

# Machine Learning - Block 2 Assignment 2

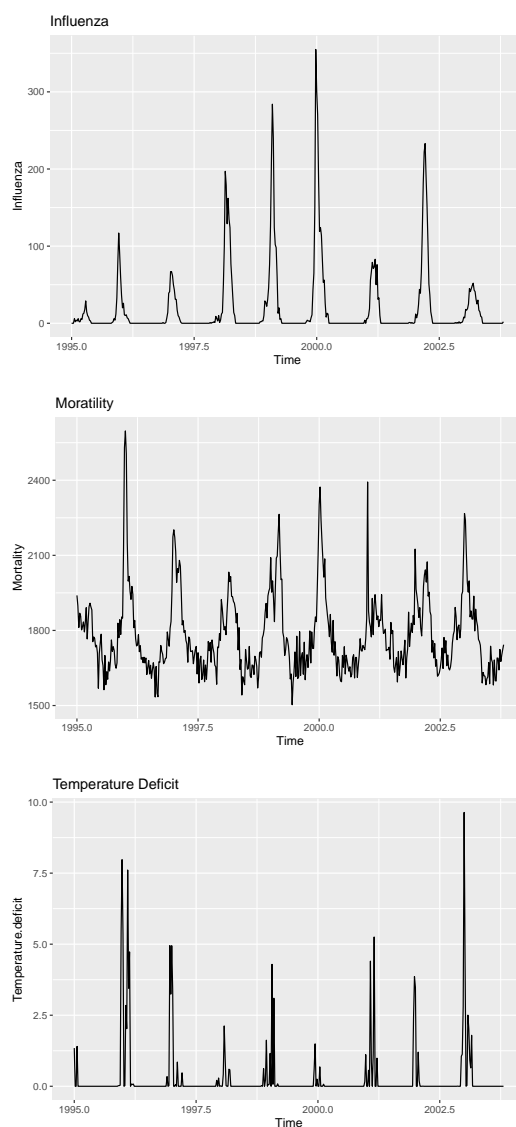
*Agustín Valencia - aguva779*

## Assignment 1. Using GAM and GLM to examine mortality rates

The Excel document `influenza.xlsx` contains weekly data on the mortality and the number of laboratory-confirmed cases of influenza in Sweden. In addition, there is information about population-weighted temperature anomalies (temperature deficits).

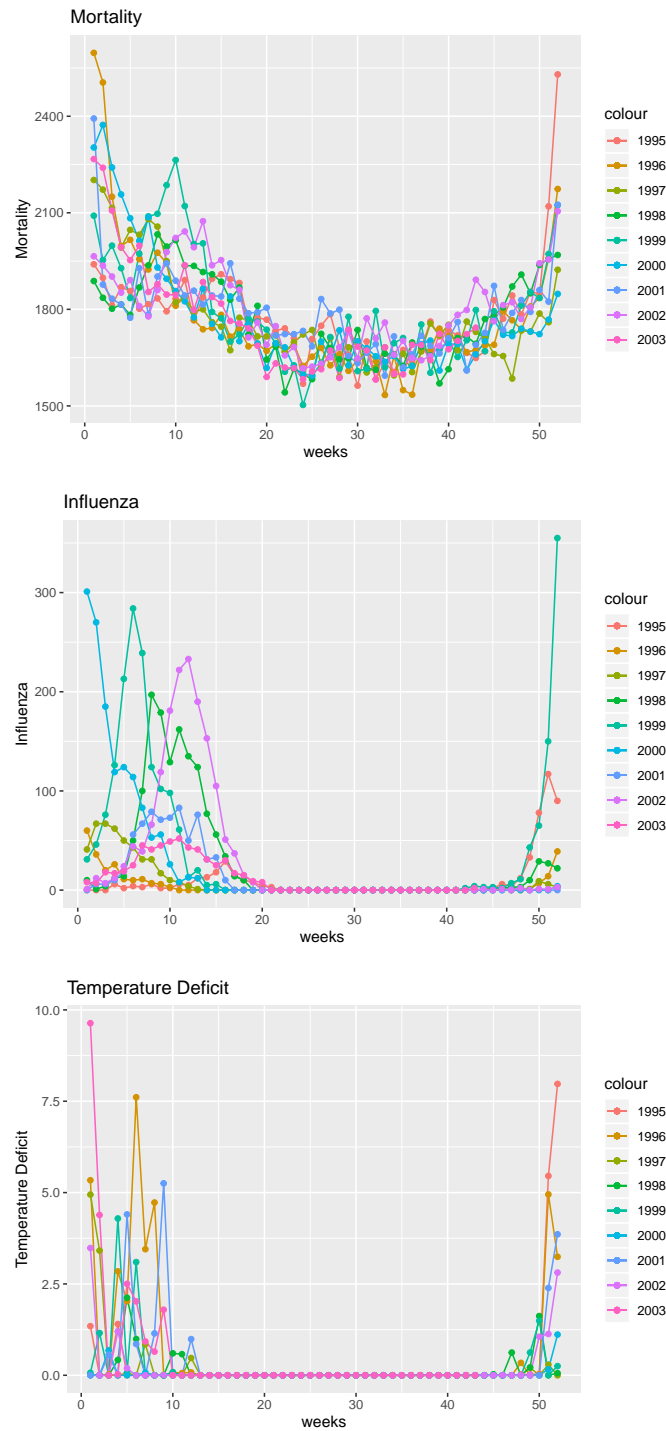
1. Use time series plots to visually inspect how the mortality and influenza number vary with time (use Time as X axis). By using this plot, comment how the amounts of influenza cases are related to mortality rates.

Plotting the data against the time:



It seems to be a modality in the data. It can be seen that Influenza peaks seems to have a correlation with peaks in mortality and temperature deficit

As the model is meant to learn from weeks variations along the year, it is better to separate the data into several time series per year and see how the data looks now.



Now it is quite clear that the data shows that during winter weeks Influenza cases and Mortality related to increases, also the temperature deficit seems to follow that trend.

2. Use `gam()` function from `mgcv` package to fit a GAM model in which Mortality is normally distributed and modelled as a linear function of Year and spline function of Week, and make sure that the model parameters are selected by the generalized cross-validation. Report the underlying probabilistic model.

