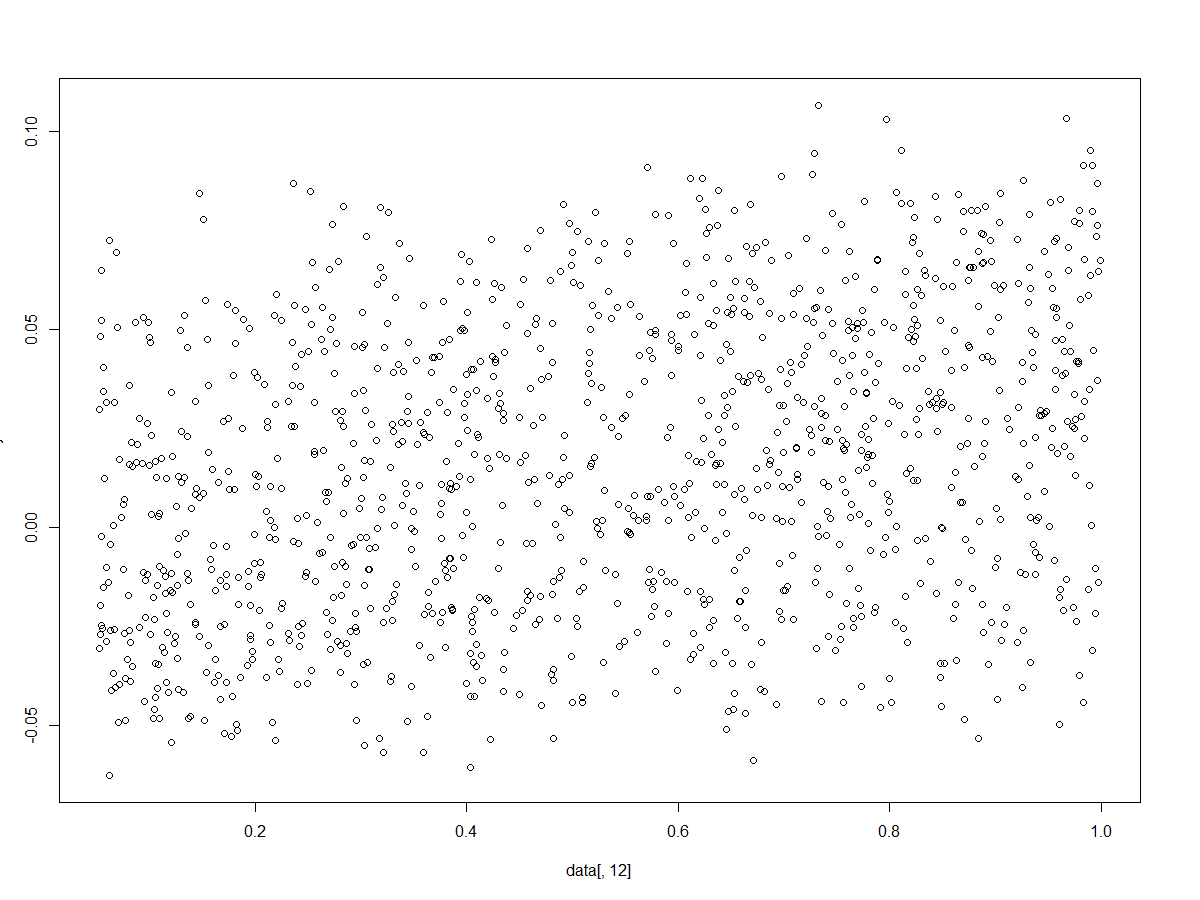


the 3 largest values among C1 ... Cp, to be denoted Cu > Cv > Cw which are

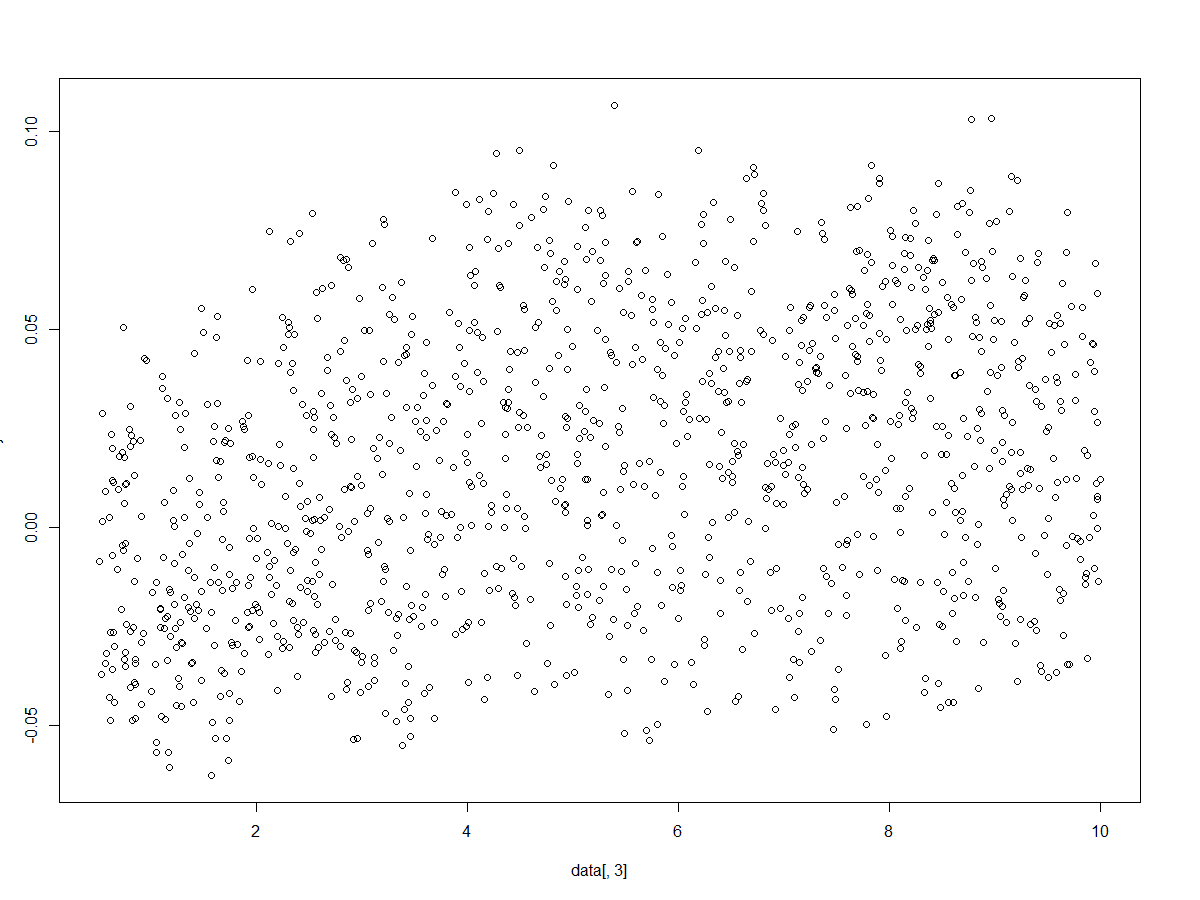
display separately the 3 scatter plots:

C11=0.32>C3=0.31>C12=0.3

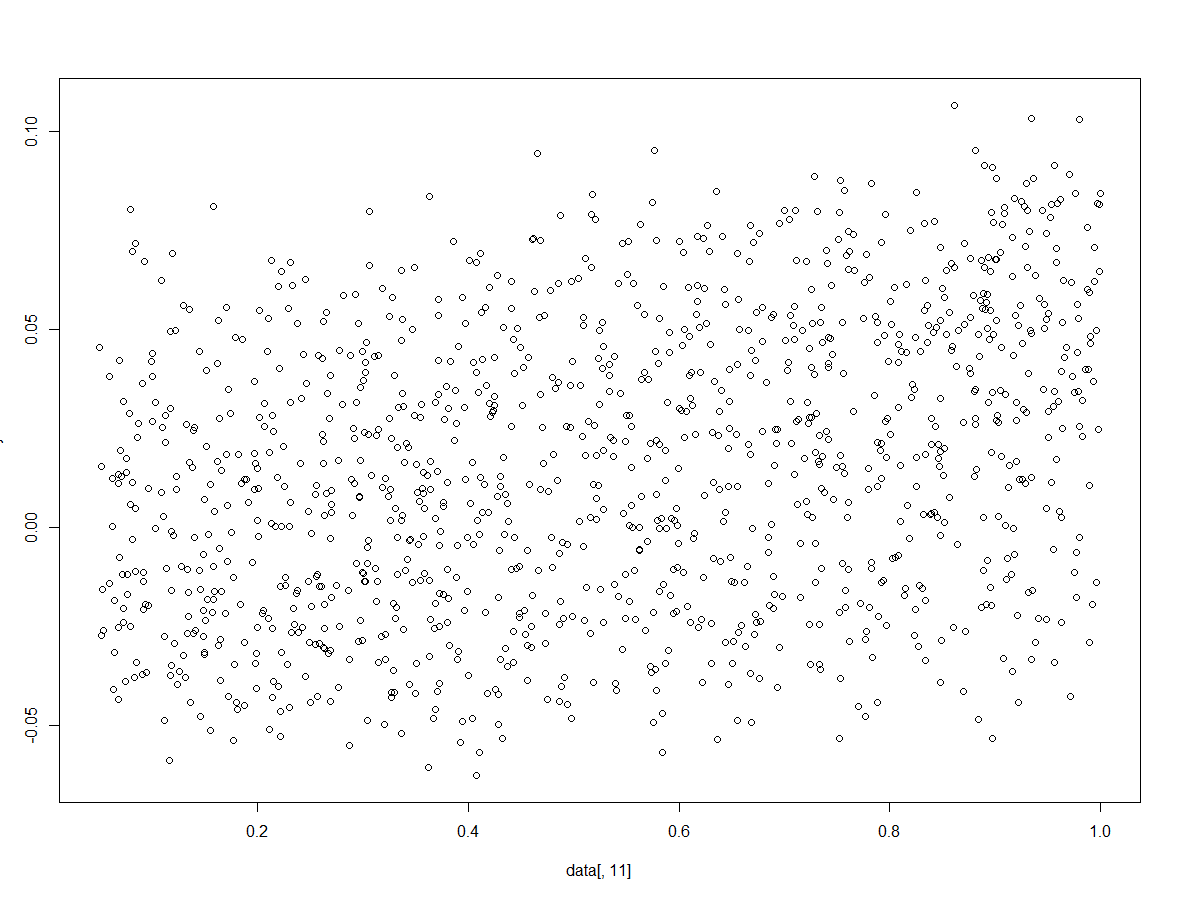
The scatter plot of (X12(j), Yj)



The scatter plot of (X3(j), Yj)



The scatter plot of (X11(j), Yj)

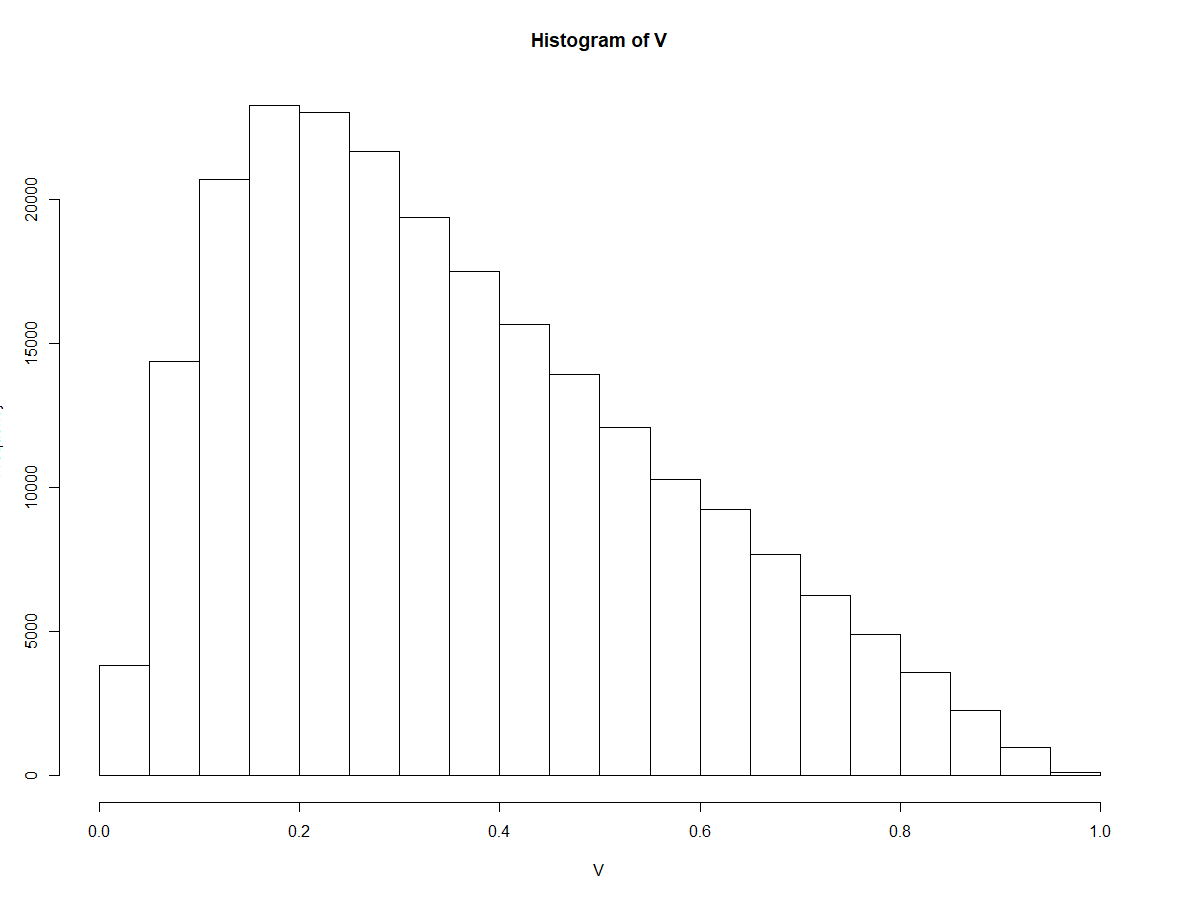


**Interpretation:**

The three largest correlation value is 0.32, 0.31, 0.3, they are all small, there is a weak correlation between these 3 features and Y. Based on the above scatter plot, we can see there are no linear correlation between each other.

