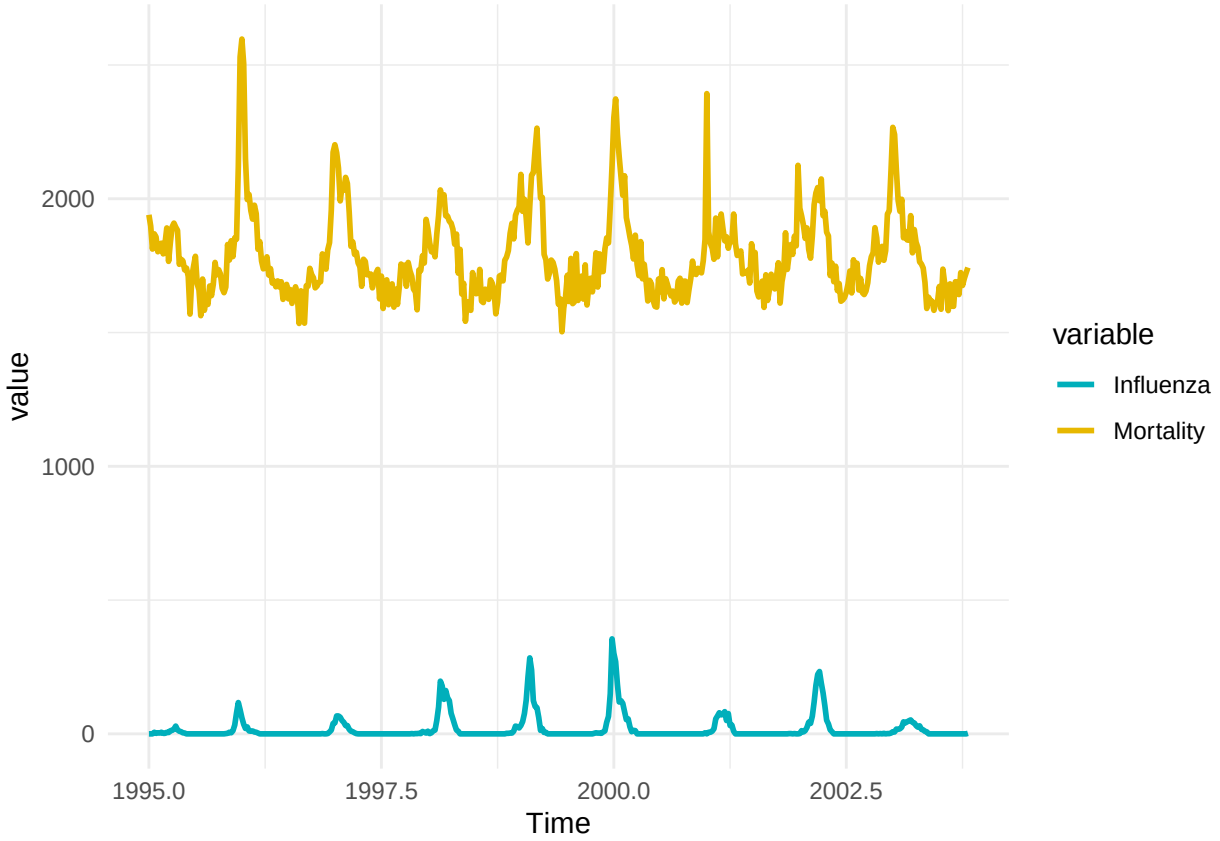
Block 2 Lab 2

Aman Kumar Nayak

12/8/2019

1.1



Post analysing both time series graphs we can say that apart from one instance, whenever there is rise in Influenza, rise in moratality can also be seen so there seem to be dependent realtion on moratality of influenza. If you also consider that spikes in Influenza is also seen mostly in winter part of years i.e year end or erly part of year so rise of inflenza is also related to time of year.

