Lab 2 Block 2 Report

Bruno Barakat, Amish Satish, Siqiang Su

16 décembre 2017

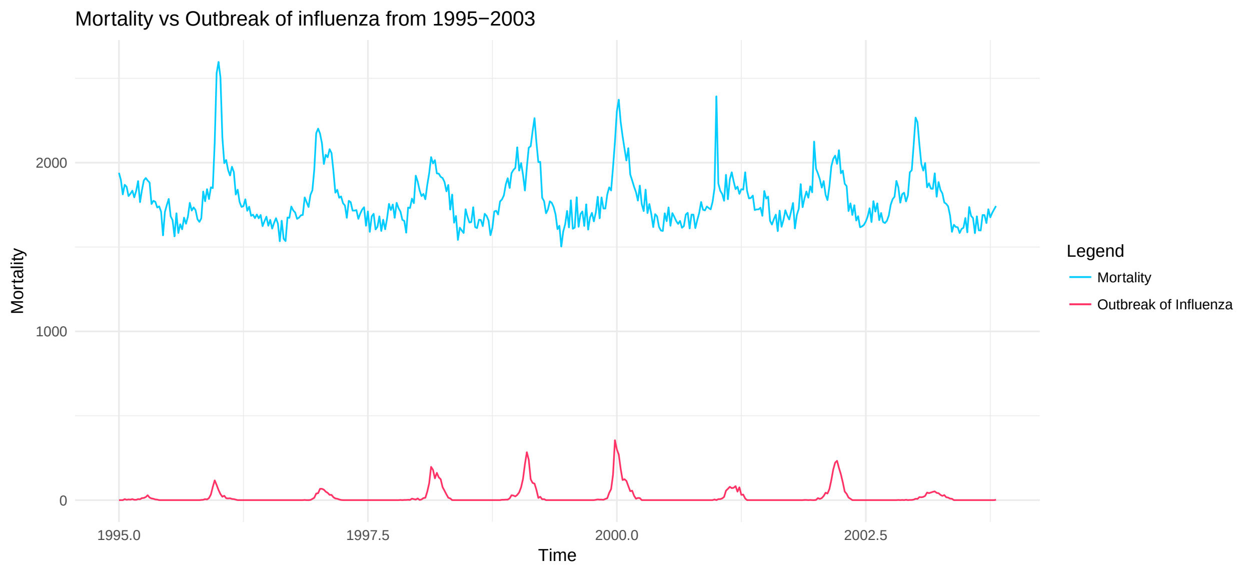
Assignment 1

1.

We can use time series plots to visually inspect how the mortality and influenza number vary with respect to time from the year 1995 to 2003.

Appendix 1.1

```r
influ<-read_excel("Influenza.xlsx")
df_plo1<-data.frame(Mortality=influ$Mortality,Influenza=influ$Influenza,Time=influ$Time)

p1<-ggplot(data=df_plo1,aes(x=Time))+
  geom_line(aes(y=Mortality,colour="Mortality"))+
  geom_line(aes(y=Influenza,colour="Outbreak of Influenza"))+
  scale_colour_manual("Legend",
                      breaks = c("Mortality", "Outbreak of Influenza"),
                      values = c("#00CCFF","#FF3366"))+
  ggtitle("Mortality vs Outbreak of influenza from 1995-2003")


p1+theme_minimal()
```



Priliminary analysis dictates that a rise in death by influenza shows a significant rise in mortality rates.

2.

Using the **gam()** function from the *mgcv* package we can fit the model in which *Mortality* is normally distributed and modelled as a linear function of *Year* and spline function of *Week*. The model parameters are set by generalizied cross-validation. The underlying probabilistic model is shown below.

Appendix 1.2

```
m<-gam(influ$Mortality~influ$Year+s(influ$Week,k=52),data = influ,
       family = "gaussian")
summary(m)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## influ$Mortality ~ influ$Year + s(influ$Week, k = 52)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -680.598   3367.760  -0.202    0.840
## influ$Year     1.233      1.685   0.732    0.465
##
## Approximate significance of smooth terms:
##                  edf Ref.df     F p-value
## s(influ$Week) 14.32  17.87 53.86  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 52/53
## R-sq.(adj) =  0.677   Deviance explained = 68.8%
## GCV = 8708.6  Scale est. = 8398.9    n = 459
```

As we can see above the parameters set can account for about 68.8 percent of the deviance.

