lab2_block2_2.0

Zhixuan_Duan(zhidu838)

12/25/2019

Orange is the correction.

Green is the comments.

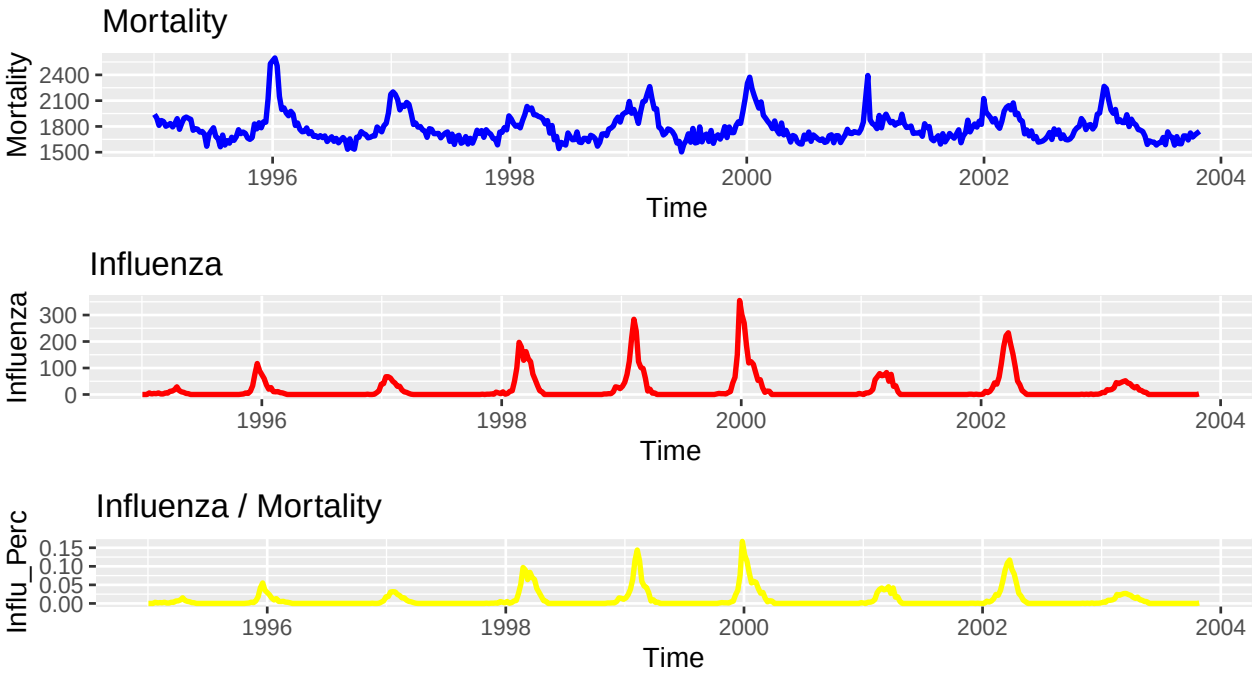
Thank you very much !

Load packages

```
library(xlsx)
library(ggplot2)
library(mgcv)
library(pamr)
library(knitr)
library(glmnet)
library(kernlab)
```

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

1.



Source: Influenza

From the third plot, we can see that Mortality and Influenza have samilar trend as time goes by. When the amounts of influenza cases rises, the mortality will increase at the same time.

And by checking the fist two plots, there are a few points seems unusual, during 1996, the influenza data is relatively lower than other time periods, but the mortality reach the highest number, and the same thing happens in 2001.

In conclusion, these two events are closely correleated, earlier seasons are likely to result in more cases of the influenza and the mortality, but there are other factors influence mortality as well.

2.

Set **family** parameter to specific mortality follows **gaussian** distribution, choose **method** as **GCV.Cp** to using Generlized cross-validation.

```
#### 1.2 ####
gam_model <- gam(data = flu, Mortality~Year+s(Week), family = gaussian(), method = "GCV.Cp")
```