1.2

Regression Equation : f(x) = yi = beta0 + f1(x1) + f2(x2) + ... + fn(xn) +e

here f1 ... fn are different non linear function on variable Week

and f1(x1)is restricted to beta1*x1

which gives

f(x) = beta0 + beta1*x1 + splineFunction + error

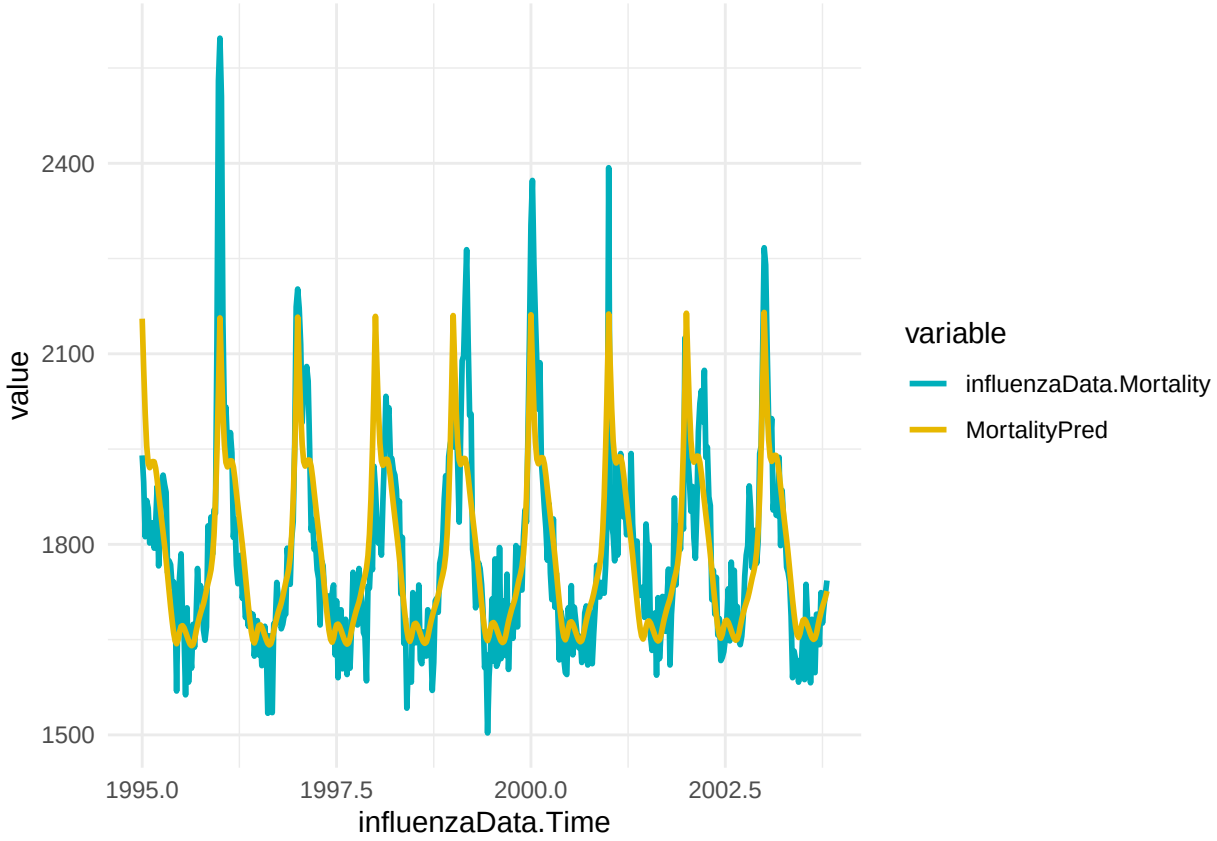
$$y = Mortality$$

$$y = N(\mu, \sigma^2)$$

Thus underlying Probabilistic model is

$$\hat{y} = \beta_0 + \beta_1 Year_i + f(Week_i) + \epsilon_i$$

1.3

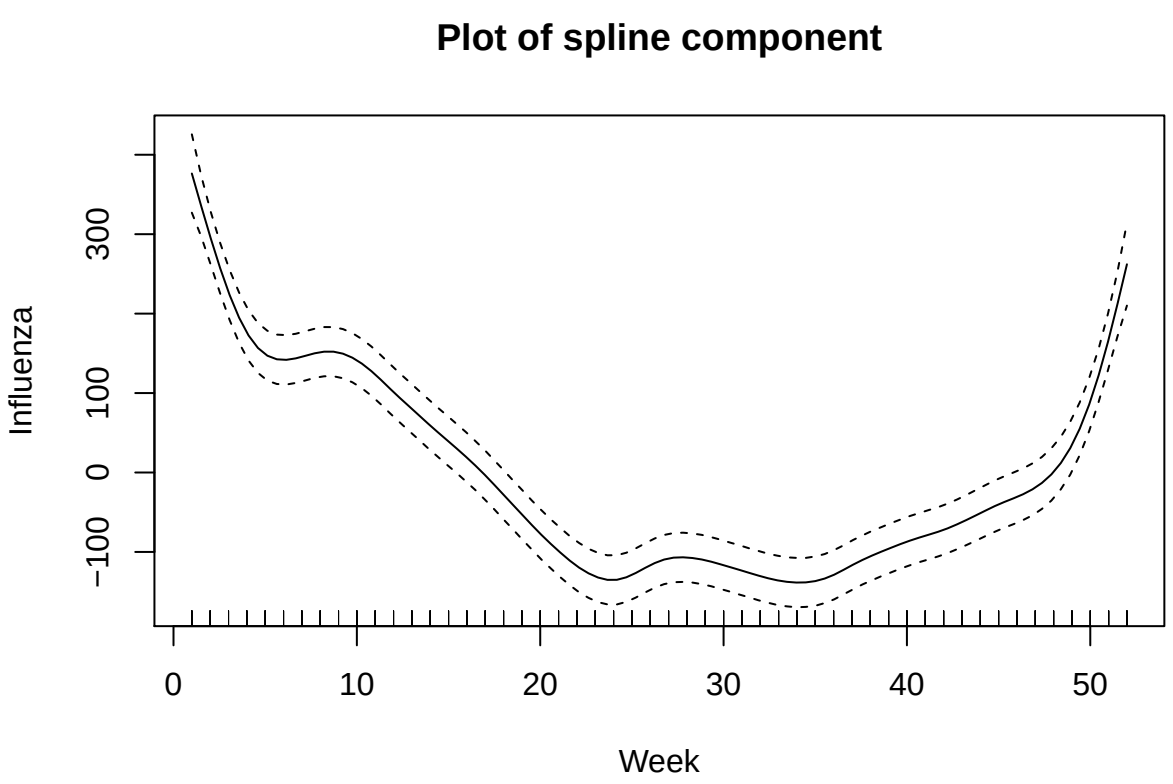


All peaks are correctly predecited while plot only missed to correctly predict possible max value in case of year end of 1995 and early 1996.

In terms of numbers we could see that predicted high do not cross 2200 mark while actual is mostly high and same goes for low values as predicted low is around 1650 actual value is always seen lower than predicted one.