**Queation2:**

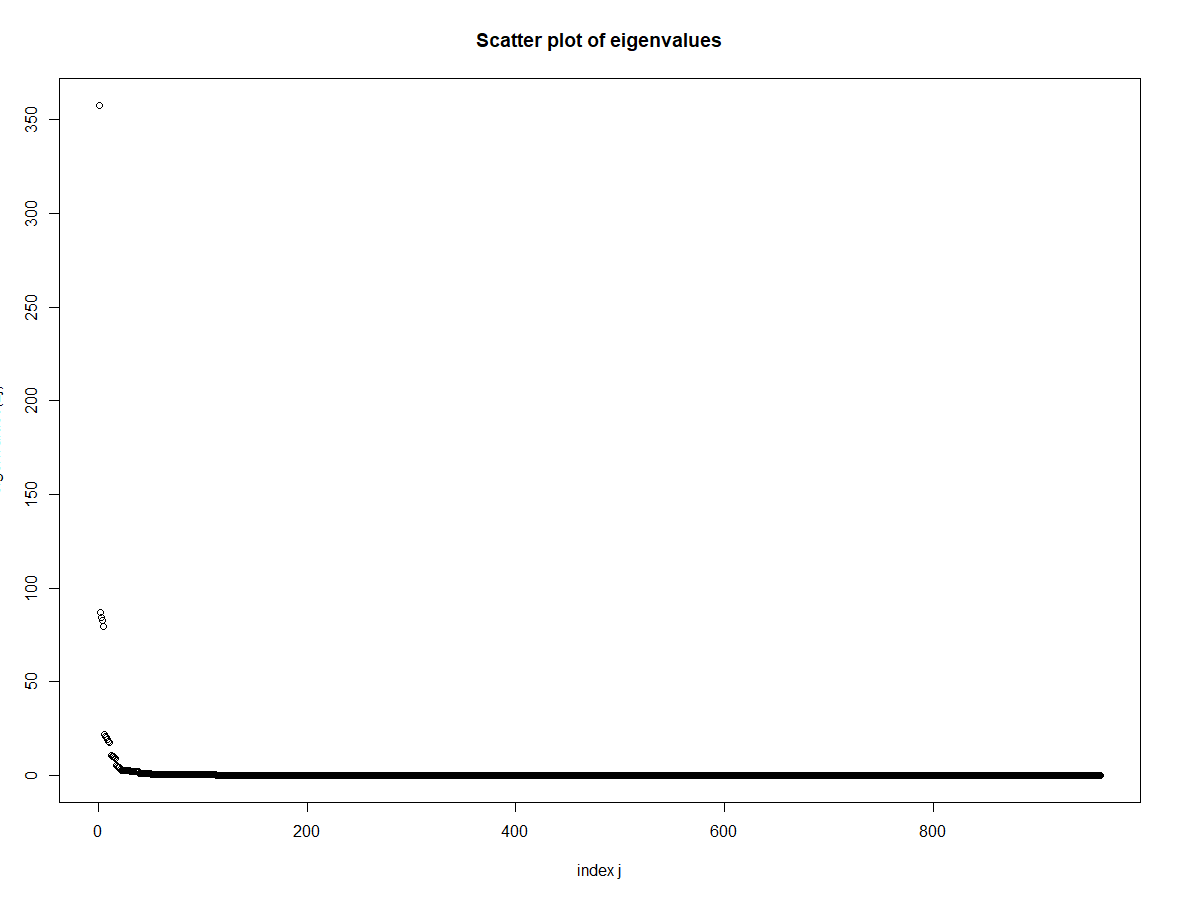
Plot the histogram of the 10000 numbers Dij:



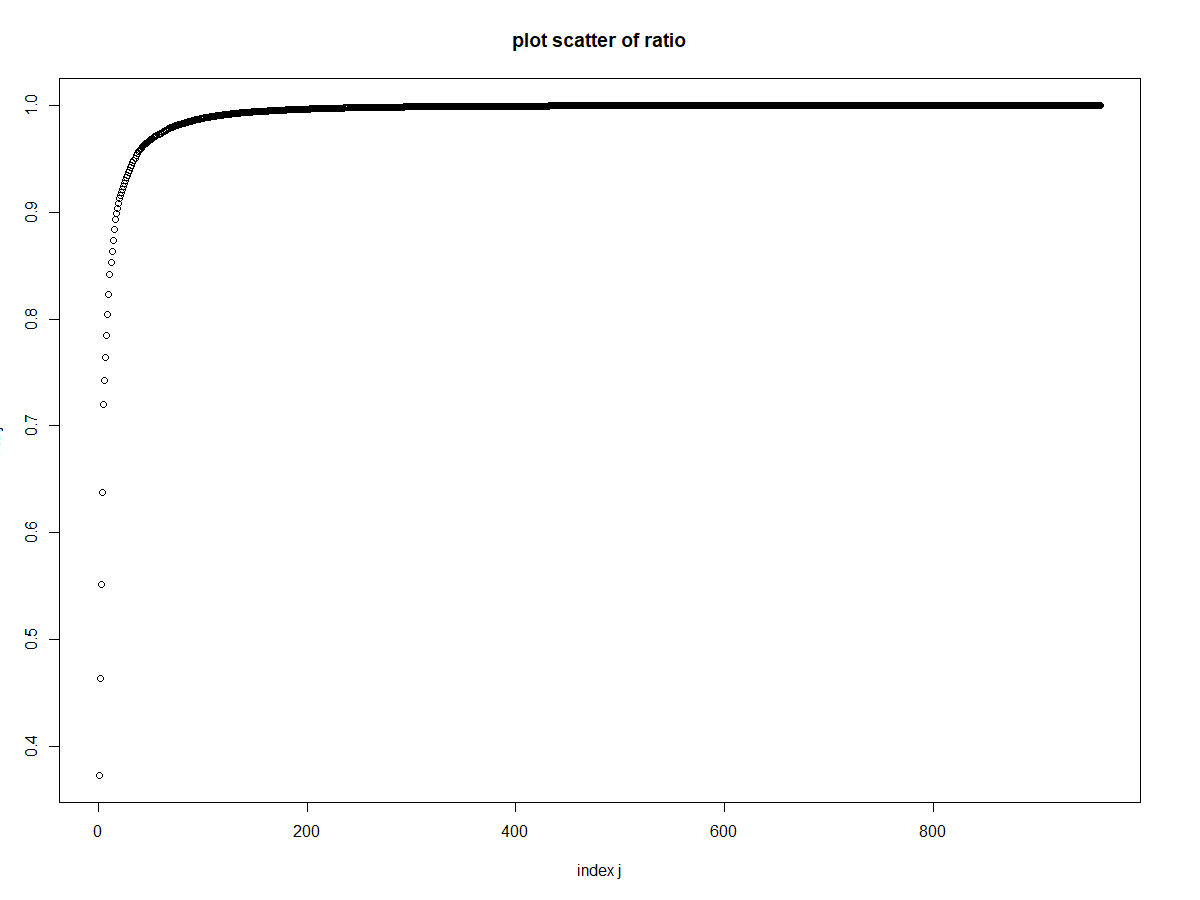
Compute q =10% quantile of the 10000 numbers Dij, q=51