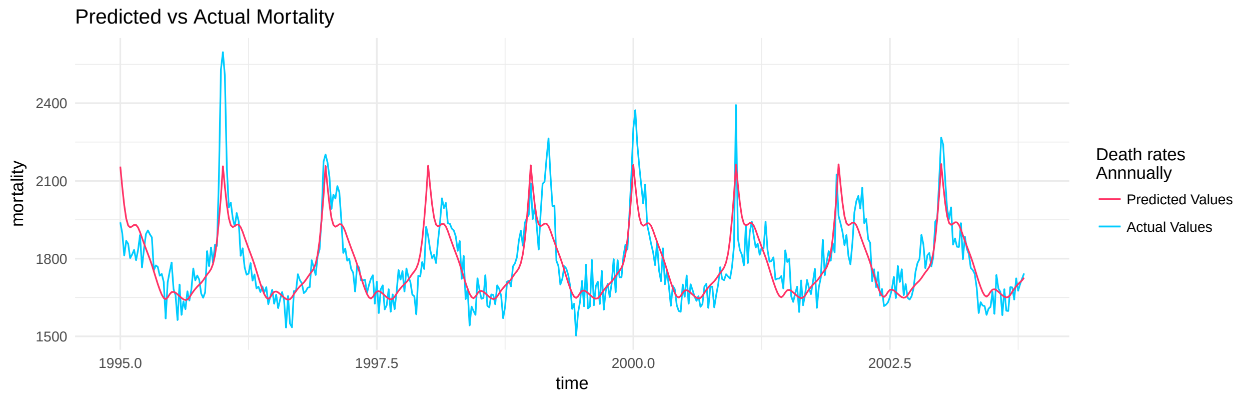
3.

The link function is $g(\mu) = \alpha + s_1(X_1) + s_2(X_2) = \mu$, the probabilistic model is :

$$Y|X \sim N(\alpha + s_1(X_1) + s_2(X_2), \sigma^2)$$

Appendix 1.3

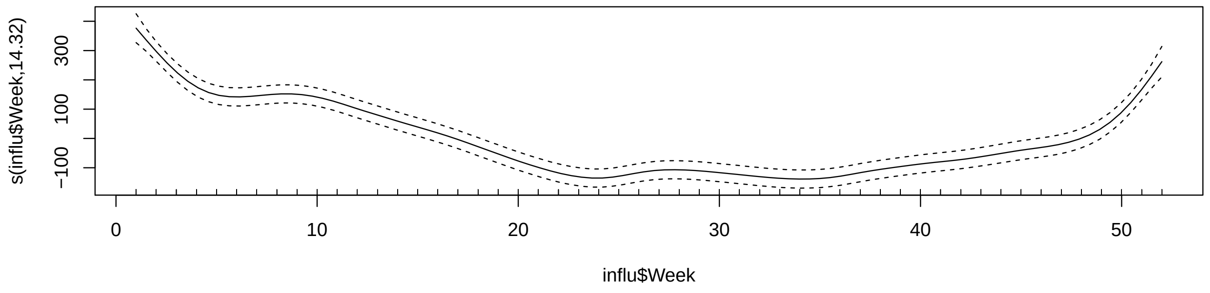
```
pred1<-predict(m)
df_plo2<-data.frame(pred1=as.vector(pred1),mortality=influ$Mortality,time=influ$Time)

p2<-ggplot(data=df_plo2,aes(x=time))+
  geom_line(aes(y=mortality,colour="Actual Values"))+
  geom_line(aes(y=pred1,colour="Predicted Values"))+
  scale_colour_manual("Death rates\nAnnnually",
                      breaks = c("Predicted Values",
                                 "Actual Values"),
                      values = c("#00CCFF","#FF3366"))+
  ggtitle("Predicted vs Actual Mortality")


p2+theme_minimal()
```



As seen from the above graph the predicted values dont exactly match the actual values. This indicates we have to make use of another parameter in the modelling function to get a better deviance explained percentage for the given data.