In terms of trend in change of Mortality, it can be seen that high mortality rate is mostly seen in between end of year and early part year which is mostly spread of winter session of year.

So during Winter it can be seen that there is heavy rise in mortality rate.

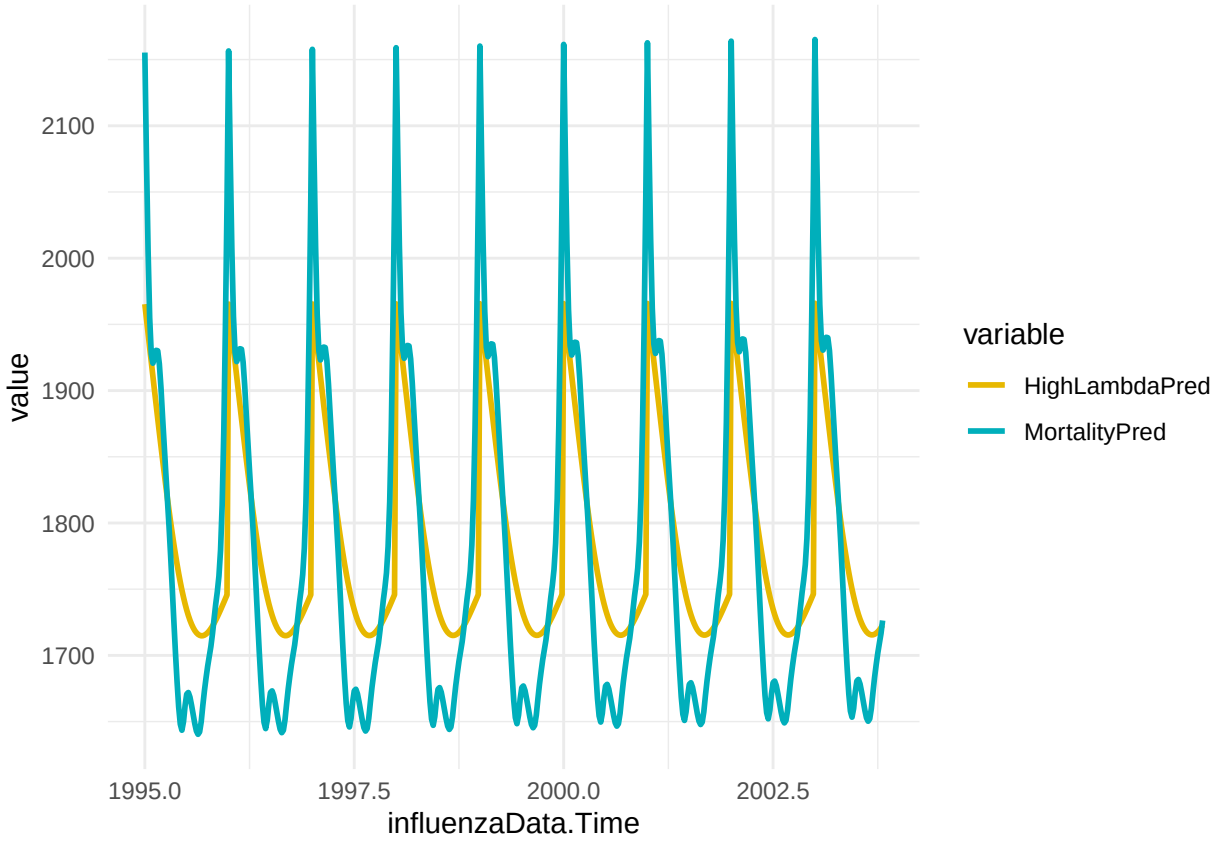


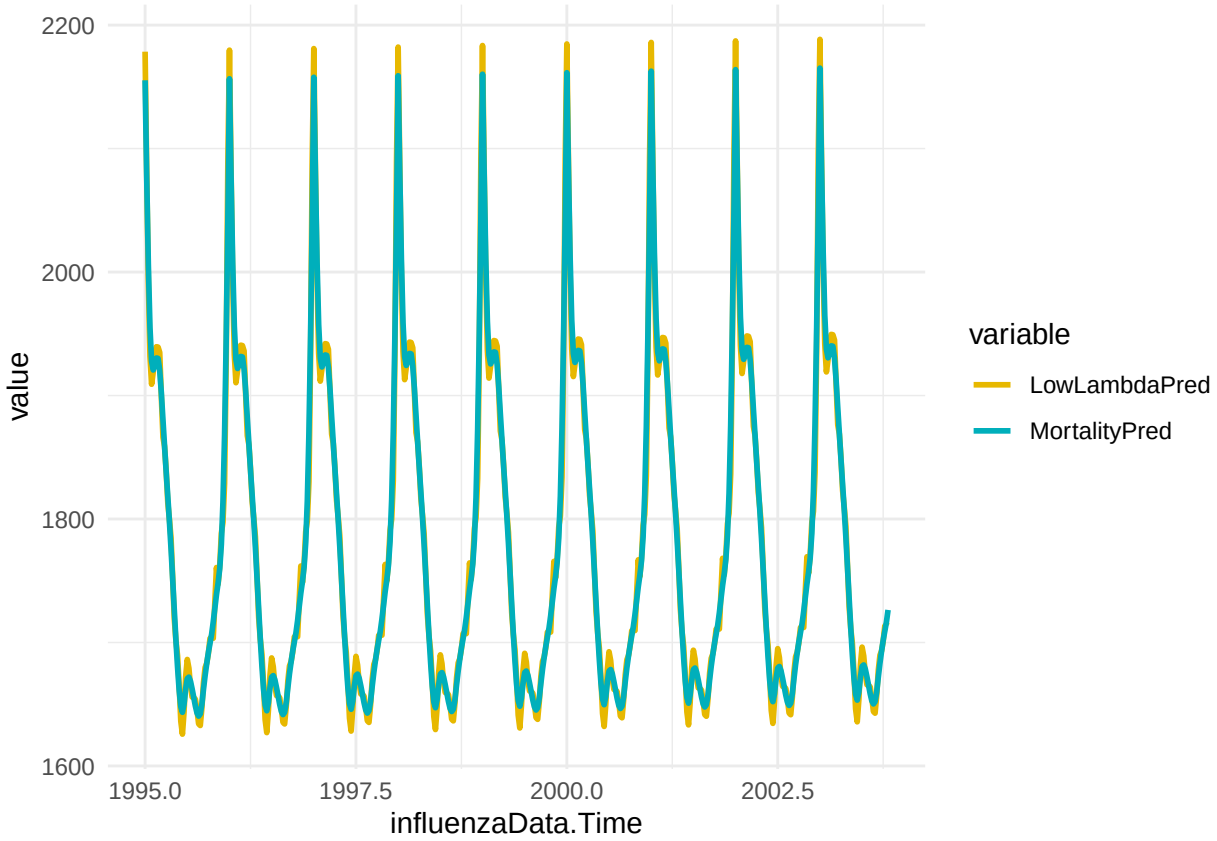
Looking it above graph it can be concluded that cases of influenza is high during winter session of the year which can be seen with high value of Influenza during week 1-7 and again raising value from week 45 onwards.