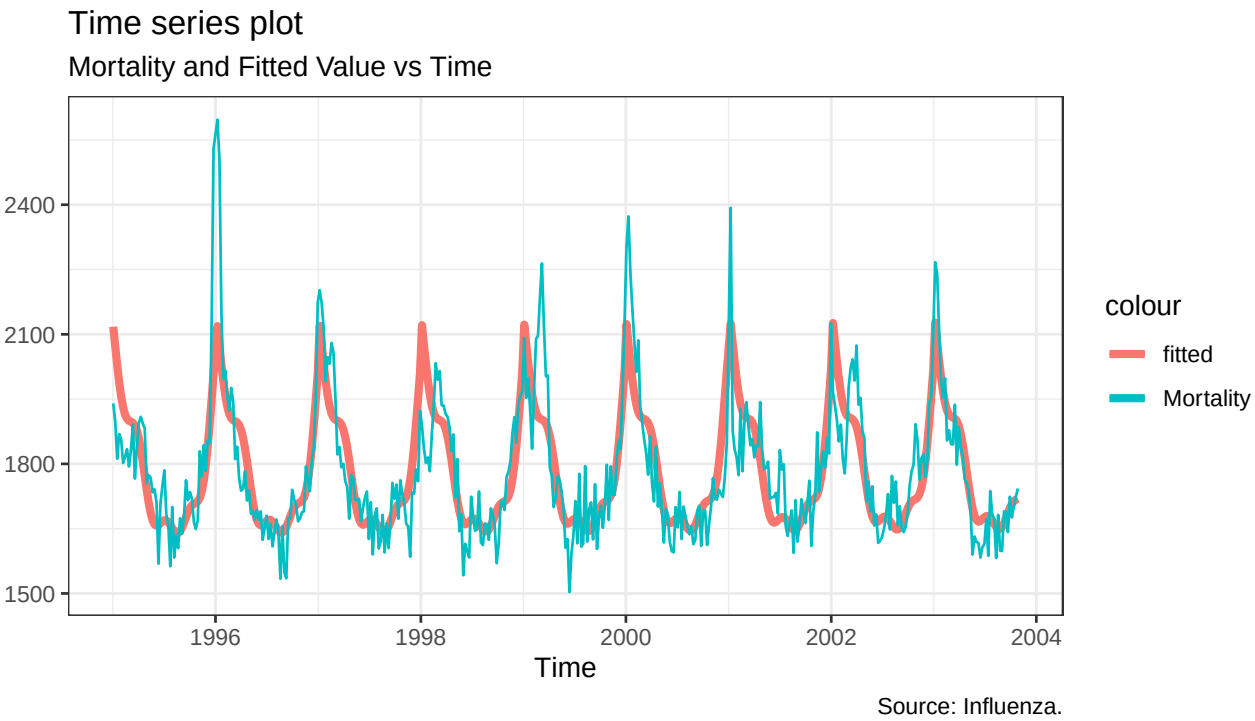
The model is as follows:

$$\hat{Mortality} = Intercept + \beta_1 Year + s(Week) + \epsilon$$

where

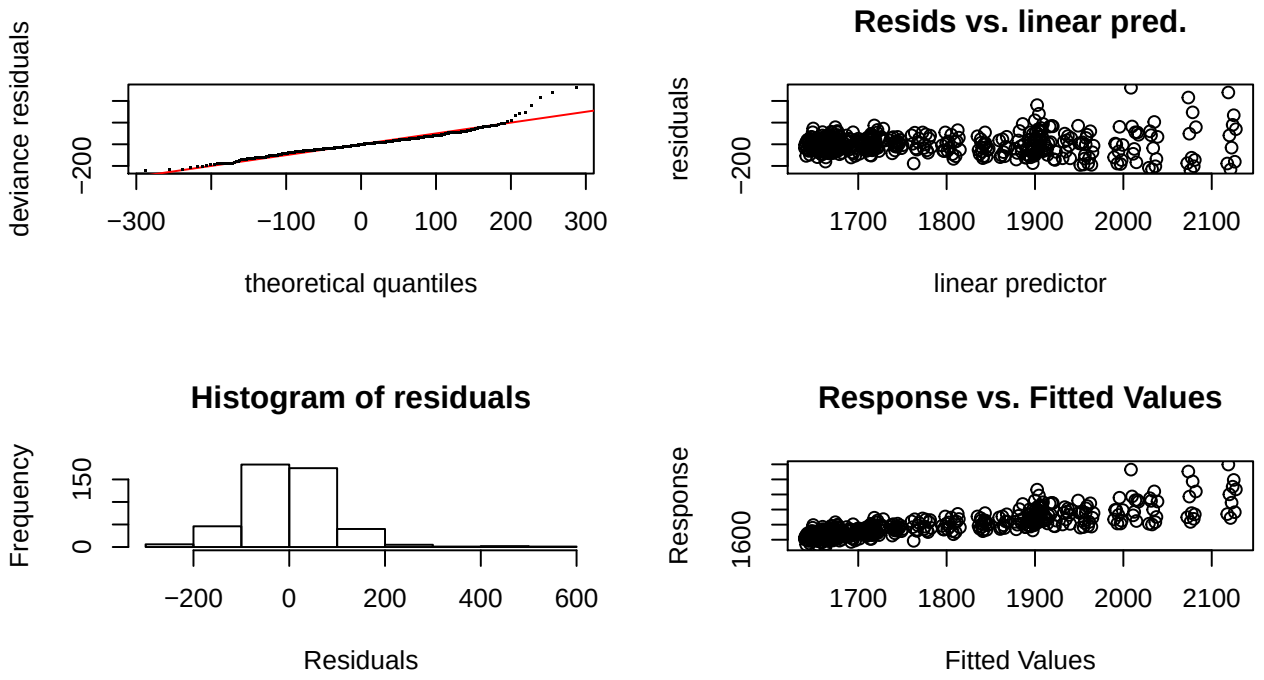
$$\epsilon \sim N(0, \sigma^2)$$

3.