The spline plot represents the change in the mortality over the course of 52 weeks

```
plot(m)
```

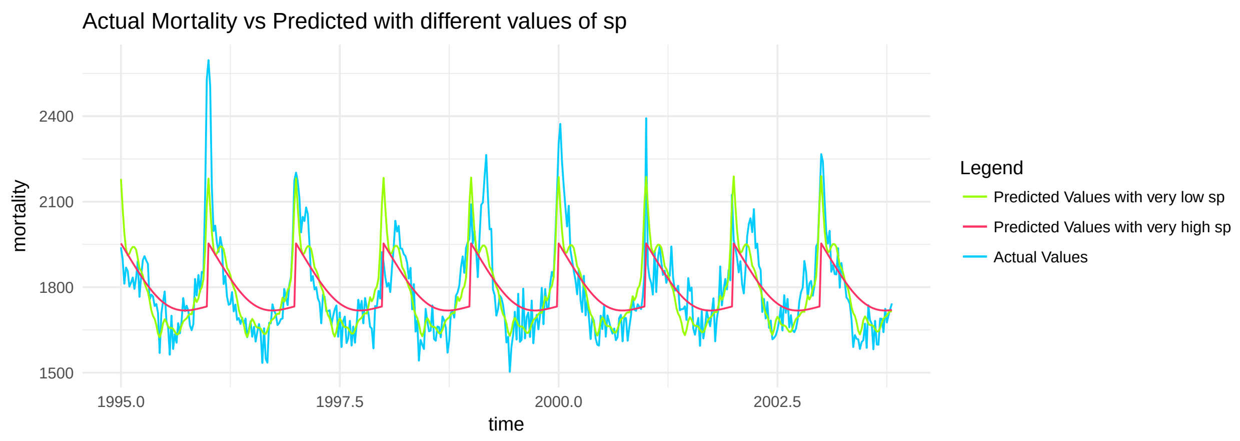


4.

Now we add the spline penalty parameter and observe how it helps change the modelling of our data. Theorectically the smaller the spline penalty the higher the degrees of freedom which leads to better prediction/representation of data, higher the value of spline penalty the lower the degrees of freesdom which leads to poor prediction/representation of the data. We can verify the same with the following plots.

Appendix 1.4

```
m1<-gam(influ$Mortality~influ$Year+s(influ$Week,k=52,sp=0.000000113),data = influ,
        family = "gaussian")


pred2<-predict(m1)

m2<-gam(influ$Mortality~influ$Year+s(influ$Week,k=52,sp=1.5),data = influ,
        family = "gaussian")


pred3<-predict(m2)

#4th


df_plo3<-data.frame(pred2=as.vector(pred2),pred3=as.vector(pred3),
                    mortality=influ$Mortality,time=influ$Time)

p3<-ggplot(data=df_plo3,aes(x=time))+
  geom_line(aes(y=mortality,colour="Actual Values"))+
  geom_line(aes(y=pred2,colour="Predicted Values with very low sp"))+
  geom_line(aes(y=pred3,colour="Predicted Values with very high sp"))+
  scale_colour_manual("Legend",
                      breaks = c("Predicted Values with very low sp",
                                 "Predicted Values with very high sp","Actual Values"),
                      values = c("#00CCFF","#FF3366","#99FF00"))+
  ggtitle("Actual Mortality vs Predicted with different values of sp")


p3+theme_minimal()
```



As we can see the plot shows better representation of the actual data when the sp vlaue is low , for very low value we get better representation of the data as the degrees of freedom also increase.

The deviance explained by the model with the very low sp is given below

```r
summary(m1)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## influ$Mortality ~ influ$Year + s(influ$Week, k = 52, sp = 1.13e-07)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -706.566   3386.737  -0.209    0.835
## influ$Year     1.246      1.694   0.735    0.463
##
## Approximate significance of smooth terms:
##                 edf Ref.df     F p-value
## s(influ$Week) 27.85     28 34.71  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 30/53
## R-sq.(adj) =  0.673   Deviance explained = 69.4%
## GCV = 9082.3  Scale est. = 8491.8    n = 459
```

As you can see the deviance explained is slightly better with 69.4 percent explained when compared to the previous 68.8 percent of the previous model.