Summary of GAM Model is as below

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ Year + s(Week, k = length(unique(influenzaData$Week)))
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -680.598   3367.760  -0.202    0.840
## Year           1.233      1.685   0.732    0.465
##
## Approximate significance of smooth terms:
##           edf Ref.df     F p-value
## s(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
```

4th Part


2100
2000
1900
1800
1700
value
1995.0
1997.5
2000.0
2002.5
influenzaData.Time
variable
HighLambdaPred
MortalityPred

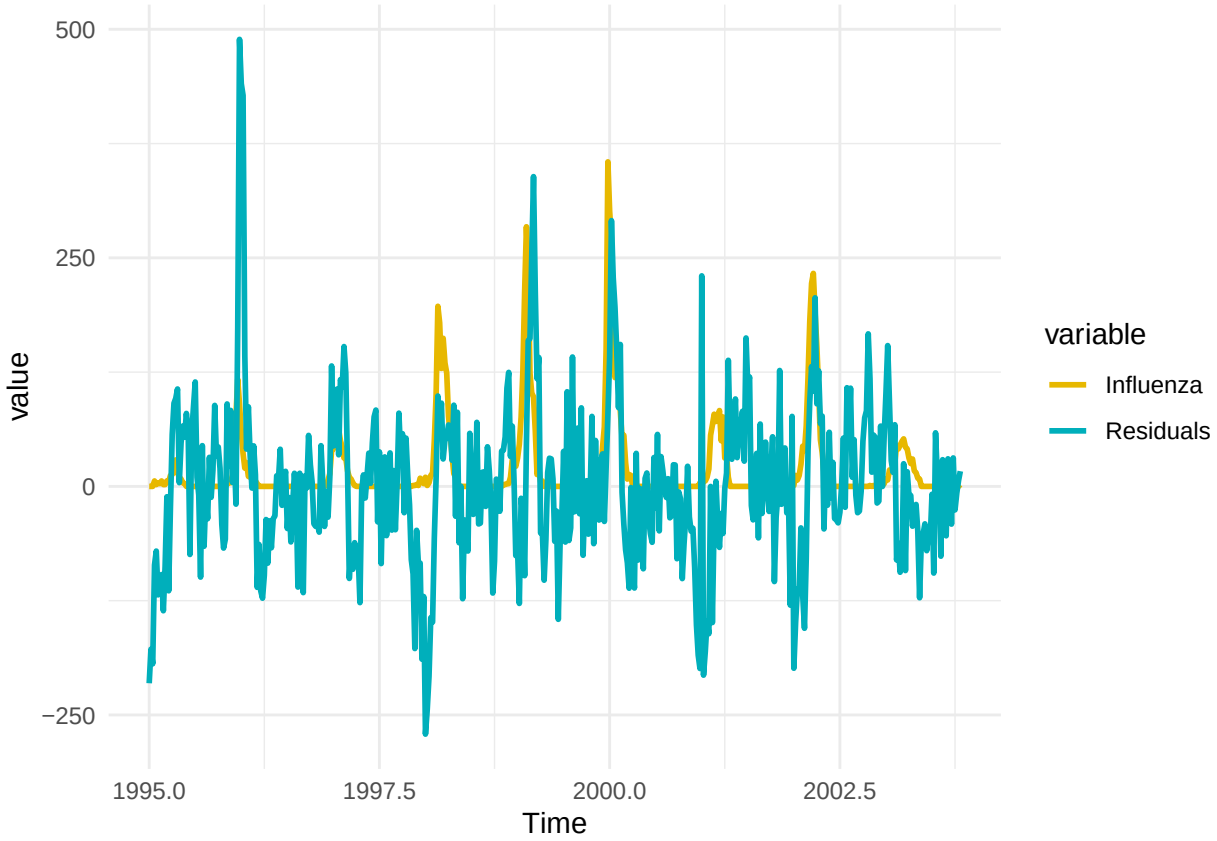


Degree of freedom is inversly proportional to penality factor(lambda).

Comapring plots of high and low lambda values of lambda, it can be analysed that smoothnes of curve is inverlsly propertion to value of degree of freedom(directly proportional to penality factor or lamda).