It is fitted one model with the following formula:

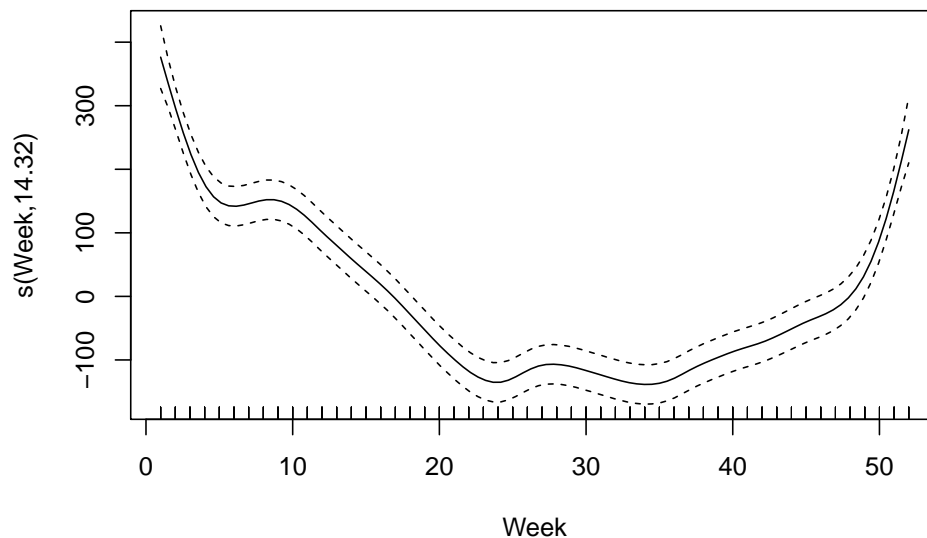
$$\text{Mortality} \approx \text{Year} + \text{spline}(\text{Week})$$

The summary of the obtained probabilistic model is:

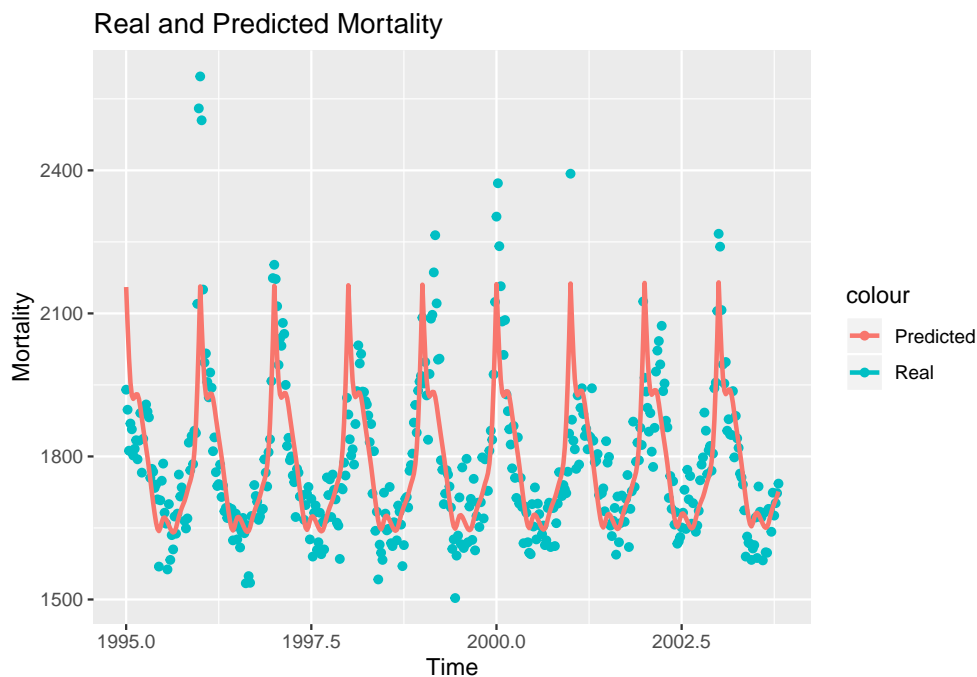
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ Year + s(Week, k = length(unique(data$Week)))
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -680.598   3367.760  -0.202   0.840
## Year          1.233     1.685    0.732   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
```

The smoothed curve approximated by the GAM model is:

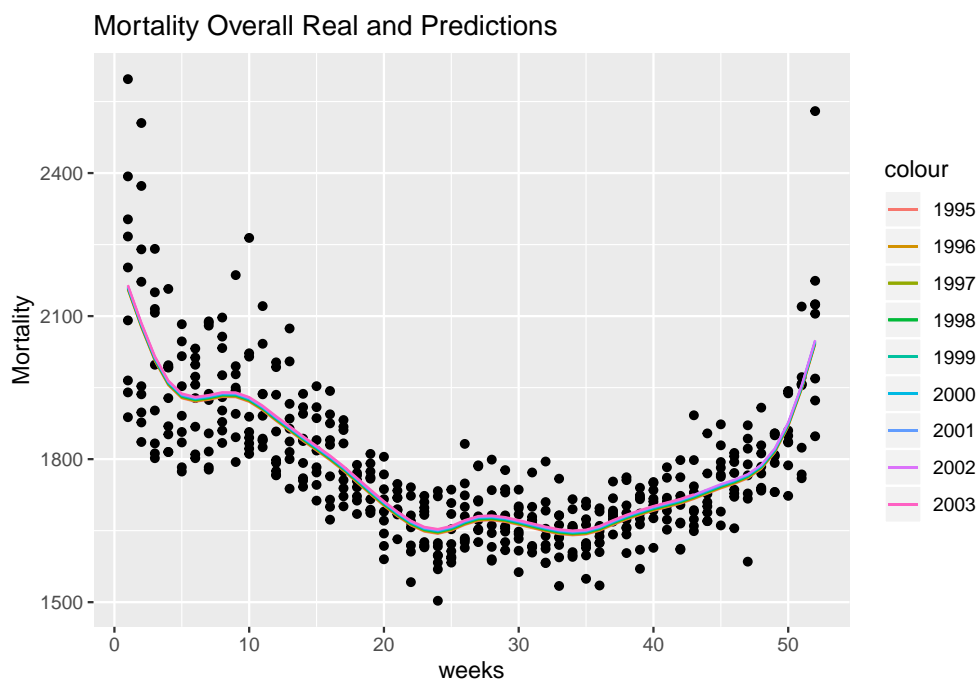
### Smooth Approximation of Mortality



- Plot predicted and observed mortality against time for the fitted model and comment on the quality of the fit. Investigate the output of the GAM model and report which terms appear to be significant in the model. Is there a trend in mortality change from one year to another? Plot the spline component and interpret the plot.



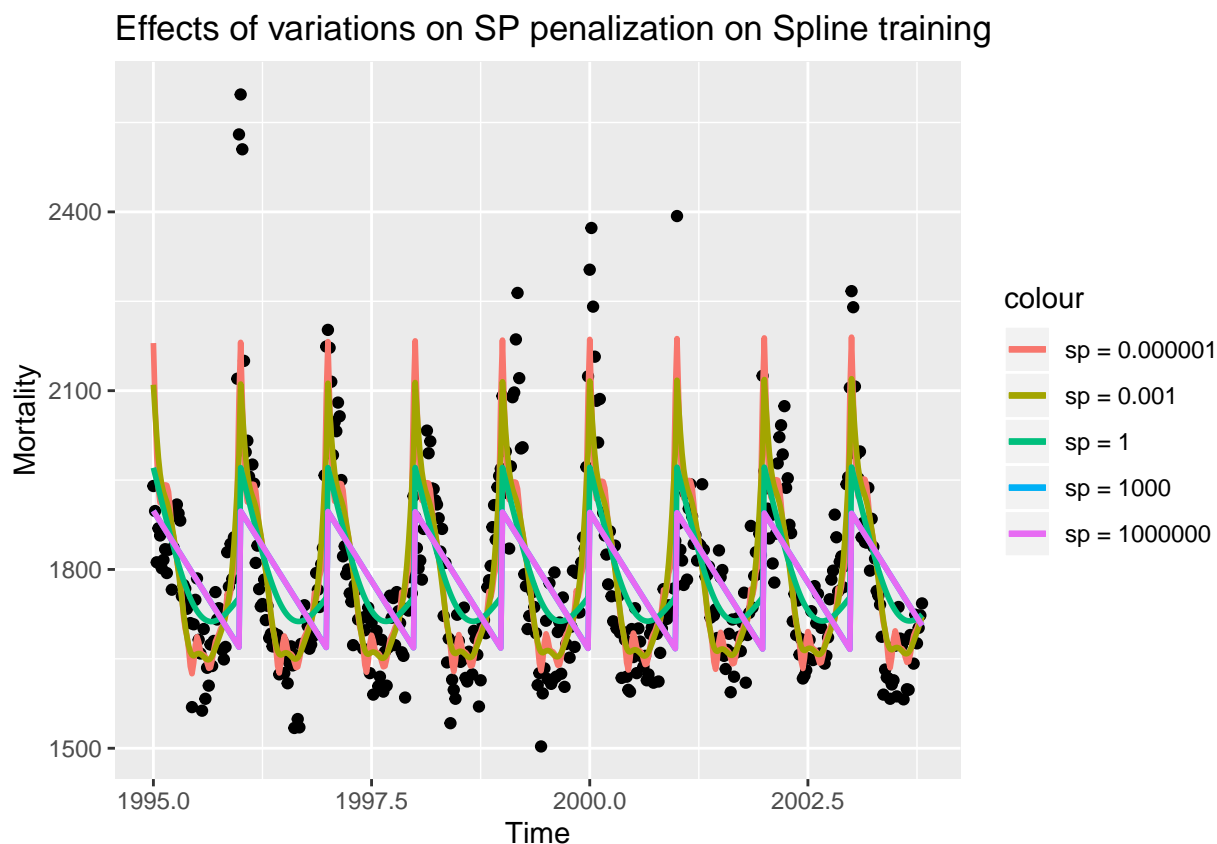
Although it seems to be a fair general approximation it seems to be missing some abnormal positive and negative peaks.



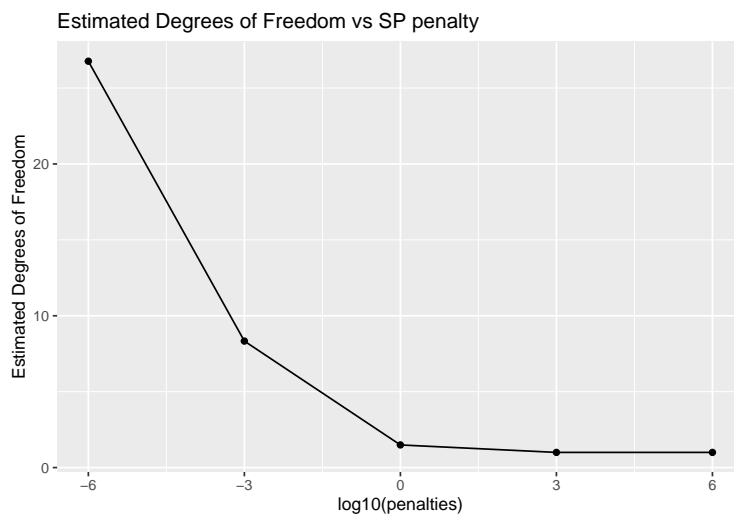
It can be seen that given the variability of mortality per week each year, the GAM fit acceptably well the data, though still too general, thus not as good as we would like to.

- Examine how the penalty factor of the spline function in the GAM model from step 2 influences the estimated deviance of the model. Make plots of the predicted and observed mortality against time for cases of very high and very low penalty factors. What is the relation of the penalty factor to the degrees of freedom? Do your results confirm this relationship?

The smoothing penalty in `gam()` is given by the `sp` parameter sent to the constructor to be used in training. For this testing purpose it have been used the following penalties  $\{10^{-6}, 10^{-3}, 1, 10^3, 10^6\}$



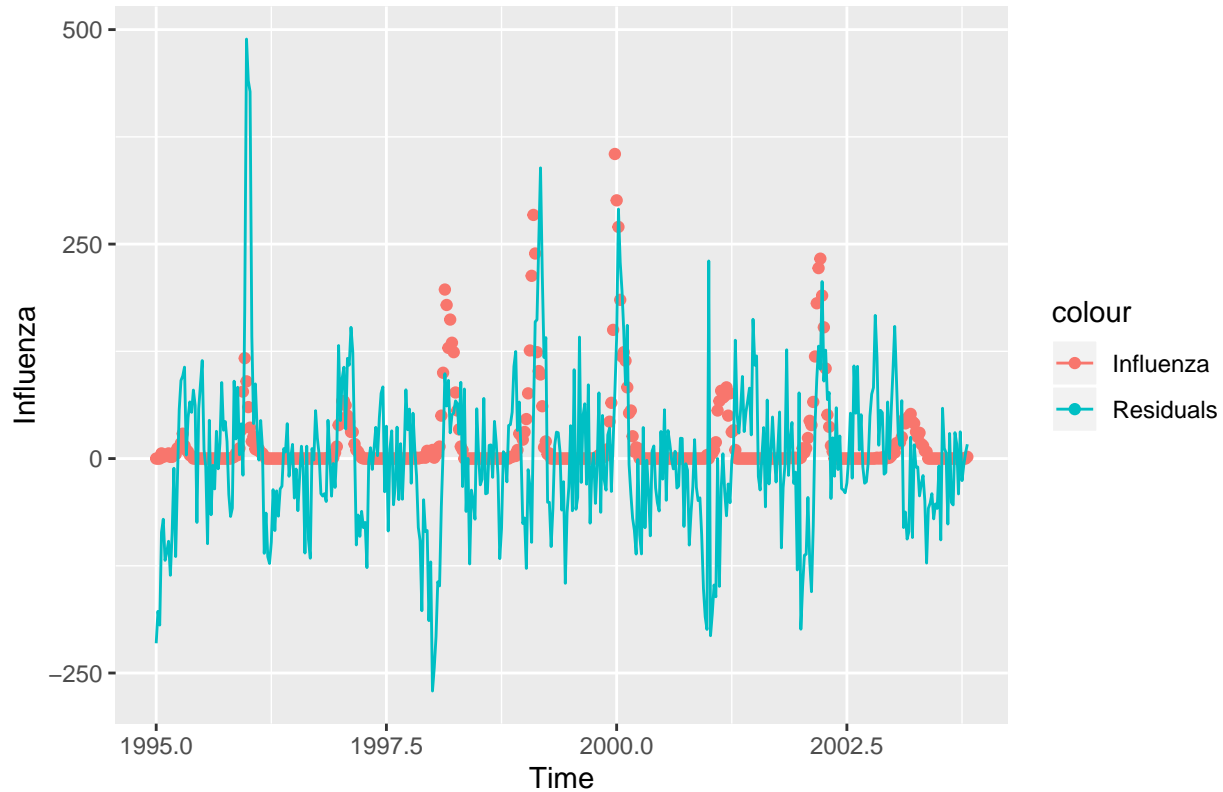
It can be seen that the lower the penalty the better the fit. In this case it looks good to have such a low penalty, though in other cases it might overfit the data.



Extracting the degrees of freedom from each penalization iteration it can be seen that the lower the penalization the more relaxed is the model, thus, the higher the degrees of freedom.

5. Use the model obtained in step 2 and plot the residuals and the influenza values against time (in one plot). Is the temporal pattern in the residuals correlated to the outbreaks of influenza?