If we observe the deviance explained of the model with the very high value of sp we see that the deviance explained is quiet low at 39.2 percent of the total data explained which is much lower than the previous model with the lower sp.

```r
summary(m2)
```

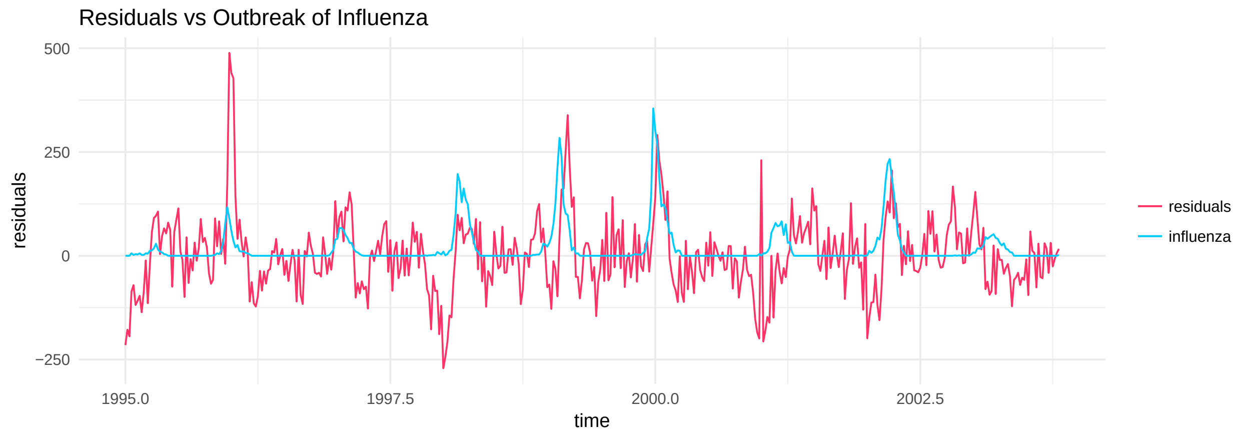
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## influ$Mortality ~ influ$Year + s(influ$Week, k = 52, sp = 1.5)
##
## Parametric coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1822.32422 4624.95623   0.394    0.694
## influ$Year    -0.01929    2.31372  -0.008    0.993
##
## Approximate significance of smooth terms:
##                 edf Ref.df     F p-value
## s(influ$Week) 1.366  1.645 161.2  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.389   Deviance explained = 39.2%
## GCV =  15993  Scale est. = 15876     n = 459
```

5.

We further examine the gam object created in the second question to understand if the residuals is correlated to the outbreaks in influenza.

Appendix 1.5

```
df_plo4<-data.frame(residuals=as.vector(m$residuals),influenza=influ$Influenza,
                    time=influ$Time)

p4<-ggplot(data = df_plo4,aes(x=time))+
  geom_line(aes(y=residuals,colour="residuals"))+
  geom_line(aes(y=influenza,colour="influenza"))+
  scale_color_manual("",breaks=c("residuals","influenza"),
                     values=c("#00CCFF","#FF3366"))+
  ggtitle("Residuals vs Outbreak of Influenza")

p4+theme_minimal()
```



The correlation between is positive which implies that the residuals changes in accordance with the outbreak pattern of influenza. The actual correlation value between the residuals and influenza can be seen below and as seen from the graph it is positive.

Appendix 1.6

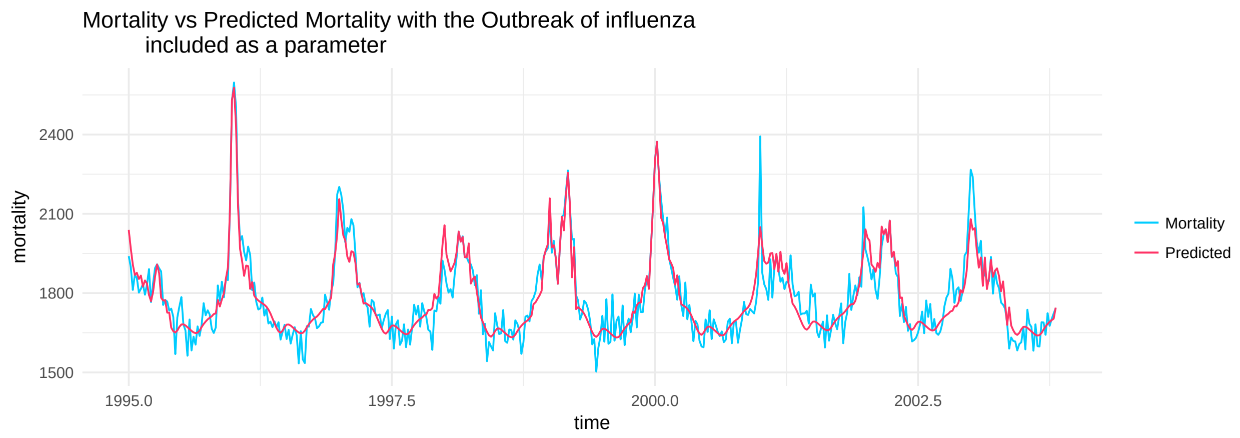
```
cor(df_plo4$residuals,influ$Influenza)