Set gamma = 1/q=1/51

**Plot Lj versus j**



**Plot the increasing ratios RATj= (L1 + ... + Lj)/(L1+ ... + Lm)**



**Identify the smallest j such that RATj ≥ 95% and set λ = Lj**

j=35, λ= 2.008661

**Compare these two RMSE values**

RMSEtrain= 0.0202

RMSEtest= 0.02

**compute their ratios RMSE/ avy**

RMSEtrain/avy= 61.08%

RMSEtest/avy= 61.85%

**Interpretation:**

The KRR model with parameters (gamma=1/51, lambda=2) does not perform well. The ratio RMSEtrain/avy =61.08%, and the ratio of RMSEtest/avy= 61.85%. They are both very high. Thus, we need to optimize the parameters.

**Queation3:**

First, I change only gamma:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **gamma** | **lambda** | **RMSEtrain** | **RMSEtest** | **RMSEtrain/avy** | **RMSEtest/avy** |
| 0.001 | 1/55 | 0.033 | 0.031 | 98.34% | 96.36% |
| 0.01 | 1/55 | 0.019 | 0.022 | 58.53% | 68.81% |
| 0.05 | 1/55 | 0.019 | 0.02 | 58.01% | 63.11% |
| 0.02 | 1/55 | 0.02 | 0.02 | 60.95% | 61.79% |

then change lambda:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **gamma** | **lambda** | **RMSEtrain** | **RMSEtest** | **RMSEtrain/avy** | **RMSEtest/avy** |
| 0.02 | 0.01 | 0.0056 | 0.0096 | 16.91% | 29.88% |
| 0.02 | 5 | 0.024 | 0.023 | 71.81% | 71.61% |
| 0.02 | 0.0001 | 0.0012 | 0.011 | 3.57% | 34.13% |
| 0.02 | 0.001 | 0.0032 | 0.0098 | 9.67% | 30.47% |

The best parameter gamma=0.02, lam=0.01

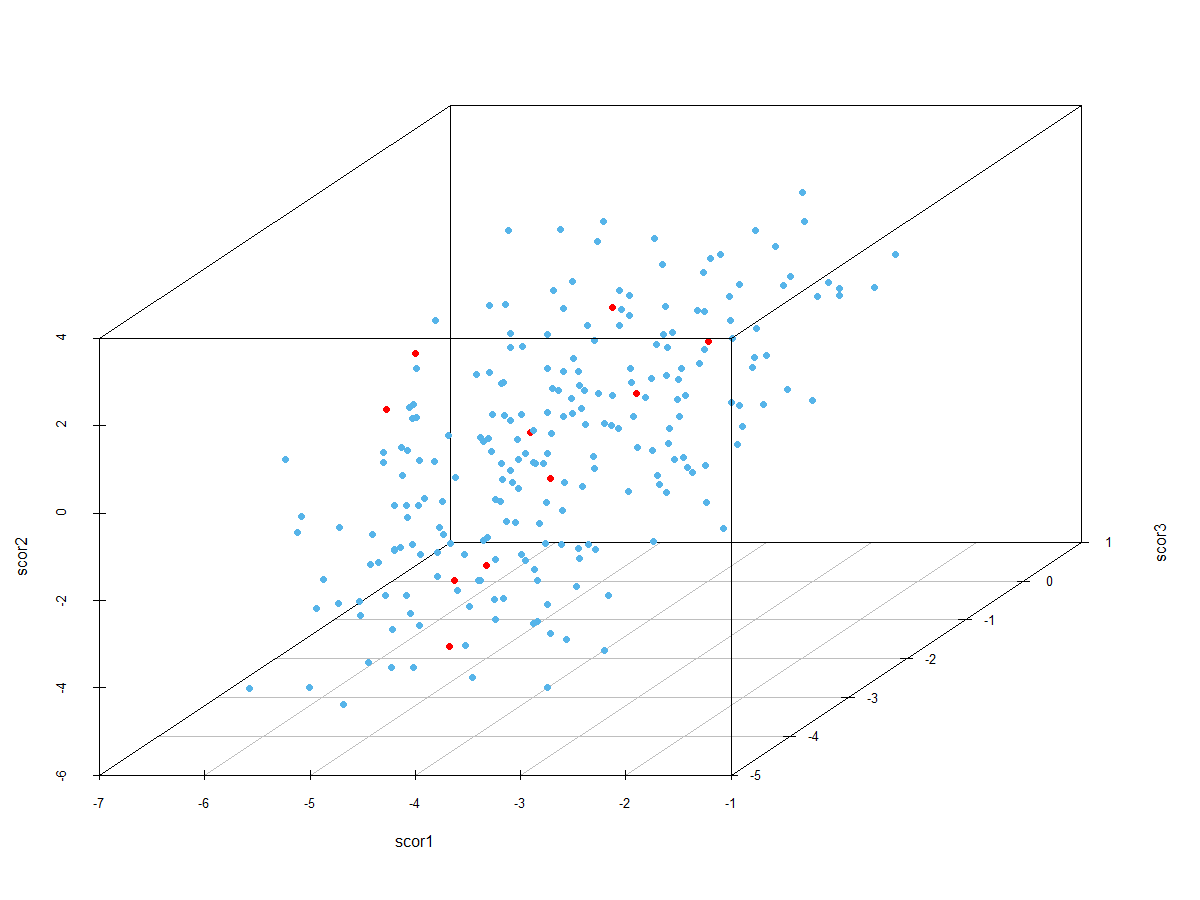
Compare to the other, this one has the smaller ratios of RMSE/avy and the RMSEtrain/avy and RMSEtest/avy are much closer than the others.

Compare to the result of question 2 that the ratios are more than 61%, the result after optimize the parameter is much better.

**Identify the 10 cases in the TEST set for which the squared prediction error is the largest**

Case: 214 208 18 169 22 171 37 101 96 34

**Visualize the 10 cases by performing a PCA analysis and projecting all the TEST cases onto the first 3 principal eigenvectors of the PCA correlation matrix**



The red dots represent the 10 cases, and blue bots represent all the TEST cases.