At a first glance Residuals seem to be noise compared to Influenza behavior.



Nonetheless visual assumptions might be incorrect. Testing numerically by getting their correlation statistic is low, thus there is no evidence to say that the residuals from the model and the influenza measures are correlated.

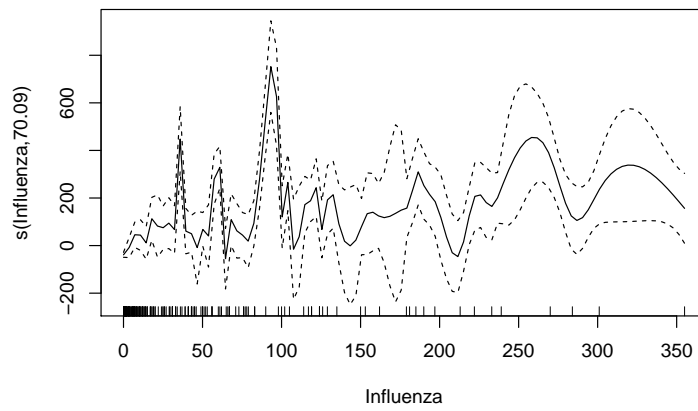
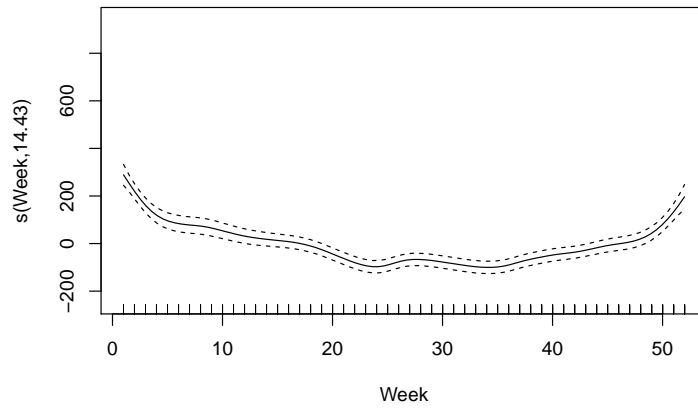
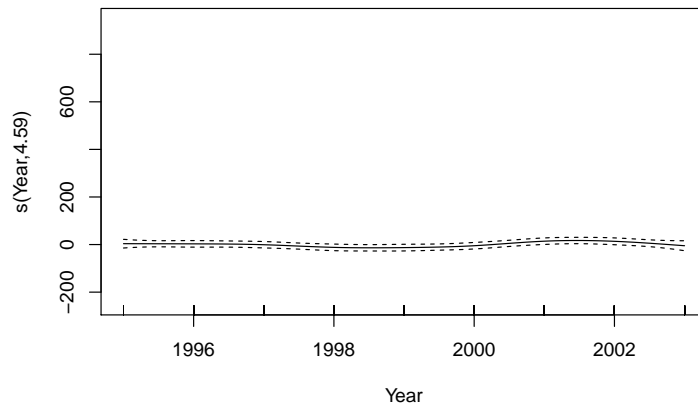
```
## [1] 0.3397395
```

6. Fit a GAM model in R in which mortality is be modelled as an additive function of the spline functions of year, week, and the number of confirmed cases of influenza. Use the output of this GAM function to conclude whether or not the mortality is influenced by the outbreaks of influenza. Provide the plot of the original and fitted Mortality against Time and comment whether the model seems to be better than the previous GAM models.

It has been trained a GAM using the following formula:

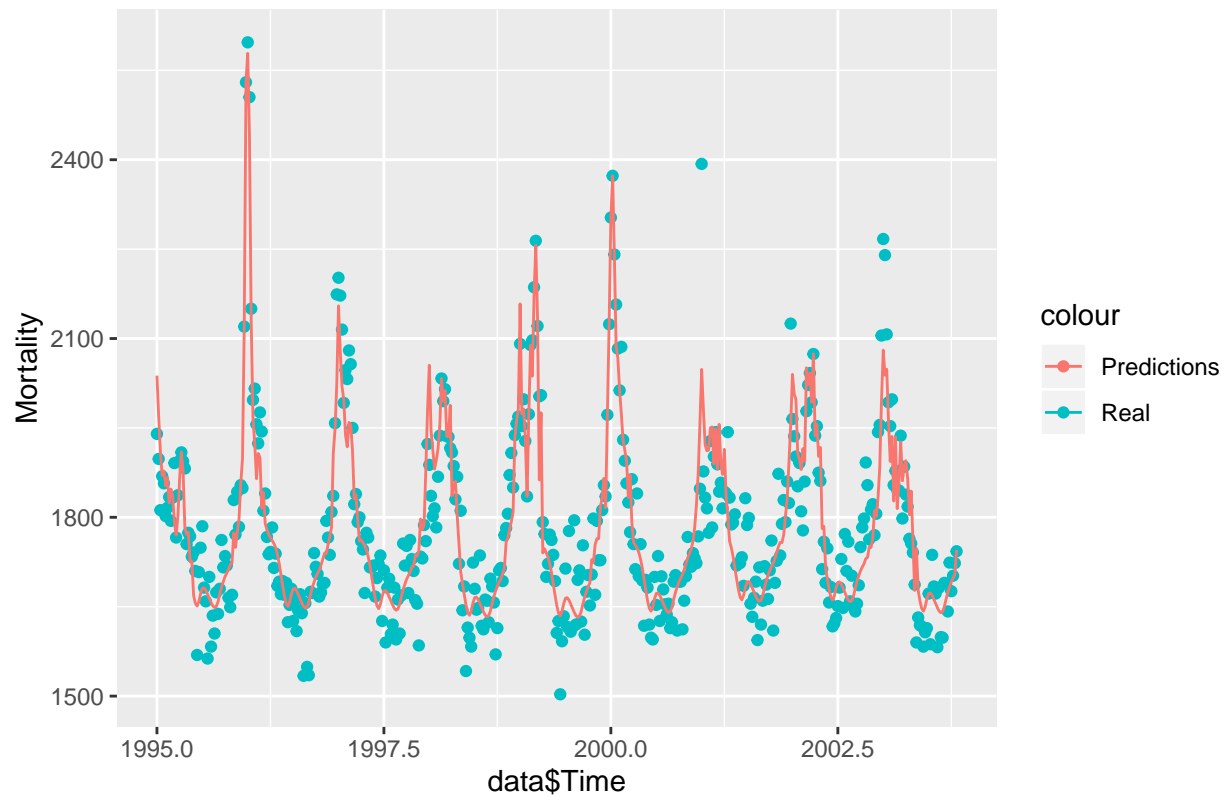
$$Mortality \approx \text{spline}(Year) + \text{spline}(Week) + \text{spline}(Influenza)$$

The obtained smoothers are the following.



Now, using the model to predict the Mortality :

Second GAM predictions



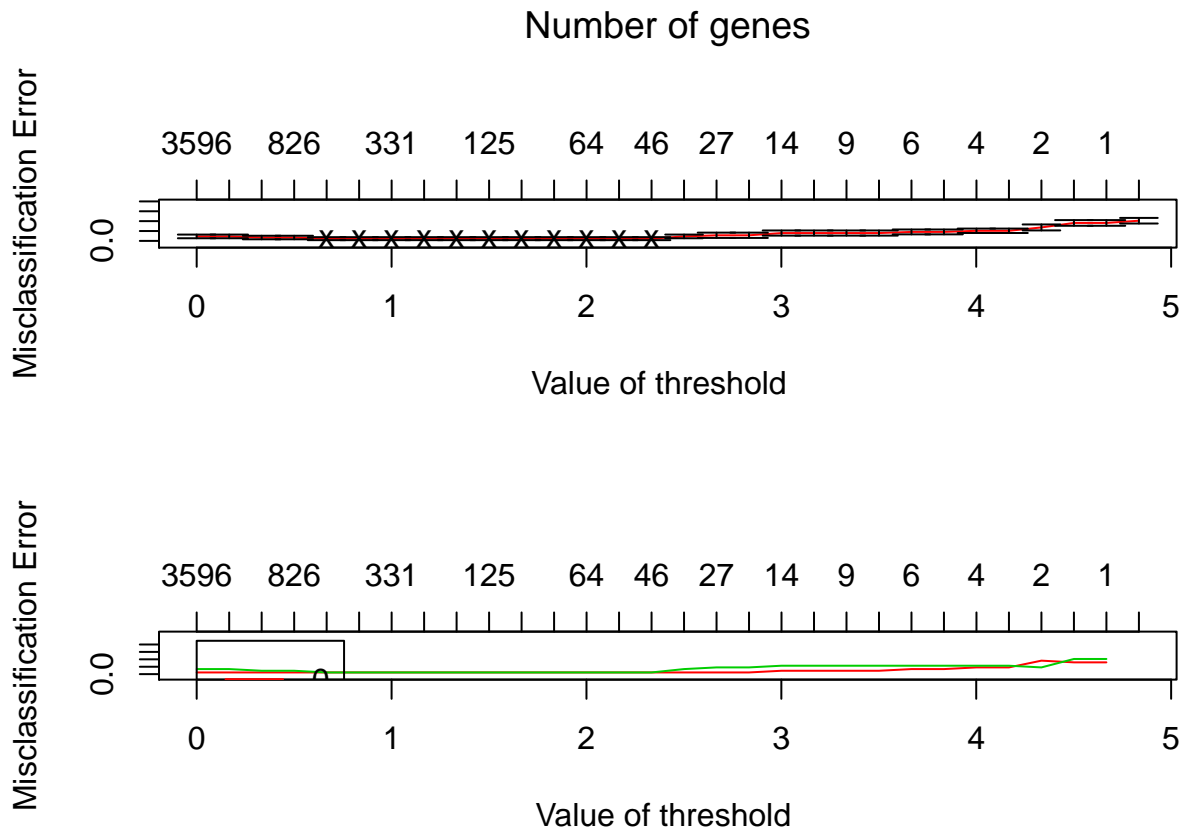
The model seems to fit better the data, though there are still some data points not being well approximated. This is sign that the predictor is not overfitted, so compared against the previous predictors, this is the best one.



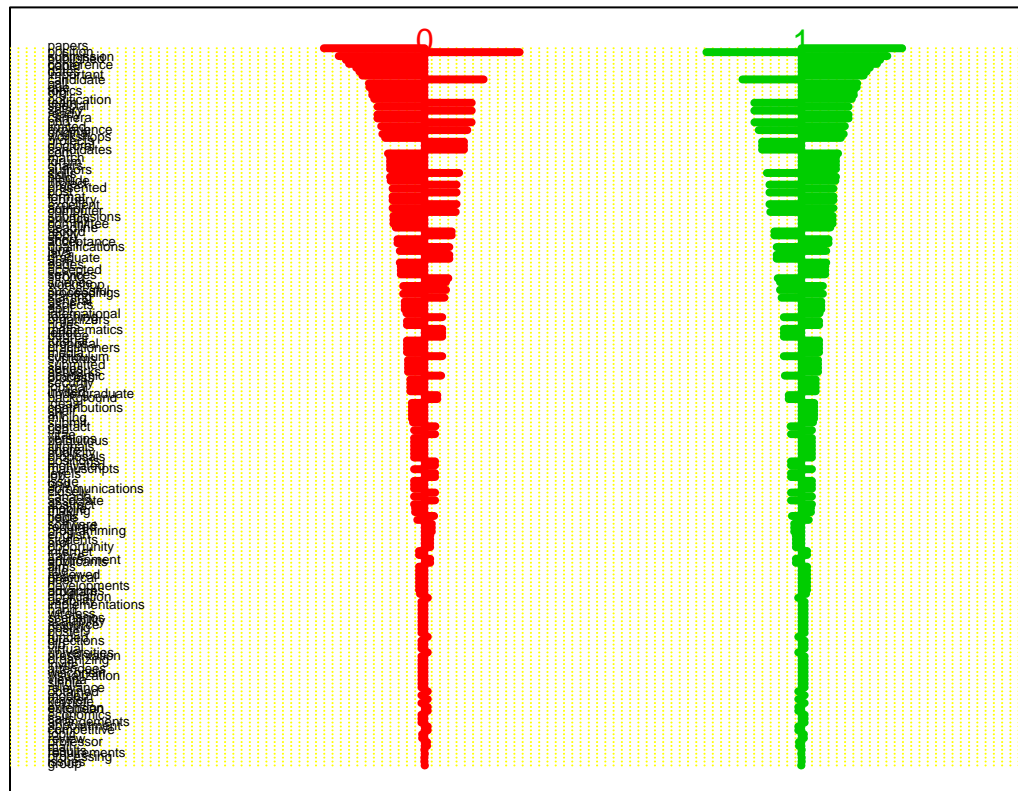
## Assignment 2. High-dimensional methods

The data file data.csv contains information about 64 e-mails which were manually collected from DBWorld mailing list. They were classified as: 'announces of conferences' (1) and 'everything else' (0) (variable Conference)

1. Divide data into training and test sets (70/30) without scaling. Perform nearest shrunken centroid classification of training data in which the threshold is chosen by cross-validation. Provide a centroid plot and interpret it. How many features were selected by the method? List the names of the 10 most contributing features and comment whether it is reasonable that they have strong effect on the discrimination between the conference mails and other mails? Report the test error.



Therefore from the crossvalidation results it can be seen that the best performance error-wise is obtained for threshold = 1.29.