## [1] 0.3397395
```

6.

Now we observe the change in the models ability to represent and predict the actual data if we include the outbreak of influenza as a paramater of spline in the **gam** function along with the year and weeks also splined in the equation.

Appendix 1.7

```
m3<-gam(influ$Mortality~s(influ$Year,k=9)+s(influ$Week,k=52)+s(influ$Influenza,k=85),
        data = influ,family = "gaussian")

pred4<-predict(m3)

df_plot5<-data.frame(mortality=influ$Mortality,time=influ$Time,predicted=as.vector(pred4))

p5<-ggplot(data=df_plot5,aes(x=time))+
  geom_line(aes(y=mortality,color="Mortality"))+
  geom_line(aes(y=predicted,color="Predicted"))+
  scale_color_manual("",
                     breaks=c("Mortality","Predicted"),
                     values=c("#00CCFF","#FF3366"))+
  ggtitle("Mortality vs Predicted Mortality with the Outbreak of influenza
          included as a parameter")

p5+theme_minimal()
```



This is a more accurate representation and predition of the data. The outbreak of influenza was a driving factor in the mortality rate as it is clearly seen in the graph. Going one step further we can see that the deviance explained percentage is much higher than the previous models at 85.4 percent.

```
summary(m3)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## influ$Mortality ~ s(influ$Year, k = 9) + s(influ$Week, k = 52) +
##     s(influ$Influenza, k = 85)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1783.8        3.2   557.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Approximate significance of smooth terms:
##                         edf Ref.df      F p-value
## s(influ$Year)         4.663  5.677  1.487   0.181
## s(influ$Week)        14.641 18.248 18.533  <2e-16 ***
## s(influ$Influenza) 69.740 72.833  5.600  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 134/144
## R-sq.(adj) =  0.819   Deviance explained = 85.4%
## GCV = 5846.7  Scale est. = 4699.8    n = 459
```

We can conclude that the influence of influenza had a definite impact on the mortality rate throughout the years.

Assignment 2

1.

We want to perform the nearest shrunken centroid classification on our training data, using cross validation to select the best threshold. First step We first simply split the data

```
rm(list = ls())
set.seed(12345)
library(pamr)
data <- read.csv2("data.csv")
#splitting data
index<-sample(1:nrow(data),0.7*nrow(data))
train<-data[index,]
test<-data[-index,]
```

Then we separate response ($Conference) from predictors (everything else). We can train our model for a sequence of thresholds, and cross validation will help us to choose the best one for the training data.

```
## creating list for pamr.train
x=t(train[,-4703])
y=train[,4703]
mytraindata<-list(x=x,y=y,genenames=rownames(x),geneid=as.character(1:nrow(x)))

model<-pamr.train(data=mytraindata,threshold = seq(0,4,0.1))
cvmodel<-pamr.cv(model,mytraindata)
```

By looking at the number of error, we can set the threshold to be 1.4.