We cannot see the significant difference between the 10 cases and the other cases on the 3 dimensional plot.

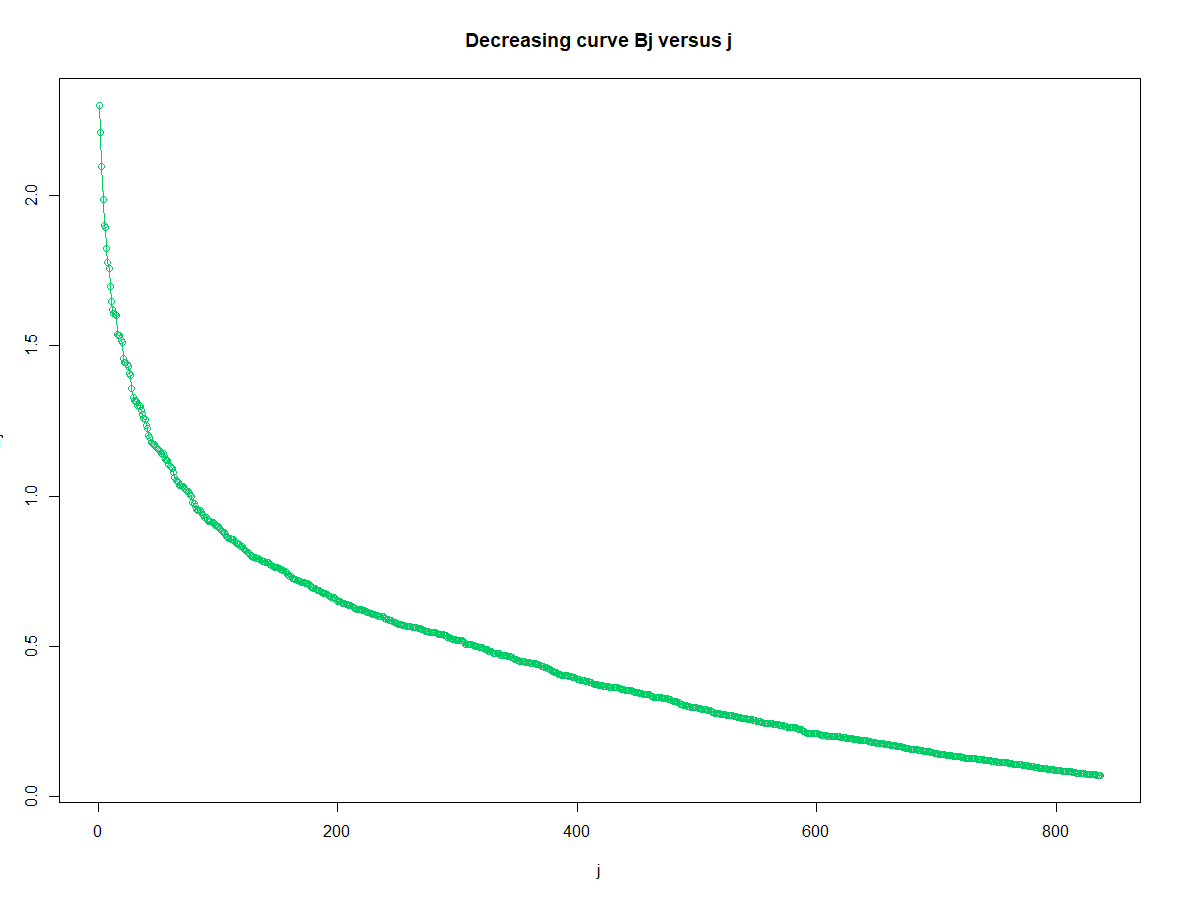
**Interpretation:**

The prediction does not do well, the RMSE is high, so we cannot accurately predict if the electrical grid is stable or not. So, we need to use other model to do the further research. Because the stability of the electrical grid is very important in our daily life, if it is not stable, then it may increase the frequency of outages and bring many problems.