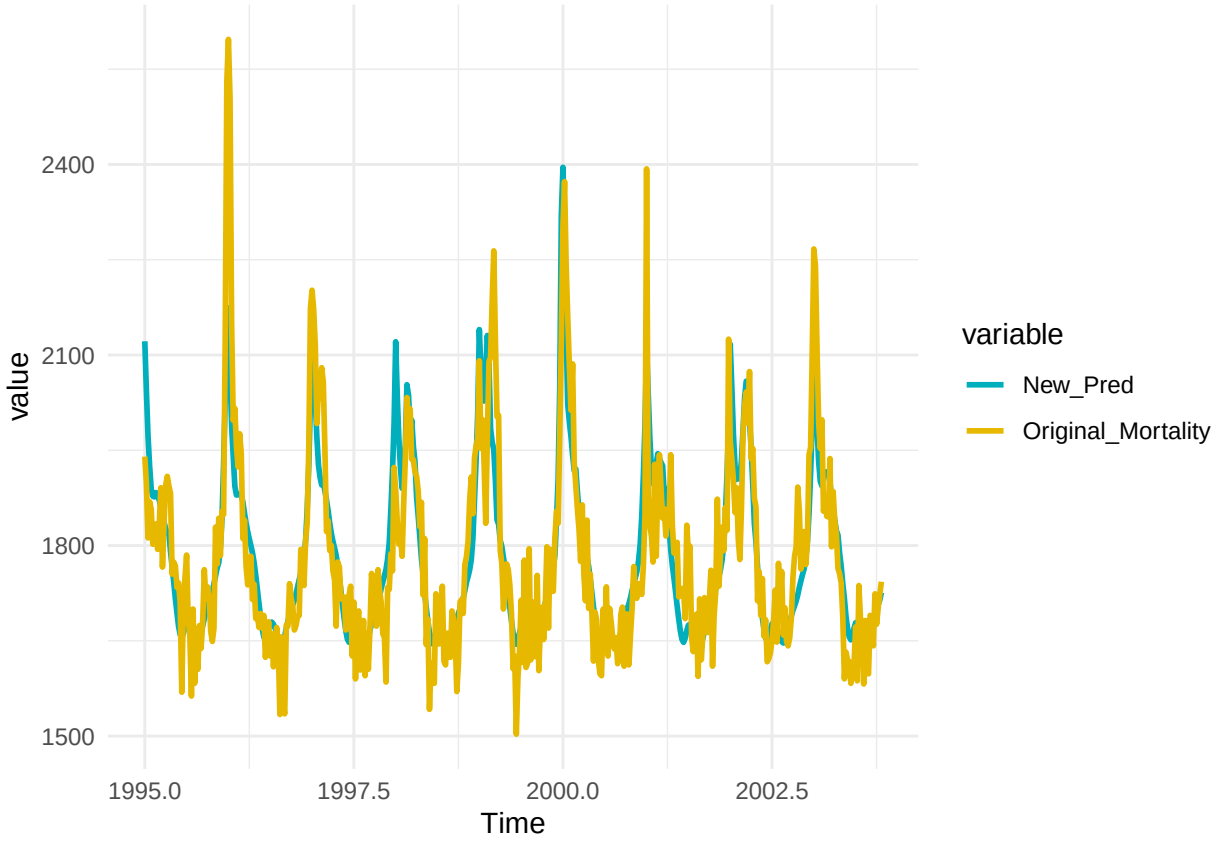
So when lamda is extremly low it can be seen that graph is extremly waggling and thus prone to over fitting. But for extremly high value of penality is also not good as it make graph way more smoother and it will decrease edge points prediction capability of model.

5th Part



Mostly higher spikes in residual can be seen with higher value of Influenza but Influenza is insufficient in explaining all spikes of Residuals.

6th



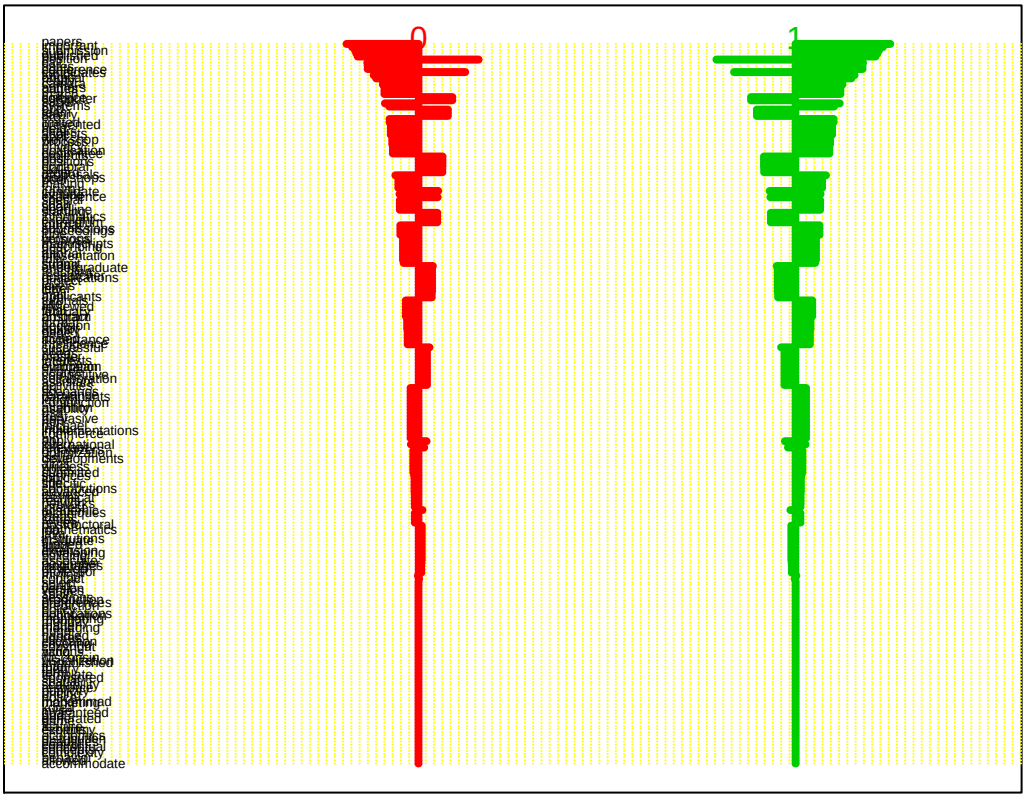
When compared to prediction in absence of impact of influenza, max high was 2200 while here we could see that max peak of 2400 is predicted to while considering impact of influenza, we could see that prediction capability increases.

Part 2

1