```
print(cvmodel)
```

```
## Call:
## pamr.cv(fit = model, data = mytraindata)
##    threshold nonzero errors
## 1  0.0       3428    6
## 2  0.1       3409    6
## 3  0.2       3114    7
## 4  0.3       3024    7
## 5  0.4       3000    7
## 6  0.5       1979    6
## 7  0.6        852    6
## 8  0.7        841    6
## 9  0.8        673    6
## 10 0.9        622    6
## 11 1.0        297    6
## 12 1.1        293    6
## 13 1.2        272    6
## 14 1.3        231    5
## 15 1.4        170    5
## 16 1.5        138    6
## 17 1.6        129    7
## 18 1.7         98    7
## 19 1.8         88    7
## 20 1.9         71    7
## 21 2.0         62    7
## 22 2.1         47    7
## 23 2.2          43    7
## 24 2.3          36    7
## 25 2.4          30    7
## 26 2.5          20    7
## 27 2.6          20    7
## 28 2.7          14    7
## 29 2.8          12    7
## 30 2.9          12    9
## 31 3.0          12    9
## 32 3.1          11    9
## 33 3.2           9   10
## 34 3.3           9   11
## 35 3.4           6   11
## 36 3.5           6   13
## 37 3.6           6   14
## 38 3.7           6   16
## 39 3.8           4   16
## 40 3.9           2   20
## 41 4.0           1   20
```

In most cases we've seen that several thresholds can give the same number of errors for the training set (especially for the lowest errors). We decided to choose the higher one : 1.4 instead of 1.3 because the higher the threshold is the less variables we keep. We have noticed also that the choice of the threshold doesn't affect the most contributing features.

With a threshold of 1.4, we're selecting 170 features out of the 4703.

Here are the 10 most contributing features:

```r
#names of first variables
variables_id<-as.numeric(pamr.listgenes(fit = model,data=mytraindata,threshold = 1.4)[,1])
```

```r
cat(colnames(data)[variables_id][1:5],"\n",colnames(data)[variables_id][6:10])
```

```
## papers important submission due published
##  position call conference dates candidates
```

The words papers, important, sunmission, due, can let us think of assignment that has nothing to do with a conference so their presence could indicate other mails. The words conference and dates are making sense to classify emails as conference.

We want now to classify the test data and produce a confusion matrix :

```r
# predict on test
predictions<-pamr.predict(fit=model,newx = t(test[,-4703]),threshold = 1.4)
table(predicted=predictions,real=test[,4703])
```

```
##          real
## predicted  0  1
##         0 10  2
##         1  0  8
```

Only two mistakes were made, that's an error rate of 0.1. The error rate with the training data was 5/40=0.125. This is apparenty better but the number of samples is too low to make conclusions, except that the test and train error rates are in the same range.

2.

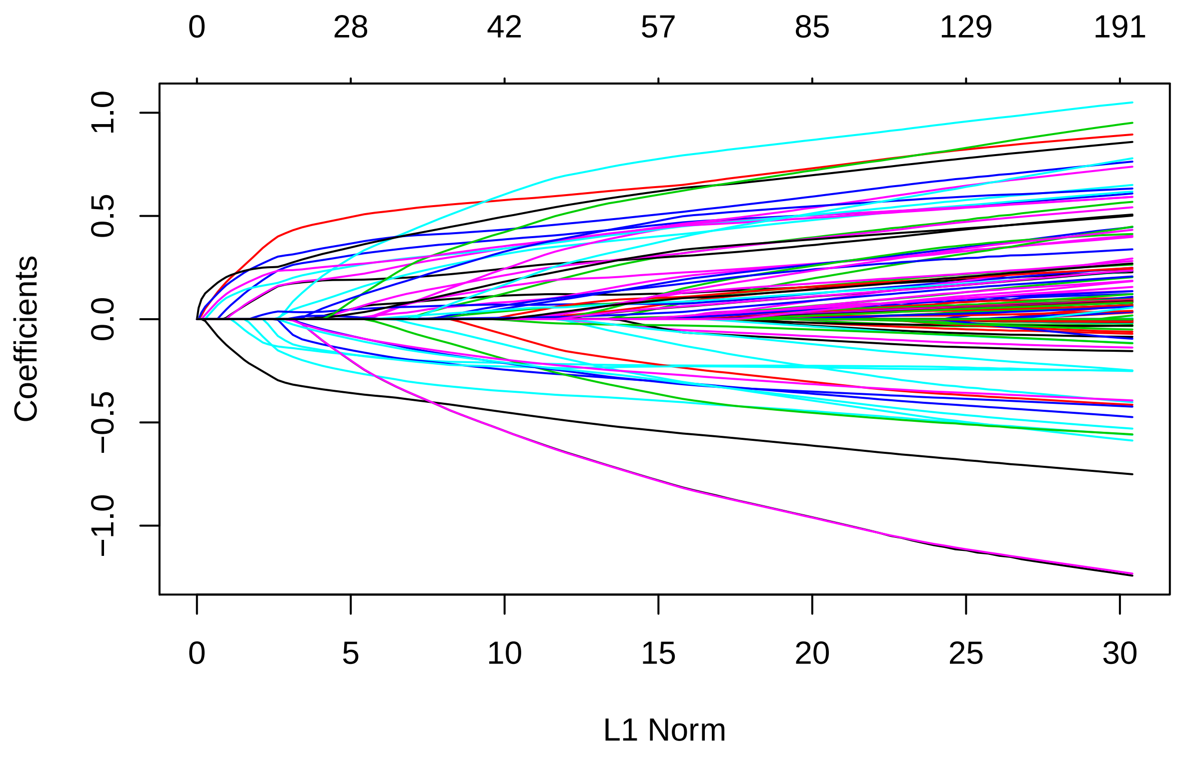
Elastic Net

Divide the data into 0.7 train and 0.3 test