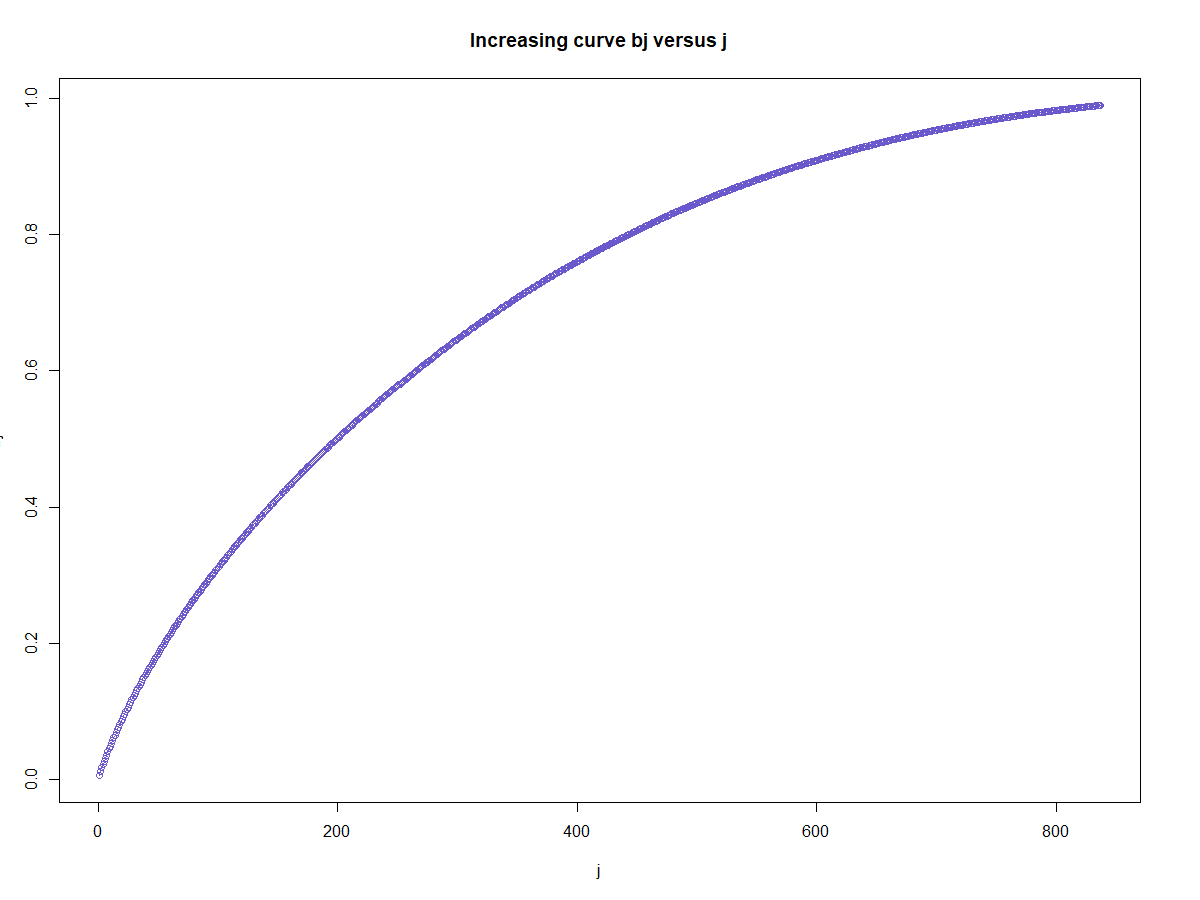
**Question4:**

The smallest j=837

**plot the decreasing curve Bj versus j**



**plot the increasing curve bj versus j**



THR = Bj =0.0711

**The RMSE values**

RMSEtrain= 0.104

RMSEtest= 0.106

**The RMSE/ avy**

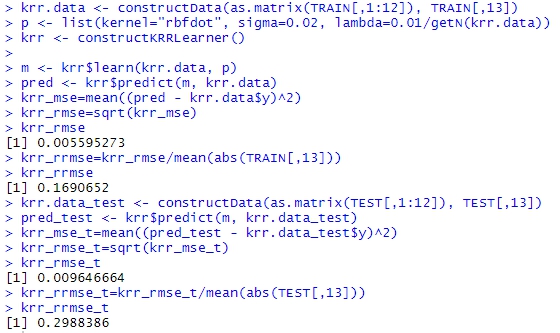
RMSEtrain/avy= 3.145

RMSEtest/avy= 3.274

**Interpretation:**

We get RMSEtrain/avy=0.169, and RMSEtest/avy=0.299 from the original formula, and we get RMSEtrain/avy=3.145, and RMSEtest/avy=3.274 from the reduced formula. So, the reduced formula is not suitable.

**Question5:**

****

The result is same as Question 3 with the best parameter.

**Code:**

data=read.csv("C:/Users/yingy/Desktop/11 data mining/final project/data2.csv")

attach(data)

colnames(data)[1] <- "tau1"

y = data[,13]

#display its mean and standard deviation

mean<- sapply(data[,1:12],mean)

mean

sd<- sapply(data[,1:12],sd)

sd

mean(data[,13])

sd(data[,13])

#Split the data set DS into a training set TRAIN and a test set TEST, with respective proportions 80% , 20%

n <- 1200

ntrain <- round(n\*0.8)

set.seed(1)

tindex <- sample(n, ntrain)

TRAIN<- data[tindex,]

TEST<- data[-tindex,]

#Compute the empirical correlations cor(X1, Y) ... cor(Xp,Y) and their absolute values C1 ... Cp

corr <- numeric(0)

for (i in 1:12){

corr[i] =cor(data[,i],data[,13])

print (corr)

print(abs(corr))

}

data.frame(corr)

CORR=data.frame(abs(corr))

#compute the 3 largest values among C1 ... Cp

order(CORR)

#display separately the 3 scatter plots (Xu(j), Yj) , (Xv(j), Yj) , (Xw(j), Yj) where j= 1...n

plot(data[,12],y)

plot(data[,3],y)

plot(data[,11],y)

#Q2

set.seed(100)

list1=sample(1:960, 100, replace=T)

list2=sample(1:960, 100, replace=T)

#For all i in List 1 and all j in List2 compute Dij = ||X(i) -X(j)||

D = matrix(nrow = length(list1), ncol = length(list2))

for (i in 1:length(list1)){

for (j in 1:length(list2)){

D[i,j]=list1[i]-list2[j]

}

}

print(D)

V = abs(as.vector(t(D)))

print(V)

#Plot the histogram of the 10000 numbers Dij

hist(V)

#Compute q =10% quantile of the 10000 numbers Dij

quantile(V, probs = c(0.1))

#Set gamma = 1/q

gamma=1/51

#krr function

radial\_k <- function (x, x.prime, g) {

r <- apply( x.prime

, 1

, function(a)

colSums(apply ( x

, 1

, function(b)

(b - a)^2)

)

);

return(exp(-g\*r));

};

#G matrix

G <- radial\_k(TRAIN[,1:12],TRAIN[,1:12],1/51)

#compute eigenvalues

eigenvalues.1 <- eigen(G,symmetric = TRUE,only.values = TRUE)

e\_values.1 <- eigenvalues.1$values

plot(c(1:960),e\_values.1,main="Scatter plot of eigenvalues",xlab="index j",ylab="eigenvalues(Lj)")

#determine lambda

Rat=0

for (j in 1:960){

Rat[j]<-sum(e\_values.1[1:j])/sum(e\_values.1)

}

#plot increasing

plot(c(1:960),Rat,main="plot scatter of ratio",xlab="index j",ylab="Ratj")

#identify smallest j

for (j in 1:960){

if (Rat[j]>0.95){

r=j

break

}

}

r

Rat[r]

lam=e\_values.1[r]

M=G+(lam)\*diag(960)

M\_i=solve(M)

y=TRAIN[,13]

A=y%\*%M\_i

A

#train

Y.hat.train <- A%\*%G

Y.hat.train

MSE.TRAIN <- mean((Y.hat.train-TRAIN[,13])^2)

RMSE.TRAIN <- sqrt(MSE.TRAIN)

RMSE.TRAIN

rRMSE.TRAIN <- RMSE.TRAIN/mean(abs(TRAIN[,13]))

rRMSE.TRAIN

#test

V <- radial\_k(TRAIN[,1:12],TEST[,1:12],1/51)

Y.hat.TEST <- A%\*%V

Y.hat.TEST

MSE.TEST <- mean((Y.hat.TEST-TEST[,13])^2)

RMSE.TEST <- sqrt(MSE.TEST)

RMSE.TEST

rRMSE.TEST <- RMSE.TEST/mean(abs(TEST[,13]))

rRMSE.TEST

#Q3

#set gamma=1000

gamma1=1000

G1 <- radial\_k(TRAIN[,1:12],TRAIN[,1:12],gamma1)

M1=G1+(lam)\*diag(960)

M1\_i=solve(M1)

y=TRAIN[,13]

A1=y%\*%M1\_i

A1

#train

Y.hat.train1 <- A1%\*%G1

Y.hat.train1

MSE.TRAIN1 <- mean((Y.hat.train1-TRAIN[,13])^2)

RMSE.TRAIN1 <- sqrt(MSE.TRAIN1)

RMSE.TRAIN1

rRMSE.TRAIN1 <- RMSE.TRAIN1/mean(abs(TRAIN[,13]))

rRMSE.TRAIN1

#test

V1 <- radial\_k(TRAIN[,1:12],TEST[,1:12],gamma1)

Y.hat.TEST1 <- A1%\*%V1

Y.hat.TEST1

MSE.TEST1 <- mean((Y.hat.TEST1-TEST[,13])^2)

RMSE.TEST1 <- sqrt(MSE.TEST1)