From the plot we can see the model capture the trend of original dataset, but in some points, it can not fit well.

```r
# Investigate the output of the GAM model and report which terms appear to be significant
par(mfrow = c(2,2))
gam.check(gam_model)
```




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 8 iterations.
## The RMS GCV score gradient at convergence was 0.06298513 .
## The Hessian was positive definite.
## Model rank =  11 / 11
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##           k'  edf k-index p-value
## s(Week) 9.00 8.59    1.04    0.81
```

From the gam.check() output, the p value is 0.78, which indicates the k_nodes used is of sufficent size, and from the residuals plots, the residuals follow normal distribution.

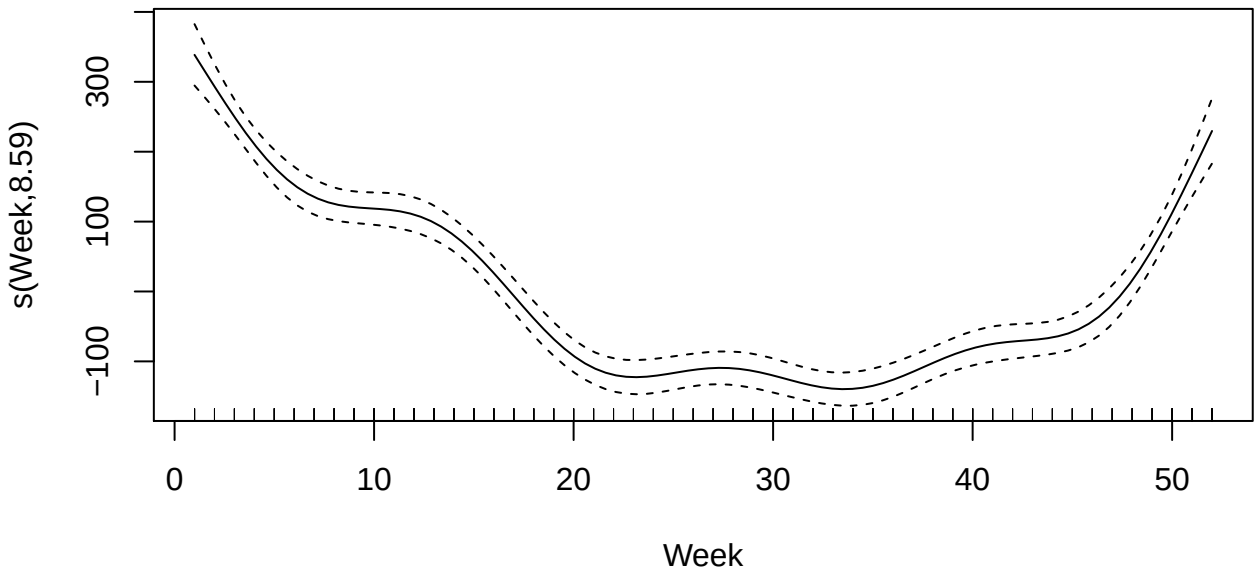
```r
summary(gam_model)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ Year + s(Week)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -652.058   3448.379  -0.189     0.85
```

```
## Year           1.219      1.725   0.706     0.48
##
## Approximate significance of smooth terms:
##           edf Ref.df     F p-value
## s(Week) 8.587  8.951 100.6  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.661   Deviance explained = 66.8%
## GCV = 9014.6  Scale est. = 8806.7    n = 459
```

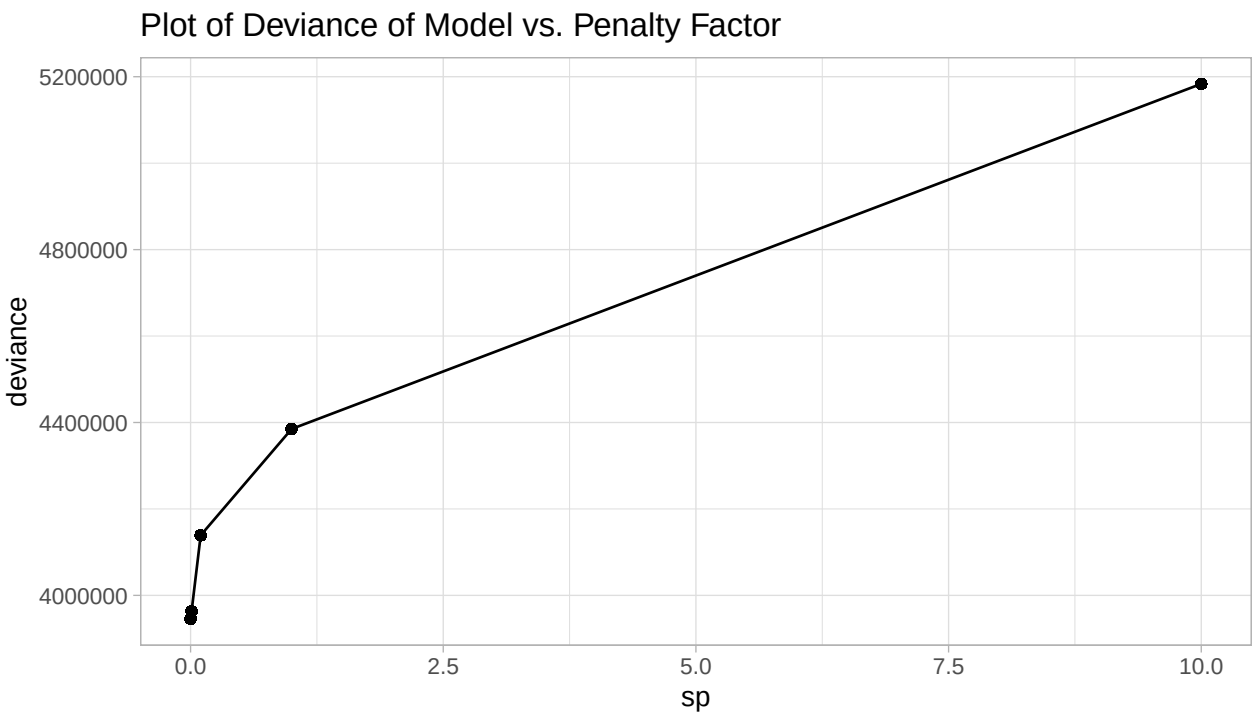
From the summary() output, we can see the overall model explains 66.1% of variance, and at a 95% confidence level, the intercept and year are not statistically significant. And there is a significant effect of week, However, the p value is approximate, so it should be handled carefully. And the effective degrees of freedom are 8.59.