```
set.seed(12345)
train_rows <- sample(1:64, .70*64)
x=data.frame(data[,-4703])
y=data.frame(data[,4703])
x.train <- x[train_rows, ]
x.test <- x[-train_rows, ]

y.train <- y[train_rows,]
y.test <- y[-train_rows,]
```

Fit elastic net model

```
fit.elnet <- glmnet(as.matrix(x.train),as.factor(y.train) , family="binomial", alpha=0.5)
plot(fit.elnet)
```

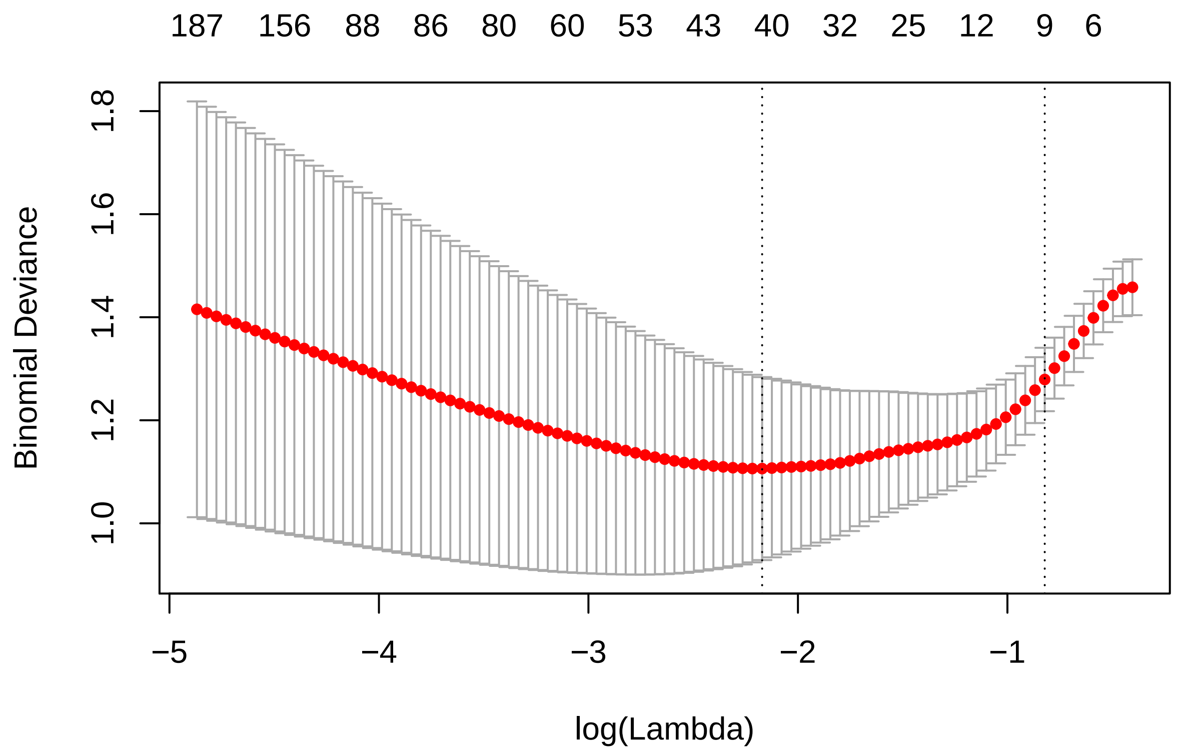


use cross validation to find penalty