```
## 12345678910111213141516171819202122232425262728293031323334353637383940 41
```

```
## 12Fold 1 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 2 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 3 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 4 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 5 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 6 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 7 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 8 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 9 :12345678910111213141516171819202122232425262728293031323334353637383940 41
## Fold 10 :12345678910111213141516171819202122232425262728293031323334353637383940 41
```



We could out of 4703 variable, 231 festures are selected from it.

Top 10 Features are:

```
##         top10
## 1      papers
## 2   important
## 3  submission
## 4         due
## 5   published
## 6    position
## 7        call
## 8  conference
## 9       dates
## 10 candidates
```

Test Error

```
## [1] 0.1
```

2.2

Performing CV in order to find best lamda so that we can obtain lowest prediction Error

```
## [1] 0.1
```

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

```
## [1] 0.05
```

##**Comparing Models**

```
##   MCR.Shrunken.Centroid MCR.Elastic MCR.SVM
## 1                   0.1         0.1    0.05
```

On comparing Error Rate or Misclassification Rate, I could see that Error Rate for SVM is lowest thus SVM will be used for classification.

2.3

```
## [1] "Total number of rejected features"
```

```
## [1] 39
```

Total 39 features are rejected here as part of rejected hypothesis.

Below features are rejected based on there Benjamini L Values which if less than 0 is marked as rejected hypothesis.

```
##            colName         pVal featureProbability rejected
## 3036        papers 1.116910e-10      -1.063366e-05      yes
## 4060    submission 7.949969e-10      -2.126675e-05      yes
## 3187      position 8.219362e-09      -3.189310e-05      yes
## 3364     published 1.835157e-07      -4.235158e-05      yes
## 2049     important 3.040833e-07      -5.286478e-05      yes
## 596           call 3.983540e-07      -6.340428e-05      yes
## 869     conference 5.091970e-07      -7.392721e-05      yes
## 607     candidates 8.612259e-07      -8.420896e-05      yes
## 1045         dates 1.398619e-06      -9.430534e-05      yes
## 3035         paper 1.398619e-06      -1.049391e-04      yes
## 4282        topics 5.068373e-06      -1.119031e-04      yes
## 2463       limited 7.907976e-06      -1.196973e-04      yes
## 606      candidate 1.190607e-05      -1.263330e-04      yes
## 599         camera 2.099119e-05      -1.278816e-04      yes
## 3433         ready 2.099119e-05      -1.385154e-04      yes
## 389        authors 2.154461e-05      -1.485958e-04      yes
## 3125           phd 3.382671e-05      -1.469474e-04      yes
## 3312      projects 3.499123e-05      -1.564167e-04      yes
## 2974           org 3.742010e-05      -1.646216e-04      yes
## 681         chairs 5.860175e-05      -1.540737e-04      yes
## 1262           due 6.488781e-05      -1.584214e-04      yes
## 2990      original 6.488781e-05      -1.690552e-04      yes
## 2889  notification 6.882210e-05      -1.757547e-04      yes
## 3671        salary 7.971981e-05      -1.754907e-04      yes
## 3458        record 9.090038e-05      -1.749439e-04      yes
## 3891        skills 9.090038e-05      -1.855777e-04      yes
## 1891          held 1.529174e-04      -1.341945e-04      yes
## 4177          team 1.757570e-04      -1.219886e-04      yes
## 3022         pages 2.007353e-04      -1.076441e-04      yes
```

```
## 4628      workshop 2.007353e-04      -1.182779e-04      yes
## 810      committee 2.117020e-04      -1.179450e-04      yes
## 3285   proceedings 2.117020e-04      -1.285788e-04      yes
## 272          apply 2.166414e-04      -1.342731e-04      yes
## 4039        strong 2.246309e-04      -1.369174e-04      yes
## 2175 international 2.295684e-04      -1.426137e-04      yes
## 1088        degree 3.762328e-04      -6.582996e-06      yes
## 1477     excellent 3.762328e-04      -1.721677e-05      yes
## 3191          post 3.762328e-04      -2.785054e-05      yes
## 3243     presented 3.765147e-04      -3.820241e-05      yes
```