```
# plot the spline component
plot(gam_model)
```

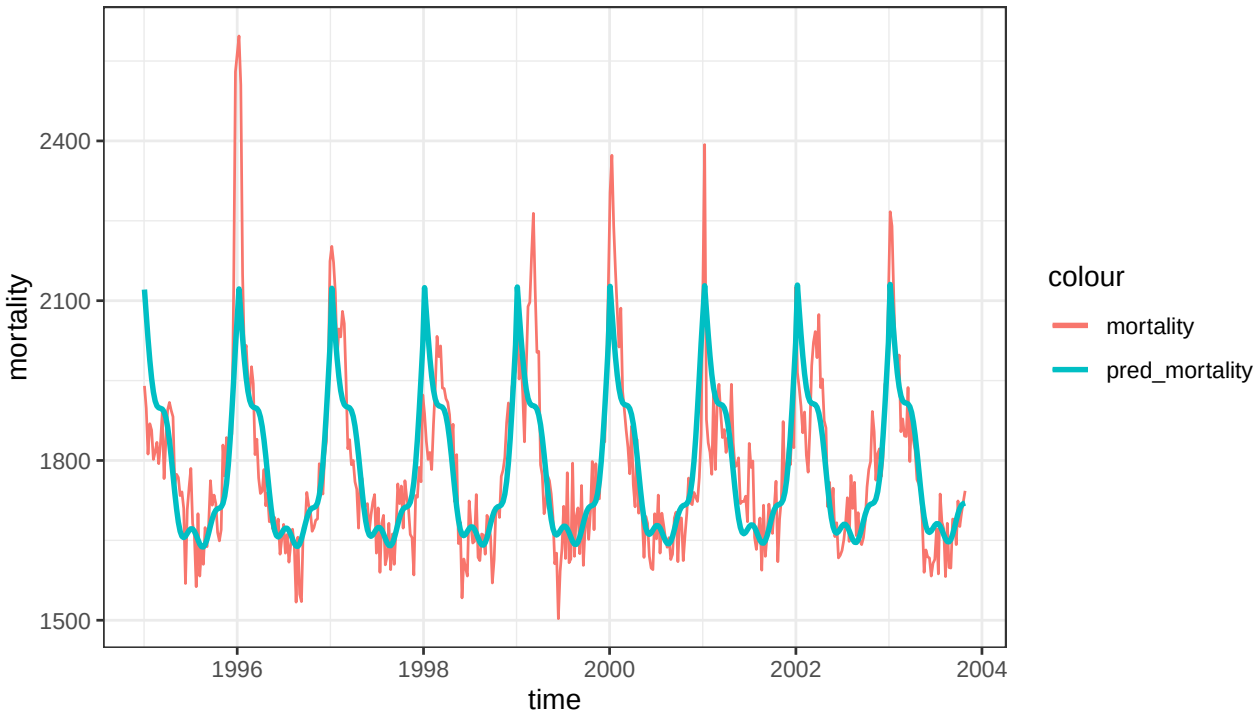


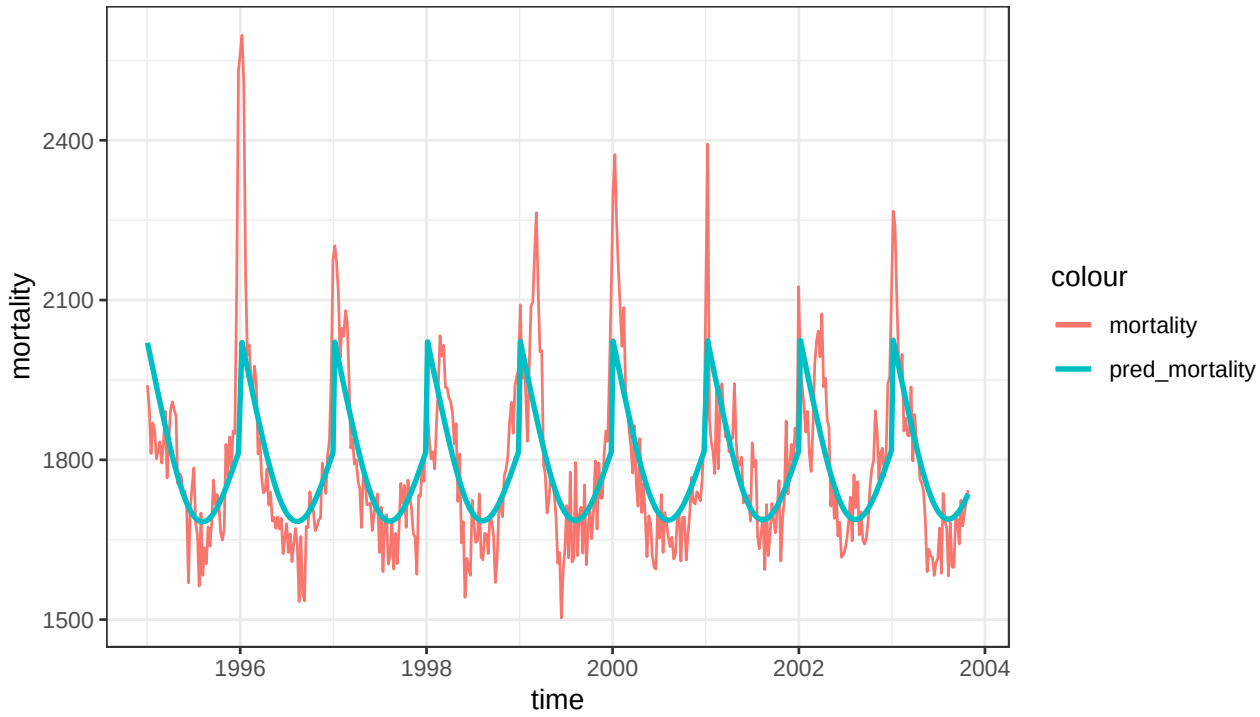
And from abline plot above, the s(week) function drops down from the beginning of ever year to June, and climb up to December, reaches the highest record.

4.

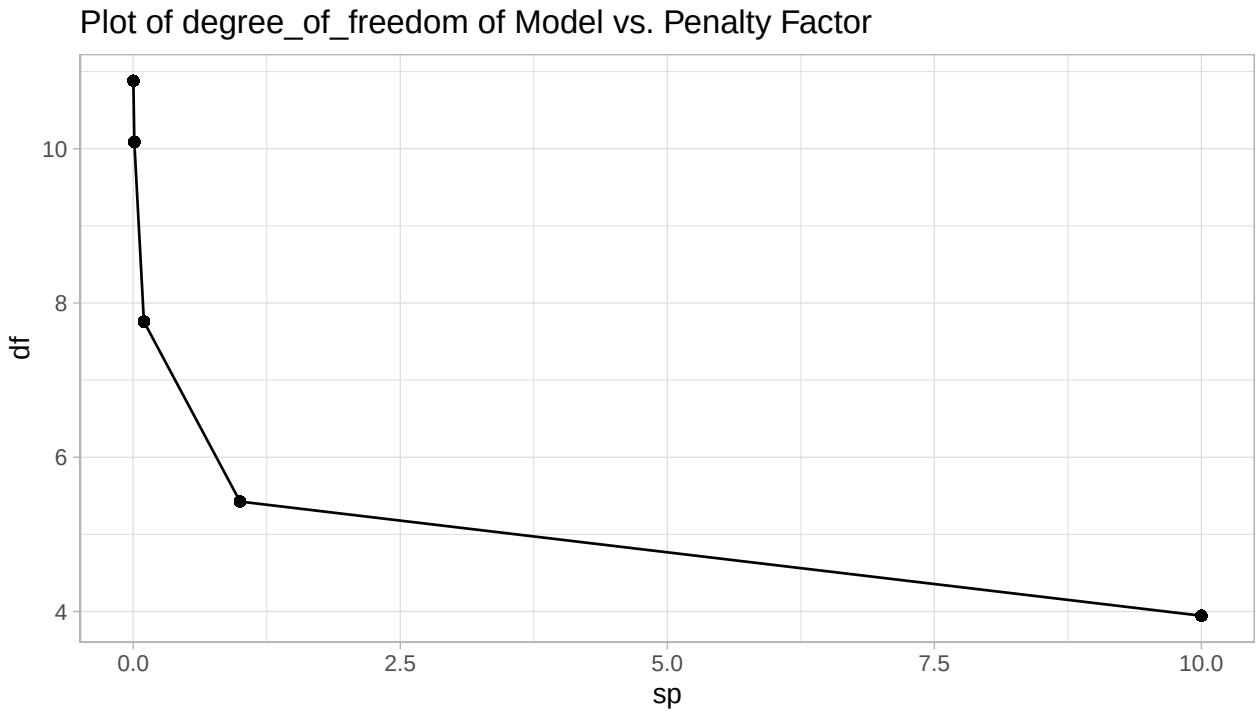


From the plot above, as penalty factor(sp) increase, the deviance increases dramatically, and sp controls the complexity of the spline, so in the beginning, it has huge impact to the whole model, but the impact will go down as main components kept in the model.