RMSE.TEST1

rRMSE.TEST1 <- RMSE.TEST1/mean(abs(TEST[,13]))

rRMSE.TEST1

#set gamma=0.001

gamma2=0.0001

G2 <- radial\_k(TRAIN[,1:12],TRAIN[,1:12],gamma2)

M2=G2+(lam)\*diag(960)

M2\_i=solve(M2)

y=TRAIN[,13]

A2=y%\*%M2\_i

A2

#train

Y.hat.train2 <- A2%\*%G2

Y.hat.train2

MSE.TRAIN2 <- mean((Y.hat.train2-TRAIN[,13])^2)

RMSE.TRAIN2 <- sqrt(MSE.TRAIN2)

RMSE.TRAIN2

rRMSE.TRAIN2 <- RMSE.TRAIN2/mean(abs(TRAIN[,13]))

rRMSE.TRAIN2

#test

V2 <- radial\_k(TRAIN[,1:12],TEST[,1:12],gamma2)

Y.hat.TEST2 <- A2%\*%V2

Y.hat.TEST2

MSE.TEST2 <- mean((Y.hat.TEST2-TEST[,13])^2)

RMSE.TEST2 <- sqrt(MSE.TEST2)

RMSE.TEST2

rRMSE.TEST2 <- RMSE.TEST2/mean(abs(TEST[,13]))

rRMSE.TEST2

#set gamma=0.1

gamma3=0.1

G3 <- radial\_k(TRAIN[,1:12],TRAIN[,1:12],gamma3)

M3=G3+(lam)\*diag(960)

M3\_i=solve(M3)

y=TRAIN[,13]

A3=y%\*%M3\_i

#train

Y.hat.train3 <- A3%\*%G3

MSE.TRAIN3 <- mean((Y.hat.train3-TRAIN[,13])^2)

RMSE.TRAIN3 <- sqrt(MSE.TRAIN3)

RMSE.TRAIN3

rRMSE.TRAIN3 <- RMSE.TRAIN3/mean(abs(TRAIN[,13]))

rRMSE.TRAIN3

#test

V3 <- radial\_k(TRAIN[,1:12],TEST[,1:12],gamma3)

Y.hat.TEST3 <- A3%\*%V3

MSE.TEST3 <- mean((Y.hat.TEST3-TEST[,13])^2)

RMSE.TEST3 <- sqrt(MSE.TEST3)

RMSE.TEST3

rRMSE.TEST3 <- RMSE.TEST3/mean(abs(TEST[,13]))

rRMSE.TEST3

#set gamma=0.05

gamma4=0.05

G4 <- radial\_k(TRAIN[,1:12],TRAIN[,1:12],gamma4)

M4=G4+(lam)\*diag(960)

M4\_i=solve(M4)

y=TRAIN[,13]

A4=y%\*%M4\_i

#train

Y.hat.train4 <- A4%\*%G4

MSE.TRAIN4 <- mean((Y.hat.train4-TRAIN[,13])^2)

RMSE.TRAIN4 <- sqrt(MSE.TRAIN4)

RMSE.TRAIN4

rRMSE.TRAIN4 <- RMSE.TRAIN4/mean(abs(TRAIN[,13]))

rRMSE.TRAIN4

#test

V4 <- radial\_k(TRAIN[,1:12],TEST[,1:12],gamma4)

Y.hat.TEST4 <- A4%\*%V4

MSE.TEST4 <- mean((Y.hat.TEST4-TEST[,13])^2)

RMSE.TEST4 <- sqrt(MSE.TEST4)

RMSE.TEST4

rRMSE.TEST4 <- RMSE.TEST4/mean(abs(TEST[,13]))

rRMSE.TEST4

#set gamma=0.02

gamma5=0.02

G5 <- radial\_k(TRAIN[,1:12],TRAIN[,1:12],gamma5)

M5=G5+(lam)\*diag(960)

M5\_i=solve(M5)

y=TRAIN[,13]

A5=y%\*%M5\_i

#train

Y.hat.train5 <- A5%\*%G5

MSE.TRAIN5 <- mean((Y.hat.train5-TRAIN[,13])^2)

RMSE.TRAIN5 <- sqrt(MSE.TRAIN5)

RMSE.TRAIN5

rRMSE.TRAIN5 <- RMSE.TRAIN5/mean(abs(TRAIN[,13]))

rRMSE.TRAIN5

#test

V5 <- radial\_k(TRAIN[,1:12],TEST[,1:12],gamma5)

Y.hat.TEST5 <- A5%\*%V5

MSE.TEST5 <- mean((Y.hat.TEST5-TEST[,13])^2)

RMSE.TEST5 <- sqrt(MSE.TEST5)

RMSE.TEST5

rRMSE.TEST5 <- RMSE.TEST5/mean(abs(TEST[,13]))

rRMSE.TEST5

#######################################################################

###set lam

lam1=0.01

M11=G5+(lam1)\*diag(960)

M11\_i=solve(M11)

y=TRAIN[,13]

A11=y%\*%M11\_i

#train

Y.hat.train11 <- A11%\*%G5

MSE.TRAIN11 <- mean((Y.hat.train11-TRAIN[,13])^2)

RMSE.TRAIN11 <- sqrt(MSE.TRAIN11)

RMSE.TRAIN11

rRMSE.TRAIN11 <- RMSE.TRAIN11/mean(abs(TRAIN[,13]))

rRMSE.TRAIN11

#test

Y.hat.TEST11 <- A11%\*%V5

MSE.TEST11 <- mean((Y.hat.TEST11-TEST[,13])^2)

RMSE.TEST11 <- sqrt(MSE.TEST11)

RMSE.TEST11

rRMSE.TEST11 <- RMSE.TEST11/mean(abs(TEST[,13]))

rRMSE.TEST11

###################################################################3

###set lam2

lam2=5

M12=G5+(lam2)\*diag(960)

M12\_i=solve(M12)

y=TRAIN[,13]

A12=y%\*%M12\_i

#train

Y.hat.train12 <- A12%\*%G5

MSE.TRAIN12 <- mean((Y.hat.train12-TRAIN[,13])^2)

RMSE.TRAIN12 <- sqrt(MSE.TRAIN12)

RMSE.TRAIN12

rRMSE.TRAIN12 <- RMSE.TRAIN12/mean(abs(TRAIN[,13]))

rRMSE.TRAIN12

#test

Y.hat.TEST12 <- A12%\*%V5

MSE.TEST12 <- mean((Y.hat.TEST12-TEST[,13])^2)

RMSE.TEST12 <- sqrt(MSE.TEST12)

RMSE.TEST12

rRMSE.TEST12 <- RMSE.TEST12/mean(abs(TEST[,13]))

rRMSE.TEST12

#set lam3

lam3=0.0001

M13=G5+(lam3)\*diag(960)

M13\_i=solve(M13)

y=TRAIN[,13]

A13=y%\*%M13\_i

#train

Y.hat.train13 <- A13%\*%G5

MSE.TRAIN13 <- mean((Y.hat.train13-TRAIN[,13])^2)

RMSE.TRAIN13 <- sqrt(MSE.TRAIN13)