```
## There were selected, 185 features
## [1] "The best 10 selected features are:"

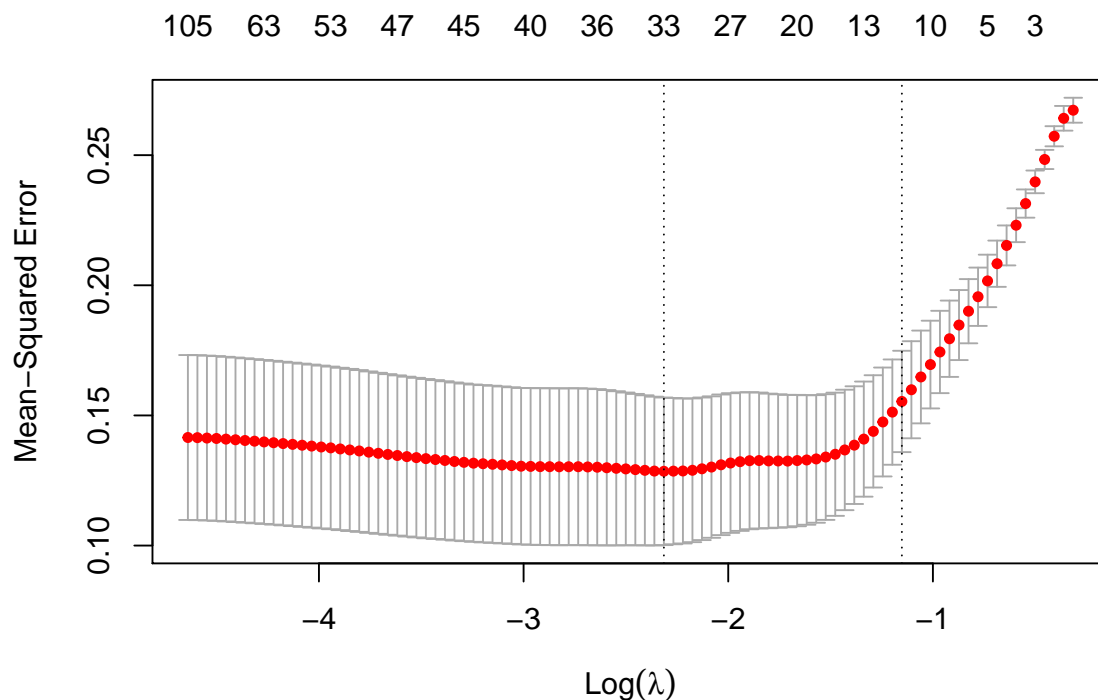
##      id      name      0-score  1-score
## [1,] "3036" "papers"    "-0.5344" "0.5344"
## [2,] "3187" "position"  "0.5043" "-0.5043"
## [3,] "4060" "submission" "-0.4556" "0.4556"
## [4,] "3364" "published" "-0.4174" "0.4174"
## [5,] "869"  "conference" "-0.4009" "0.4009"
## [6,] "3035" "paper"     "-0.3628" "0.3628"
## [7,] "1045" "dates"     "-0.3458" "0.3458"
## [8,] "2049" "important" "-0.3303" "0.3303"
## [9,] "606"  "candidate" "0.3132" "-0.3132"
## [10,] "596" "call"      "-0.2957" "0.2957"