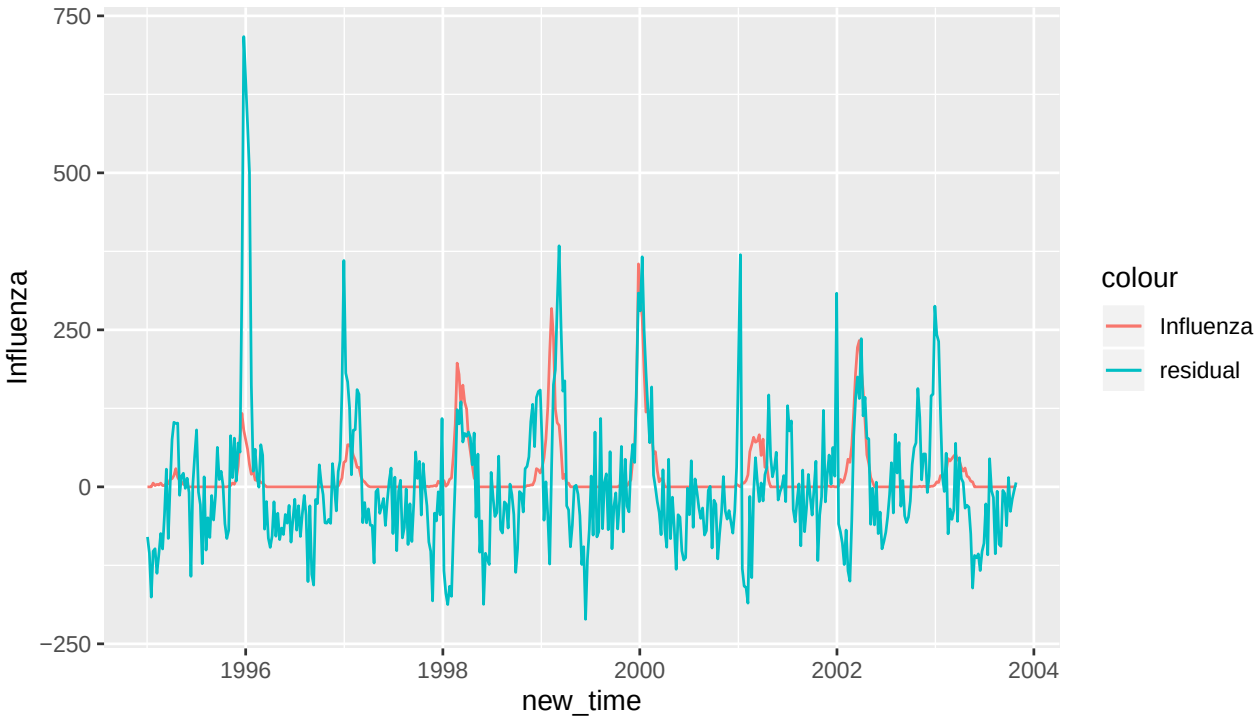


From the plots, we can see as sp increases from 0.01 to 10, the predict line become more smoother.



From the **DF vs PF** plot, we can see as sp increases, the degree of freedom of model decreases, the less complexity of model generates, so it leads to the previous plots as discussed before.

5.

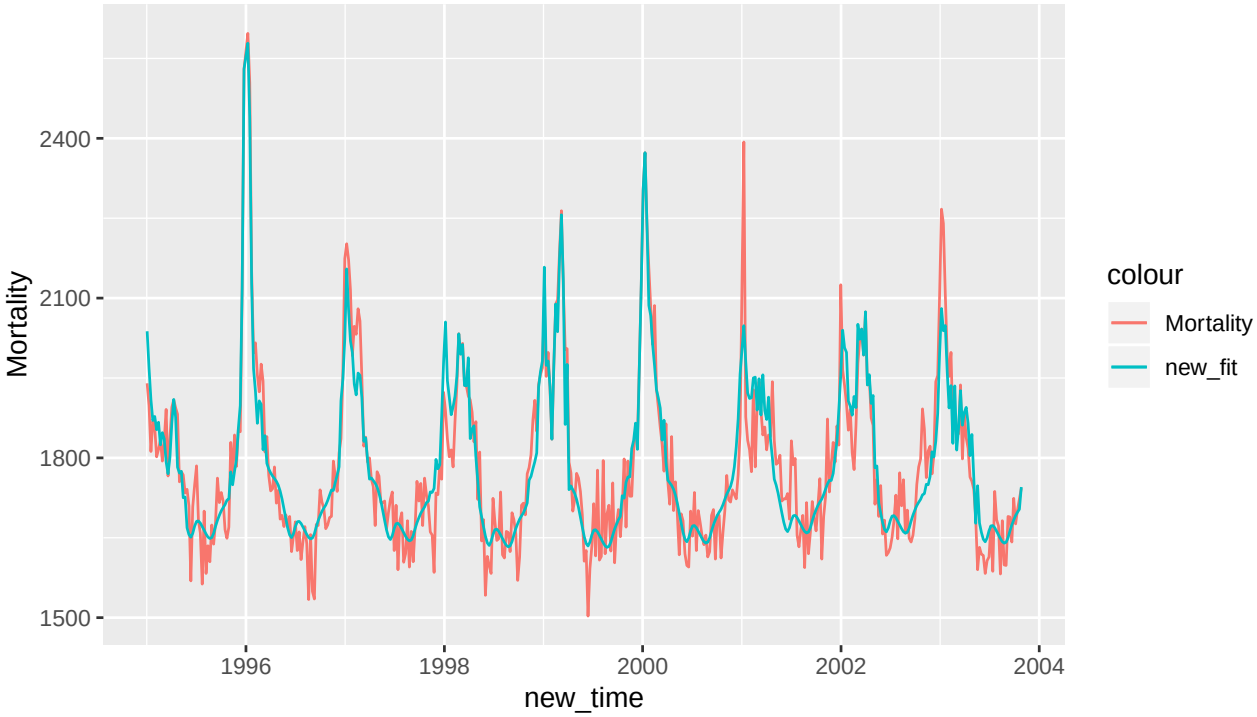


From the plot, at some specific time points, the trends of influenza and residual show similar movements, but in most of periods, these two lines do not in sync, so there is no correlation between them.

6.

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ s(Year, k = k1) + s(Week, k = k2) + s(Influenza,
##     k = k3)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1783.765      3.198   557.8   <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(Year)       4.587  5.592  1.500   0.178
## s(Week)      14.431 17.990 18.763  <2e-16 ***
## s(Influenza) 70.094 72.998  5.622  <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 = 5840.5  Scale est. = 4693.7    n = 459
```