```r
#knitr::opts_chunk$set(echo = TRUE , fig.width = 16 , fig.height = 6)
knitr::opts_chunk$set(echo = TRUE)
library(readxl)
influenzaData = data.frame(read_xlsx(path = "G:/MS Machine Learning/Term/Term2/ML/ML Assignment/2b/Influ

library(tidyr)
library(dplyr)
library(ggplot2)
df = influenzaData %>% select(Time , Influenza , Mortality) %>% gather(key = "variable" , value = "valu

ggplot(df, aes(x = Time, y = value)) +
  geom_line(aes(color = variable), size = 1) +
  scale_color_manual(values = c("#00AFBB", "#E7B800")) +
  theme_minimal()

library(mgcv)
gamFit = gam(formula = Mortality~Year+s(Week , k = length(unique(influenzaData$Week))) , data = influenz


# summary.gam(gamFit)
#
# plot.gam(gamFit)



# par(mfrow = c(2,2))
# gam.check(gamFit)


MortalityPred = predict.gam(gamFit)

df1 = data.frame(influenzaData$Time , influenzaData$Mortality , MortalityPred)

df2 = df1 %>% select(influenzaData.Time , influenzaData.Mortality , MortalityPred) %>% gather(key = "va

ggplot(df2, aes(x = influenzaData.Time, y = value)) +
  geom_line(aes(color = variable), size = 1) +
  scale_color_manual(values = c("#00AFBB", "#E7B800")) +
  theme_minimal()


plot.gam(gamFit , ylab = "Influenza",main = "Plot of spline component")
summary.gam(gamFit)
```

```r
LowLambda = smooth.spline(x = influenzaData$Time , y = influenzaData$Mortality , lambda = 2.579*10^(-08)
#LowLambda
LowLambdaPred = predict(LowLambda)

#plot(LowLambda , main = "With low value of penality factor" , ylab = "Mortality" , xlab = "Time")




HighLambda = smooth.spline(x = influenzaData$Time , y = influenzaData$Mortality , lambda = .00109)

#HighLambda
HighLambdaPred = predict(HighLambda)
#plot(HighLambda, main = "With High value of penality factor" , ylab = "Mortality" , xlab = "Time")
LowLambda = gam(formula = Mortality~Year+s(Week , k = length(unique(influenzaData$Week)) , sp = 0.000000

LowLambdaPred = predict.gam(LowLambda)

HighLambda = gam(formula = Mortality~Year+s(Week , k = length(unique(influenzaData$Week)) , sp = 1.13193

HighLambdaPred = predict.gam(HighLambda)


LamdaDf = data.frame(HighLambdaPred , LowLambdaPred , MortalityPred, influenzaData$Time )

dfLamdaA = LamdaDf %>% select(influenzaData.Time , HighLambdaPred , MortalityPred) %>% gather(key = "va

ggplot(dfLamdaA, aes(x = influenzaData.Time, y = value)) +
  geom_line(aes(color = variable) , size = 1) +
  scale_color_manual(values = c( "#E7B800" , "#00AFBB")) +
  theme_minimal()


dfLamdaB = LamdaDf %>% select(influenzaData.Time , LowLambdaPred , MortalityPred) %>% gather(key = "var

ggplot(dfLamdaB, aes(x = influenzaData.Time, y = value)) +
  geom_line(aes(color = variable) , size = 1) +
  scale_color_manual(values = c( "#E7B800" , "#00AFBB")) +
  theme_minimal()

#
# dfLamdaB = LamdaDf %>% select(influenzaData.Time , LowLambda.y , influenzaData.Mortality) %>% gather(
#
# ggplot(dfLamdaB, aes(x = influenzaData.Time, y = value)) +
#   geom_line(aes(color = variable) , size = 1) +
#   scale_color_manual(values = c( "#E7B800" , "#00AFBB")) +
#   theme_minimal()
#


InfDf = data.frame(MortalityPred , influenzaData$Influenza , influenzaData$Time )
#colnames(InfDf) = c("MortalityPred" , "Influenza" , "Time" )
```

```r
#InfDf = as.data.frame(InfDf)

# dfA = InfDf %>% select(influenzaData.Time, MortalityPred , influenzaData.Influenza) %>% gather(key =
#
# ggplot(InfDf, aes(x = influenzaData.Time, y = value)) +
#   geom_line(aes(color = variable) , size = 1) +
#   scale_color_manual(values = c( "#E7B800" , "#00AFBB")) +
#   theme_minimal()



Df5 = data.frame("Time" = influenzaData$Time , "Influenza" = influenzaData$Influenza ,  "Residuals" = ga

# Df5A = Df5 %>% select(influenzaData.Time , influenzaData.Influenza , gamFit.residuals ) %>% gather(ke

Df5A = Df5 %>% select(Time , Influenza , Residuals ) %>% gather(key = "variable" , value = "value" , -Ti


ggplot(Df5A, aes(x = Time, y = value)) +
  geom_line(aes(color = variable) , size = 1) +
  scale_color_manual(values = c( "#E7B800" , "#00AFBB")) +
  theme_minimal()




gam2 = gam(formula = Mortality~s(Year , k = length(unique(influenzaData$Year)))+s(Week , k = length(unic


newYpred = predict.gam(gam2)


df6 = data.frame("Time" = influenzaData$Time , "Original_Mortality" = influenzaData$Mortality , "New_Pre

df6B = df6 %>% select(Time , Original_Mortality , New_Pred) %>% gather(key = "variable" , value = "valu

ggplot(df6B, aes(x = Time, y = value)) +
  geom_line(aes(color = variable), size = 1) +
  scale_color_manual(values = c("#00AFBB", "#E7B800")) +
  theme_minimal()