```
cv.elnet <- cv.glmnet(as.matrix(x.train),as.factor(y.train) , family="binomial", alpha=0.5)
```

Find optimal panelty

```
plot(cv.elnet)
```



```
lambda= cv.elnet$lambda.min
```

Count contributions:

By refering to the CV plot generated, we find that under optimal lambda, there are around 40 contribution variables selected, which reduce the dimensionality significantly.

computing error from prediction on the test set:

```
fit=glmnet(as.matrix(x.train),as.matrix(y.train) , family="binomial", alpha=0.5,lambda = lambda)
pd1=predict.glmnet(fit,newx = as.matrix(x.test))
err1= sum((pd1-y.test)^2)
```

Support Vector Machine

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'

## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
svp <- ksvm(as.matrix(x.train),as.matrix(y.train),type="C-svc",kernel='vanilladot',scaled=c())
```

```
##  Setting default kernel parameters
```

```
svp
```

```
## Support Vector Machine object of class "ksvm"
```

```
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 43
##
## Objective Function Value : -2.0817
## Training error : 0.022727
```

By refering to the summary of the model, there are 41 support vectors. Nearly all data points are support vectors, this is the indication of high sparsity in high dimensional case. Even if the data are sparse, SVM can still find the maximum margin and not suffer from the p>>n problem without any regularization. Overfitting will not be a problem because of the insensitivity of misclassification loss.'

Prediction and compute test error

```
pd2=predict(svp,x.test)
table(y.test,pd2)
```

```
##       pd2
## y.test  0  1
##      0 10  0
##      1  1  9
```

```
err2=sum((pd2-y.test)^2)
```

Making Comparative Table

```
TestErrorRate=c(0.1,"-",0.05)
ContributionFeatures=c(170,40,4702)
TestError=c("-",43.999,"-")
comp=data.frame(row.names = c("Nearest Shrunken Centroid","Elastic Net","SVM"),ContributionFeatures
                ,TestErrorRate,TestError)
comp
```

```
##                           ContributionFeatures TestErrorRate TestError
## Nearest Shrunken Centroid                  170           0.1         -
## Elastic Net                                 40             -    43.999
## SVM                                       4702          0.05         -
```

By inspecting the comparative table, nearest shrunken centroid is preferred due to its smaller dimensions than SVM, and low misclassification rate.

###.3.

Benjamini-Hochberg method

```
p=vector(length = 4702)
for(i in 1:4702){

  pvalue=t.test(as.matrix(x[,i])~as.matrix(y),data=data,alternative="two.sided")
  p[i]=pvalue$p.value
}
```

According to the Benjamini-Hochberg method, we need to sort the p-values in ascending order:

```
sortp=sort(p,decreasing=F)
```

Then we define L by choosing alpha=0.15,and compare the ordered p-values and its associated L:

```
alpha=0.15
M=4702
L=vector(length = 4702)
for(j in 1:4702){
  for(i in 1:4702){
    if(sortp[i]>=alpha*j/M){
      L[j]=i-1
      break

  }

  }
}
```

Then we compare each pair of pj and pL:

```
PL=vector(length = 4702)
for(k in 1:4702){
  PL[k]=sortp[L[k]]
}
dp=sortp-PL
```

Counting which features are rejected under this method (return negative entries index)

```
reject=vector(length = 4702)

  for(j in 1:4702){
    if(dp[j]<=0){
      reject[j]=j

    }
}
cat(reject[1:25],"\n",reject[26:50],"\n",reject[51:75])
```

```
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
##  51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 0 0 0 0 0 0 0 0
```

By inspecting the outcome, there are totally 67 features being rejected under null hypothesis. This indicate only first 67 features are significant, others will not make much contribution to our prediction. Hence our high dimensional data now can be reduce to 67 features.