After add **influenza** as one spline function, the new model has better performence than previous one.

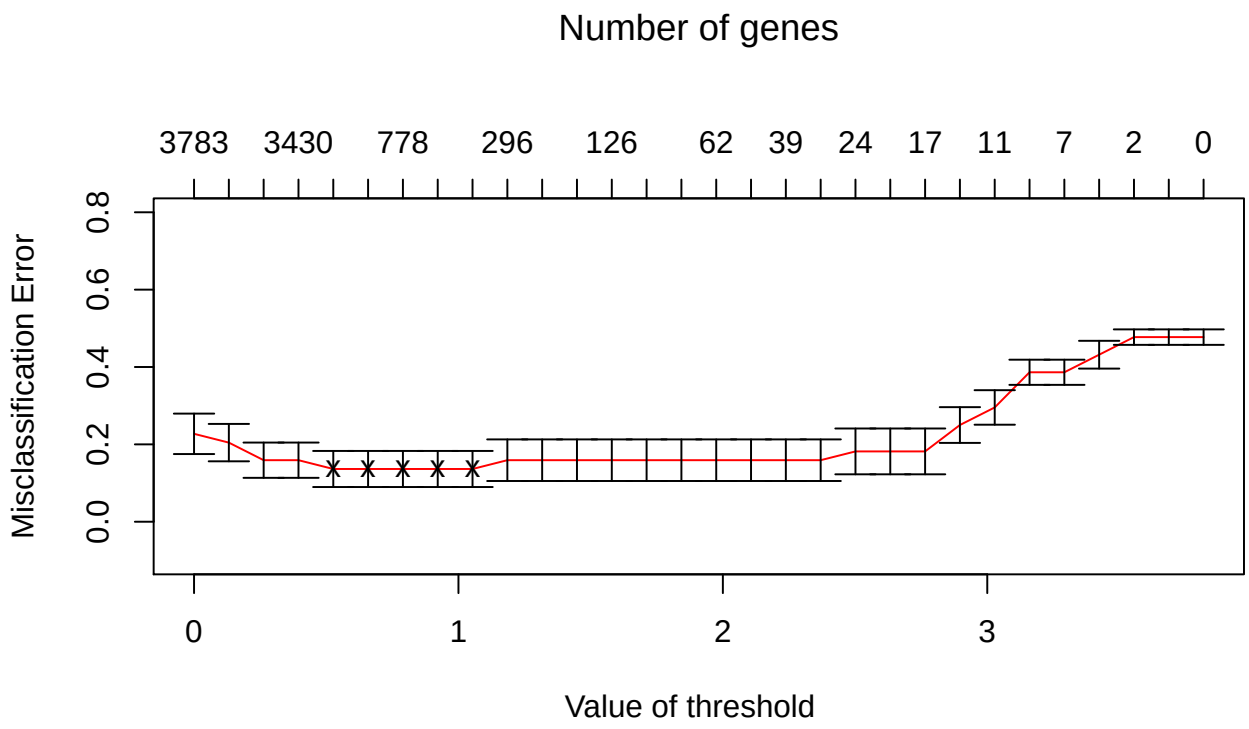
This one explains 81.9% variance of the dataset, and from the **summary()** output, we can see the **s(Influenza)** is significant.