## Classification Performance : Shrunk centroid and shit
## [1] "Confusion Matrix"
##      predictions
## targets  0  1
##      0 11  2
##      1  0  7
## Rates details:
## TPR = 77.77778 % - TNR = 100 % - FPR = 22.22222 % - FNR = 0 %
## Misclassification Rate = 10 %
```

2. Compute the test error and the number of the contributing features for the following methods fitted to the training data:

Compare the results of these models with the results of the nearest shrunken centroids (make a comparative table). Which model would you prefer and why?

- a. Elastic net with the binomial response and  $\alpha = 0.5$  in which penalty is selected by the cross-validation



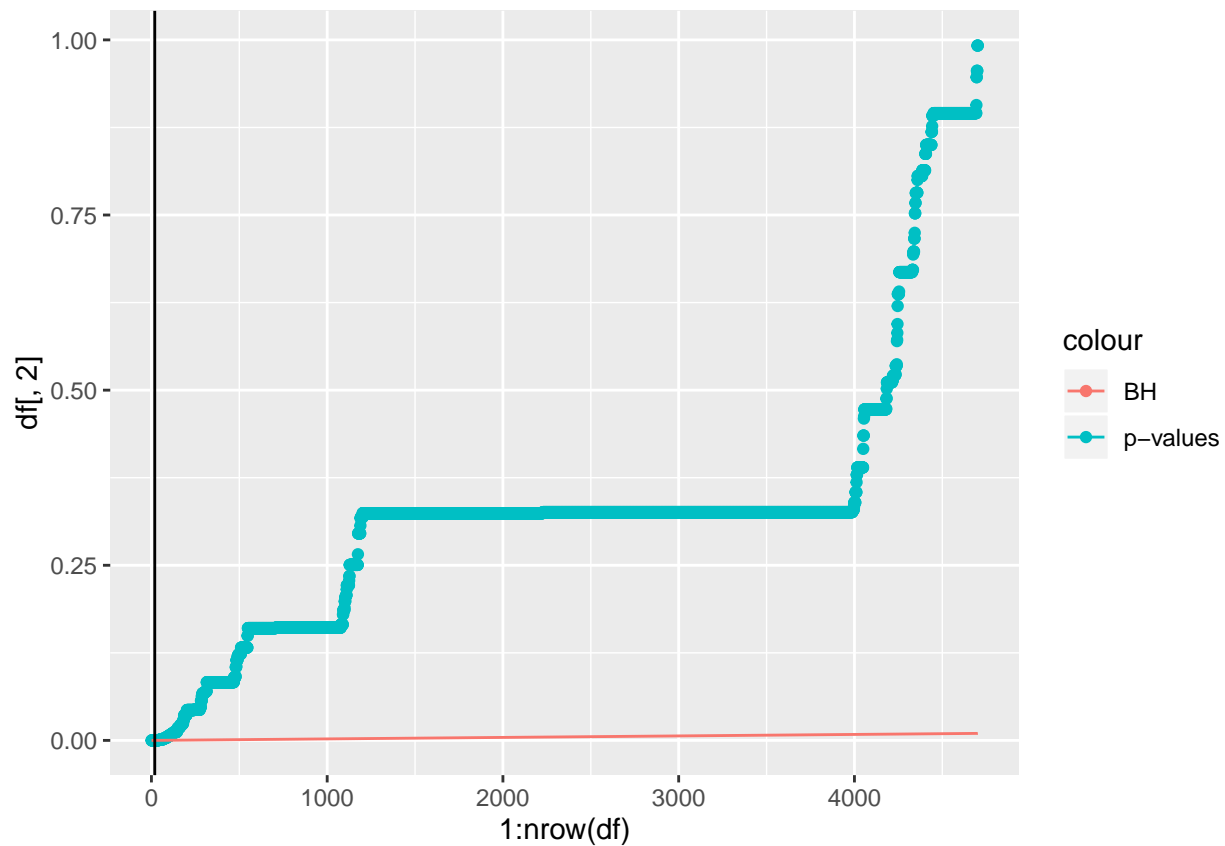
```
## Classification Performance : ElasticNet
## [1] "Confusion Matrix"
##      predictions
## targets 0  1
##      0 11  2
##      1  1  6
## Rates details:
##  TPR = 75 % - TNR = 91.66667 % - FPR = 25 % - FNR = 8.333333 %
##  Misclassification Rate = 15 %
## Elasticnet model has 165 nonzero coefficients
```

b. Support vector machine with “vanilladot” kernel.

```
## Setting default kernel parameters
## Warning in .local(x, ...): Variable(s) `` constant. Cannot scale data.
## Classification Performance : SVM - Testing performance
## [1] "Confusion Matrix"
##      predictions
## targets 0  1
##      0 11  2
##      1  1  6
## Rates details:
## TPR = 75 % - TNR = 91.66667 % - FPR = 25 % - FNR = 8.333333 %
## Misclassification Rate = 15 %
```

SVMs do not suffer the problem of highdimensional data, it will use all the features to compute the support vectors and the hyperplane fitted by them.

3. Implement Benjamini-Hochberg method for the original data, and use `t.test()` for computing p-values. Which features correspond to the rejected hypotheses? Interpret the result.