RMSE.TRAIN13

rRMSE.TRAIN13 <- RMSE.TRAIN13/mean(abs(TRAIN[,13]))

rRMSE.TRAIN13

#test

Y.hat.TEST13 <- A13%\*%V5

MSE.TEST13 <- mean((Y.hat.TEST13-TEST[,13])^2)

RMSE.TEST13 <- sqrt(MSE.TEST13)

RMSE.TEST13

rRMSE.TEST13 <- RMSE.TEST13/mean(abs(TEST[,13]))

rRMSE.TEST13

#set lam4

lam4=0.001

M14=G5+(lam4)\*diag(960)

M14\_i=solve(M14)

y=TRAIN[,13]

A14=y%\*%M14\_i

#train

Y.hat.train14 <- A14%\*%G5

MSE.TRAIN14 <- mean((Y.hat.train14-TRAIN[,13])^2)

RMSE.TRAIN14 <- sqrt(MSE.TRAIN14)

RMSE.TRAIN14

rRMSE.TRAIN14 <- RMSE.TRAIN14/mean(abs(TRAIN[,13]))

rRMSE.TRAIN14

#test

Y.hat.TEST14 <- A14%\*%V5

MSE.TEST14 <- mean((Y.hat.TEST14-TEST[,13])^2)

RMSE.TEST14 <- sqrt(MSE.TEST14)

RMSE.TEST14

rRMSE.TEST14 <- RMSE.TEST14/mean(abs(TEST[,13]))

rRMSE.TEST14

#######best gamma5=0.02 lam1=0.01

lam1=0.01

M11=G5+(lam1)\*diag(960)

M11\_i=solve(M11)

y=TRAIN[,13]

A11=y%\*%M11\_i

#train

Y.hat.train11 <- A11%\*%G5

MSE.TRAIN11 <- mean((Y.hat.train11-TRAIN[,13])^2)

RMSE.TRAIN11 <- sqrt(MSE.TRAIN11)

RMSE.TRAIN11

rRMSE.TRAIN11 <- RMSE.TRAIN11/mean(abs(TRAIN[,13]))

rRMSE.TRAIN11

#test

Y.hat.TEST11 <- A11%\*%V5

#10 case

sse=(Y.hat.TEST11-TEST[,13])^2

order(sse,decreasing=TRUE)[1:10]

sse[order(sse,decreasing=TRUE)[1:10]]

#the first 3 principal eigenvectors

cor\_test=cor(TEST[,1:12])

eigen\_TEST <- eigen(cor\_test)

eigenvectors\_TEST <- eigen\_TEST$vectors

pca\_loading\_set=eigen\_TEST$vectors[,1:3]

pca\_scores<-as.matrix(TEST[,1:12])%\*%as.matrix(pca\_loading\_set)

scor1 <- pca\_scores[,1]

scor2 <- pca\_scores[,2]

scor3 <- pca\_scores[,3]

########

pca = data.frame(pca\_scores)

pca\_scores

pca$col <- 2

pca$col[214]<-1

pca$col[208]<-1

pca$col[18]<-1

pca$col[169]<-1

pca$col[22]<-1

pca$col[171]<-1

pca$col[37]<-1

pca$col[101]<-1

pca$col[96]<-1

pca$col[34]<-1

#pca <- as.matrix(pca)

pc1<-pca[,1]

pc2<-pca[,2]

pc3<-pca[,3]

colors <- c("#FF0000", "#56B4E9")

colors <- colors[as.numeric(pca$col)]

install.packages("scatterplot3d")

library("scatterplot3d")

scatterplot3d(pc1, pc2, pc3, pch=16, grid=TRUE, box=TRUE,

color=colors, xlab="scor1", zlab="scor2", ylab="scor3")

###Q4

#reorder the |A1|, |A2|, ....|Am| in decreasing order , which gives a list B1 > B2 ... > Bm >0

A111=abs(A11)

B=A111[order(A111,decreasing=TRUE)]

B

#Compute the ratios bj = (B1 + ... + Bj)/(B1 + ...+Bm)and plot the increasing curve bj versus j

b=0

for (j in 1:960){

b[j]<-sum(B[1:j])/sum(B)

}

b

for (j in 1:960){

if (b[j]>0.99){

new\_j=j

break

}

}

new\_j

#j=837

#plot the decreasing curve Bj versus j

plot(B[1:837], type="o", col="springgreen3", ann="FALSE")

title(main="Decreasing curve Bj versus j", xlab="j", ylab="Bj")

#plot the increasing curve bj versus j

plot(b[1:837], type="o", col="slateblue3", ann="FALSE")

title(main="Increasing curve bj versus j", xlab="j", ylab="bj")

#threshold value THR = Bj

THR = B[837]

#For i =1... m, if |Ai|> THR set AAi = Ai and otherwise set AAi = 0. This yields a reduced formula

AA=0

for (i in 1:960){

if(abs(A11[i])>THR){

AA[i]=A11[i]

}else{

AA[i]=0

}

print(AA[i])

}

AA=t(as.matrix(AA))

#PRED(x) = AA1 K(x, X(1)) + ... + AAm K(x,X(m))

pred\_train=AA%\*%G5

MSE.TRAIN\_AA <- mean((pred\_train-TRAIN[,13])^2)

RMSE.TRAIN\_AA <- sqrt(MSE.TRAIN\_AA)

RMSE.TRAIN\_AA

rRMSE.TRAIN\_AA <- RMSE.TRAIN\_AA/mean(abs(TRAIN[,13]))

rRMSE.TRAIN\_AA

#test

pred\_test<-AA%\*%V5

MSE.TEST\_AA <- mean((pred\_test-TEST[,13])^2)

RMSE.TEST\_AA <- sqrt(MSE.TEST\_AA)

RMSE.TEST\_AA

rRMSE.TEST\_AA <- RMSE.TEST\_AA/mean(abs(TEST[,13]))

rRMSE.TEST\_AA

#5

#5

install.packages('CVST')

library(CVST)

krr.data <- constructData(as.matrix(TRAIN[,1:12]), TRAIN[,13])

p <- list(kernel="rbfdot", sigma=0.02, lambda=0.01/getN(krr.data))

krr <- constructKRRLearner()

m <- krr$learn(krr.data, p)

pred <- krr$predict(m, krr.data)

krr\_mse=mean((pred - krr.data$y)^2)

krr\_rmse=sqrt(krr\_mse)

krr\_rmse

krr\_rrmse=krr\_rmse/mean(abs(TRAIN[,13]))

krr\_rrmse

##test

krr.data\_test <- constructData(as.matrix(TEST[,1:12]), TEST[,13])

pred\_test <- krr$predict(m, krr.data\_test)

krr\_mse\_t=mean((pred\_test - krr.data\_test$y)^2)

krr\_rmse\_t=sqrt(krr\_mse\_t)

krr\_rmse\_t

krr\_rrmse\_t=krr\_rmse\_t/mean(abs(TEST[,13]))

krr\_rrmse\_t