HDdata = data.frame(read.csv2(file = "G:/MS Machine Learning/Term/Term2/ML/ML Assignment/2b/data.csv"))

newData = data.frame(read.csv2(file = "G:/MS Machine Learning/Term/Term2/ML/ML Assignment/2b/data.csv"))


HDdata$Conference = as.factor(HDdata$Conference)

#divide data in test and train
suppressWarnings(RNGversion("3.5.9"))
set.seed(12345)
```

```
#install.packages("pamr")
library(pamr)
n = length(HDdata[,1])
id = sample(1:n , floor(n*.7))
train = HDdata[id,]
test  = HDdata[-id,]
rownames(train) = 1:nrow(train)
x = t(train[,-4703])
y = train[[4703]]
trainData = list(x=x , y = as.factor(y) , geneid = as.character(1:nrow(x)) , genenames = rownames(x))
model = pamr.train(trainData , threshold = seq(0,4,0.1))
# pamr.plotcen(model , trainData , threshold = 1)
# pamr.plotcen(model , trainData , threshold = 2.5)



#a=pamr.listgenes(model,trainData,threshold=2.5)

#cat( paste( colnames(trainData)[as.numeric(a[,1])], collapse='\n' ) )


cvmodel = pamr.cv(model,trainData)

a=pamr.listgenes(model,trainData,threshold=1.3)
minThreshold = cvmodel$threshold[which.min(cvmodel$error)]


pamr.plotcen(model , trainData , threshold = minThreshold)



# print(cvmodel)
# windows()
# pamr.plotcv(cvmodel)

a = (pamr.listgenes(model,trainData,threshold=minThreshold))


top10 = colnames(train[,as.numeric(a[1:10,1])])


as.data.frame(top10)



rownames(test) = 1:nrow(test)
x1 = t(test[,-4703])
y1 = test[[4703]]
testData = list(x=x1 , y = as.factor(y1) ,geneid = as.character(1:nrow(x1)) , genenames = rownames(x1))


yPred = pamr.predict(model , newx = testData$x , threshold = minThreshold)
con = table(y1 , yPred)
```

```r
MCR  = 1 - (sum(diag(con))/sum(con))
MCR

#Elastic net

library(glmnet)
library(dplyr)
library(tidyr)
library(caret)
nTrain = newData[id,]
nTest = newData[-id,]

x2 = model.matrix(Conference~. , nTrain)[,-1]
y2 = nTrain$Conference

x2Test = model.matrix(Conference~. , nTest)[,-1]
y2Test = nTest$Conference


set.seed(12345)

cvElastic = cv.glmnet(x = x2 , y = y2 , alpha = 0.5)

minLambda = cvElastic$lambda.min

elsaticModel = glmnet(x = x2 , y = y2 , family = "binomial" , alpha = 0.5 , lambda = minLambda)

# ypredER = elsaticModel %>% predict(x2Test) %>% as.vector()
#
# data.frame(
#   RMSE = RMSE(y2Test , ypredER),
#   Rsquare = R2(y2Test , ypredER)
# )

ypredER = predict(elsaticModel , newx = as.matrix(x2Test) , type = "class" , s = "lambda.min")

conElastic = table(y2Test , ypredER)

MCRelastic  = 1 - (sum(diag(conElastic))/sum(conElastic))
MCRelastic
#SVM with "vanilladot" kernel
library(kernlab)
svmModel = ksvm(x = as.matrix(train[,-4703]) , y = train[,4703] , kernel = "vanilladot" , scale = FALSE

yPredSVM = predict(svmModel , newdata = test[,-4703])

conSVM = table(test[,4703] , yPredSVM)

MCRsvm  = 1 - (sum(diag(conSVM))/sum(conSVM))
MCRsvm


comparision = data.frame( "MCR Shrunken Centroid" = MCR , "MCR Elastic" = MCRelastic, "MCR SVM" = MCRsv
```

```
comparision

#HDdata = data.frame(read.csv2("~/Machine Learning/Lab2-Block2/data.csv"))

pValDF <- data.frame(name = character(), pValue = numeric(), stringsAsFactors = FALSE)

#calculating pValues
for(i in  1: (length(HDdata)-1))
{
  pVal <- t.test(HDdata[,i] ~ HDdata[,4703], data = HDdata,alternative = "two.sided")[["p.value"]]

  colName <- colnames(HDdata)[i]
  pValDF <-rbind(pValDF,data.frame(colName,pVal))
}

alpha = 0.05
colNum = ncol(HDdata)-1

ordered_pVal = pValDF[order(pValDF$pVal),]

featureProbability <- c()

#calculating feature probabilites AKA Benjamini L Values
for (i in 1:colNum)
{
  featureProbability[i] = ordered_pVal$pVal[i] - ((alpha * i)/ colNum)
}

ordered_pVal$featureProbability = featureProbability

rejectedFeatures <- which(ordered_pVal$featureProbability < 0)

totalRejectedFeatures <- length(which(featureProbability < 0))

rejectedColumn <- c()

#In order to update dataframe for individual contribution presence
for (i in 1:length(featureProbability))
{
  if( i <= totalRejectedFeatures)
  {
    rejectedColumn[i] = "yes"
  }
  else
  {
    rejectedColumn[i] = "no"
  }
}

ordered_pVal$rejected = rejectedColumn

rejectedFeatureDF = ordered_pVal[1:totalRejectedFeatures,]
```

```r
print("Total number of rejected features")
nrow(rejectedFeatureDF)

rejectedFeatureDF
```