```
## Rejected Features = 19
## Rejected Hypothesis Features =
## ( 1 ) papers
## ( 2 ) submission
## ( 3 ) position
## ( 4 ) published
## ( 5 ) important
```

## ( 6 ) call  
## ( 7 ) conference  
## ( 8 ) candidates  
## ( 9 ) dates  
## ( 10 ) paper  
## ( 11 ) topics  
## ( 12 ) limited  
## ( 13 ) candidate  
## ( 14 ) camera  
## ( 15 ) ready  
## ( 16 ) authors  
## ( 17 ) phd  
## ( 18 ) projects  
## ( 19 ) org

## Appendix A : Environment setup Code

```
knitr::opts_chunk$set(echo = FALSE)
library(openxlsx)
library(ggplot2)
library(mgcv)
library(akima)
library(plotly)
library(pamr)
library(glmnet)
library(e1071)
library(kernlab)
RNGversion('3.5.1')
set.seed(12345)
```

## Appendix B : Code for Assignment 1

```
#####  
##                               Assignment 1  
#####  
  
# Import data  
dataPath <- "data/influenza.xlsx"  
data <- read.xlsx(dataPath)  
  
# Time Series plotting  
mortPlot <- ggplot(data) +  
  geom_line(aes(x=Time, y=Mortality), color="black") + ggtitle("Moratality")  
infPlot <- ggplot(data) +  
  geom_line(aes(x=Time, y=Influenza), color="black") + ggtitle("Influenza")  
tempPlot <- ggplot(data) +  
  geom_line(aes(x=Time, y=Temperature.deficit), color="black") +  
  ggtitle("Temperature Deficit")  
infPlot  
mortPlot  
tempPlot  
  
# Time series per year  
years <- unique(data$Year)  
weeks <- unique(data$Week)  
mortData <- list()  
infData <- list()  
tempData <- list()  
for(i in 1:length(years)) {  
  year <- years[i]  
  mortData[[i]] <- data$Mortality[which(data$Year == year)]  
  infData[[i]] <- data$Influenza[which(data$Year == year)]  
  tempData[[i]] <- data$Temperature.deficit[which(data$Year == year)]  
}  
names(mortData) <- years  
names(infData) <- years  
names(tempData) <- years  
  
# create data.frames for ggplot  
plotData <- function(d, title) {  
  shortWeeks <- 1:length(d$'2003')  
  p <- ggplot() +  
    geom_line(aes(x=weeks, y=d$'1995', color="1995")) +  
    geom_line(aes(x=weeks, y=d$'1996', color="1996")) +  
    geom_line(aes(x=weeks, y=d$'1997', color="1997")) +  
    geom_line(aes(x=weeks, y=d$'1998', color="1998")) +  
    geom_line(aes(x=weeks, y=d$'1999', color="1999")) +  
    geom_line(aes(x=weeks, y=d$'2000', color="2000")) +  
    geom_line(aes(x=weeks, y=d$'2001', color="2001")) +  
    geom_line(aes(x=weeks, y=d$'2002', color="2002")) +  
    geom_line(aes(x=shortWeeks, y=d$'2003', color="2003")) +  
    geom_point(aes(x=weeks, y=d$'1995', color="1995")) +  
    geom_point(aes(x=weeks, y=d$'1996', color="1996")) +  
    geom_point(aes(x=weeks, y=d$'1997', color="1997")) +
```

```

    geom_point(aes(x=weeks, y=d$'1998', color="1998")) +
    geom_point(aes(x=weeks, y=d$'1999', color="1999")) +
    geom_point(aes(x=weeks, y=d$'2000', color="2000")) +
    geom_point(aes(x=weeks, y=d$'2001', color="2001")) +
    geom_point(aes(x=weeks, y=d$'2002', color="2002")) +
    geom_point(aes(x=shortWeeks, y=d$'2003', color="2003")) +
    ggtitle(title) + ylab(title)
  return(p)
}

mortPlot <- plotData(mortData, "Mortality")
infPlot <- plotData(infData, "Influenza")
tempPlot <- plotData(tempData, "Temperature Deficit")
mortPlot
infPlot
tempPlot

# Training GAM
model <- gam (
  Mortality ~ Year +
    s(Week, k = length(unique(data$Week))),
  data=data,
  method = "GCV.Cp"
)
summary(model)
plot(model, main="Smooth Approximation of Mortality")

# Predictions
predictions <- predict(model, data)
ggplot(data) +
  geom_point(aes(x=Time, y=Mortality, color="Real")) +
  geom_line(aes(x=Time, y=predictions, color="Predicted"), size=1) +
  ggtitle("Real and Predicted Mortality")

# Plot GAM predictions per year
plotAll <- function(d, p, title) {
  shortWeeks <- 1:length(d$'2003')
  a <- ggplot() +
    geom_point(aes(x=weeks, y=d$'1995')) +
    geom_point(aes(x=weeks, y=d$'1996')) +
    geom_point(aes(x=weeks, y=d$'1997')) +
    geom_point(aes(x=weeks, y=d$'1998')) +
    geom_point(aes(x=weeks, y=d$'1999')) +
    geom_point(aes(x=weeks, y=d$'2000')) +
    geom_point(aes(x=weeks, y=d$'2001')) +
    geom_point(aes(x=weeks, y=d$'2002')) +
    geom_point(aes(x=shortWeeks, y=d$'2003')) +
    geom_line(aes(x=weeks, y=p$'1995', color="1995")) +
    geom_line(aes(x=weeks, y=p$'1996', color="1996")) +
    geom_line(aes(x=weeks, y=p$'1997', color="1997")) +
    geom_line(aes(x=weeks, y=p$'1998', color="1998")) +
    geom_line(aes(x=weeks, y=p$'1999', color="1999")) +
    geom_line(aes(x=weeks, y=p$'2000', color="2000")) +
    geom_line(aes(x=weeks, y=p$'2001', color="2001")) +

```



```

    geom_line(aes(x=weeks, y=p$'2002', color="2002")) +
    geom_line(aes(x=shortWeeks, y=p$'2003', color="2003")) +
    ggtitle(title)
  return(a)
}

preds <- list()
for (i in 1:length(years)) {
  year <- years[i]
  d <- data[which(data$Year == year),]
  preds[[i]] <- predict(model, d)
}
names(preds) <- years
p <- plotAll(mortData, preds, "Mortality Overall Real and Predictions")
p <- p + ylab("Mortality")
p

# Penalty factor analysis
penalties <- c(1e-6, 1e-3, 1, 1e3, 1e6)
predictions <- list()
estDegFreedom <- vector(length = length(penalties))
for (i in 1:length(penalties)) {
  model <- gam ( Mortality ~ Year +
                 s(Week, k = length(unique(data$Week))),
                 data=data,
                 method = "GCV.Cp",
                 sp = penalties[i]
               )
  predictions[[i]] <- predict(model, data)
  estDegFreedom[i] <- summary(model)$edf
}
df <- data.frame(
  time = data$Time,
  real = data$Mortality,
  sp_1u = predictions[[1]],
  sp_1m = predictions[[2]],
  sp_1 = predictions[[3]],
  sp_1k = predictions[[4]],
  sp_1M = predictions[[5]]
)

ggplot(df) +
  geom_point(aes(x=time, y=real), color="black") +
  geom_line(aes(x=time, y=sp_1u, color="sp = 0.000001"), size=1) +
  geom_line(aes(x=time, y=sp_1m, color="sp = 0.001"), size=1) +
  geom_line(aes(x=time, y=sp_1, color="sp = 1" ), size=1) +
  geom_line(aes(x=time, y=sp_1k, color="sp = 1000"), size=1) +
  geom_line(aes(x=time, y=sp_1M, color="sp = 1000000"), size=1) +
  ggtitle("Effects of variations on SP penalization on Spline training") +
  xlab("Time") + ylab("Mortality")

```

```

# Relation between penalty factors and degrees of freedom
ggplot() +
  geom_point(aes(x=log10(penalties), y=estDegFreedom)) +
  geom_line(aes(x=log10(penalties), y=estDegFreedom)) +
  ggtitle("Estimated Degrees of Freedom vs SP penalty") +
  ylab("Estimated Degrees of Freedom")

# Comparing model residuals against influenza cases
model <- gam (
  Mortality ~ Year +
  s(Week, k = length(unique(data$Week))),
  data=data,
  method = "GCV.Cp"
)
ggplot(data) +
  geom_point(aes(x=Time, y=Influenza, color="Influenza")) +
  geom_line(aes(x=Time, y=model$residuals, color="Residuals")) +
  ggtitle("")

# Correlation measurement
corr <- cor(model$residuals, data$Influenza)
print(corr)

# GAM 2 training
model <- gam( formula = Mortality ~ s(Year, k = length(unique(data$Year))) +
  s(Week, k = length(unique(data$Week))) +
  s(Influenza, k=length(unique(data$Influenza))),
  data = data,
  method = "GCV.Cp"
)
plot(model)

# GAM 2 predictions
predictions <- predict(model, data)
ggplot() +
  geom_point(aes(x=data$Time, y=data$Mortality, color="Real")) +
  geom_line(aes(x=data$Time, y=predictions, color="Predictions")) +
  ggtitle("Second GAM predictions") + ylab("Mortality")

```

## Appendix C : Code for Assignment 2

```
#####  
##                               Assignment 2  
#####  
dataPath <- "data/data.csv"  
# data <- read.csv2(dataPath, encoding = "latin1", sep = ";")  
data <- read.csv2(dataPath)  
## Splitting data  
splitData <- function(data, trainRate) {  
  n <- dim(data)[1]  
  idxs <- sample(1:n, floor(trainRate*n))  
  train <- data[idxs,]  
  test <- data[-idxs,]  
  return (list(train = train, test = test))  
}  
  
split <- splitData(data, .7)  
train <- split$train  
test <- split$test  
# Preprocessing  
y <- train[[ncol(train)]]  
x <- t(train[, -ncol(train)])  
mydata <- list(  
  x = x,  
  y = as.factor(y),  
  geneid = as.character(1:nrow(x)),  
  genenames = rownames(x)  
)  
  
# Training and cross-validating threshold  
model <- pamr.train(mydata)  
cvmodel <- pamr.cv(model, mydata)  
pamr.plotcv(cvmodel)  
minError <- min(cvmodel$error)  
bestThreshold <- cvmodel$threshold[which.min(cvmodel$error)]  
  
cat("The minimum error is ", minError,  
    "and it is obtained for threshold =", bestThreshold)  
# best model  
bestThreshold <- 1.29  
bestModel <- pamr.train(mydata, threshold = bestThreshold)  
pamr.plotcen(bestModel, mydata, threshold = bestThreshold)  
selectedGenes <- pamr.listgenes(bestModel, mydata,  
                               threshold=bestThreshold,  
                               genenames = TRUE)  
  
bestGenes <- selectedGenes[1:10,]  
cat("There were selected,", nrow(selectedGenes), "features\n")  
print("The best 10 selected features are:")  
print(bestGenes)  
# util function  
get_performance <- function(targets, predictions, text) {  
  cat("Classification Performance :", text, "\n")  
  t <- table(targets, predictions)
```

```

print("Confusion Matrix")
print(t)
tn <- t[1,1]
tp <- t[2,2]
fp <- t[1,2]
fn <- t[2,1]
total <- dim(test)[1]
tpr <- tp/(tp+fp) * 100
tnr <- tn/(tn+fn) * 100
fpr <- fp/(tp+fp) * 100
fnr <- fn/(tn+fn) * 100

cat("Rates details:\n")
cat(" TPR =", tpr, "% -")
cat(" TNR =", tnr, "% -")
cat(" FPR =", fpr, "% -")
cat(" FNR =", fnr, "%")
cat("\n Misclassification Rate = ", (fp+fn)/total * 100, "%\n")
}
yTest <- test[[ncol(test)]]
xTest <- t(test[, -ncol(test)])
predictions <- pamr.predict(bestModel, newx = xTest, type = "class",
                           threshold = bestThreshold)
get_performance(yTest, predictions, "Shrunken centroid and shit")

# Elastic-net
cvElastNet <- cv.glmnet(x=t(x), y=y, alpha=.5, type.measure = "mse")
plot(cvElastNet)
optLambda <- cvElastNet$lambda.min

elasticnetModel <- glmnet(x = t(x), y = y, family="binomial",
                        alpha=.5, lambda = optLambda)
predictions <- predict(elasticnetModel, newx = t(xTest), type="class")
get_performance(yTest, predictions, "ElasticNet")

coeffs <- coef(elasticnetModel)
nonzeroCoeffs <- coeffs[which(coeffs != 0)]
selectedCoeffs <- length(nonzeroCoeffs)
cat("Elasticnet model has ", selectedCoeffs, " nonzero coefficients\n")

# SVM
svmModel <- ksvm(Conference ~ ., data=train, type="C-svc", kernel="vanilladot")
svmPredictions <- predict(svmModel, test)
get_performance(yTest, svmPredictions, "SVM - Testing performance")

y <- as.factor(data[,ncol(data)])
data <- as.matrix(data[, -ncol(data)])
pvals <- as.numeric(vector(length = ncol(data)))
df <- data.frame(
  word = colnames(data),
  pvals <- pvals
)
BH <- as.numeric(vector(length = ncol(data)))

```

```

FDR <- 0.01
CL  <- 0.95

for(i in 1:length(pvals)) {
  test <- t.test(data[,i] ~ y,
                 alternative = "two.sided",
                 conf.level = CL )
  df[i,2] <- test$p.value
  BH[i] <- (FDR * i / length(pvals))
}

# order increasingly the pvalues
df <- df[order(df$pvals, decreasing = FALSE),]
df <- cbind(df, BH)
L <- max(which(df[,2] < df[,3]))

ggplot() +
  geom_point(aes(x=1:nrow(df), y=df[,2], color="p-values")) +
  geom_line(aes(x=1:nrow(df), y=df[,3], color="BH")) +
  geom_vline(xintercept = L)
cat("Rejected Features =", L, "\n",
    "Rejected Hypothesis Features =" )
for(i in 1:L) {
  cat("\t(", i, ")", as.character(df[i,1]), "\n")
}

```