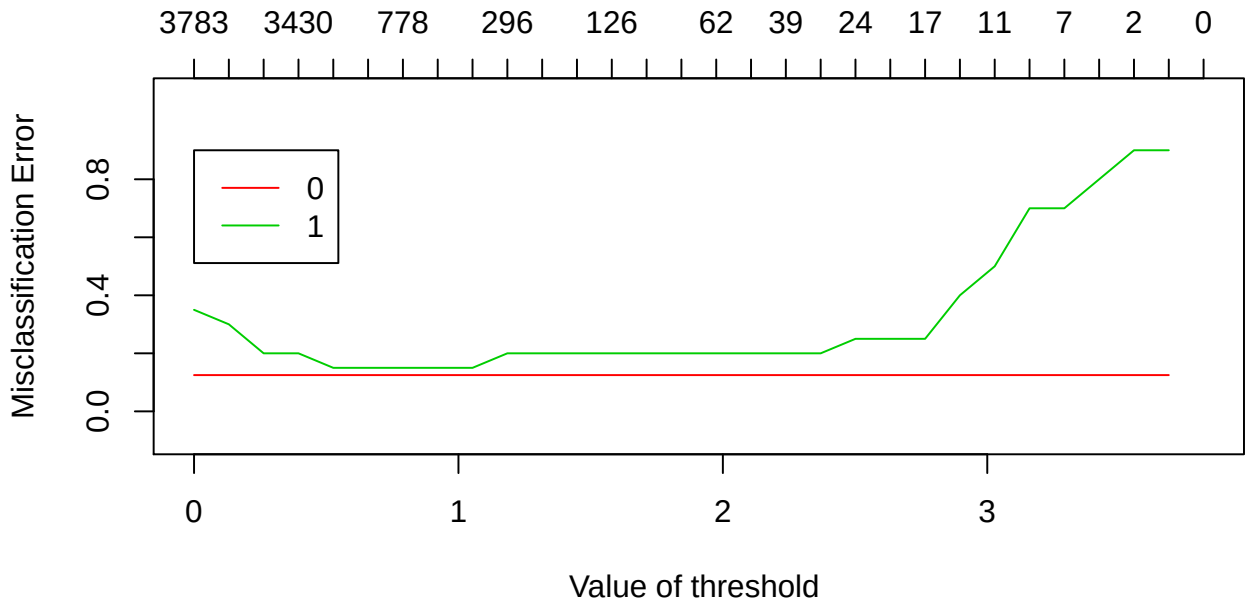
Assignment 2. High-dimensional methods

1.

```
## 1234567891011121314151617181920212223242526272829 30
```

```
## 12Fold 1 :1234567891011121314151617181920212223242526272829 30
## Fold 2 :1234567891011121314151617181920212223242526272829 30
## Fold 3 :1234567891011121314151617181920212223242526272829 30
## Fold 4 :1234567891011121314151617181920212223242526272829 30
## Fold 5 :1234567891011121314151617181920212223242526272829 30
## Fold 6 :1234567891011121314151617181920212223242526272829 30
## Fold 7 :1234567891011121314151617181920212223242526272829 30
## Fold 8 :1234567891011121314151617181920212223242526272829 30
## Fold 9 :1234567891011121314151617181920212223242526272829 30
## Fold 10 :1234567891011121314151617181920212223242526272829 30
```





From the plots, we can see when the thershold value equals 0.5 to 1.2, the misclassification error is the lowest.

```
##       id     name         0-score   1-score
##  [1,] "3036" "papers"     "-0.3878" "0.4654"
##  [2,] "3187" "position"   "0.3687"  "-0.4424"
##  [3,] "596"  "call"       "-0.3492" "0.419"
##  [4,] "4282" "topics"     "-0.3393" "0.4072"
##  [5,] "1045" "dates"      "-0.3386" "0.4063"
```

```
##  [6,] "3364" "published"  "-0.3386" "0.4063"
##  [7,] "869"  "conference" "-0.3344" "0.4013"
##  [8,] "4628" "workshop"   "-0.3016" "0.3619"
##  [9,] "1262" "due"        "-0.3016" "0.3619"
## [10,] "810"  "committee"  "-0.2906" "0.3487"
```

The most important ten features of the model are given in the results above.

```
## Confusion Matrix and Statistics
##
##          Predict
## Original  0  1
##        0 11  0
##        1  1  8
##
##                Accuracy : 0.95
##                  95% CI : (0.7513, 0.9987)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.000524
##
##                   Kappa : 0.898
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9167
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.8889
##              Prevalence : 0.6000
##          Detection Rate : 0.5500
##    Detection Prevalence : 0.5500
##       Balanced Accuracy : 0.9583
##
##        'Positive' Class : 0
##
```

The test error is 5%(accuracy is 95%), and the p-value is 0.0005.

2.

Table 1: Contributing features in the elastic model

name	coefficient
(Intercept)	-0.4998350
call	0.2151775
committee	0.0079551
conference	0.1523930
dates	0.1652610
due	0.1055721
papers	0.1673090
phd	-0.0211337
position	-0.2706216
published	0.0543361
record	-0.0775107
topics	0.1453442

name	coefficient
workshop	0.0741774
workshops	0.1217586

From the table above, there are 13 features in the elastic model.

```
## Confusion Matrix and Statistics
##
##              Test_Predict
## Test_Original  0  1
##             0 11  0
##             1  1  8
##
##                Accuracy : 0.95
##                  95% CI : (0.7513, 0.9987)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.000524
##
##                   Kappa : 0.898
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9167
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.8889
##              Prevalence : 0.6000
##          Detection Rate : 0.5500
##    Detection Prevalence : 0.5500
##       Balanced Accuracy : 0.9583
##
##        'Positive' Class : 0
##
```

This is the confusion matrix when deploying elastic model on test dataset.

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

```
## Confusion Matrix and Statistics
##
##            Predicted SVM Test
## Actual Test  0  1
##           0 10  1
##           1  2  7
##
##                Accuracy : 0.85
##                  95% CI : (0.6211, 0.9679)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.01596
##
##                   Kappa : 0.6939
##
##  Mcnemar's Test P-Value : 1.00000
##
##             Sensitivity : 0.8333
##             Specificity : 0.8750
##          Pos Pred Value : 0.9091
##          Neg Pred Value : 0.7778
##              Prevalence : 0.6000
##          Detection Rate : 0.5000
##    Detection Prevalence : 0.5500
```

```
##       Balanced Accuracy : 0.8542
##
##        'Positive' Class : 0
##
```

This is the confusion matrix when deploying support vector machine on test dataset.

Table 2: Comparsion of Models on Test dataset

	Nearest Shrunken Centroid Model	ElasticNet Model	SVM Model
Test Error	0.05	0.05	0.15
Number of Features	10	13	4702

This is the comparison table between these three models.

And I would choose the **Nearest Shrunken Centroid model** here, since it uses the least amount of feature, and gets the lowest test error.

3.

```
##          feature      P_value         bh_p
## 1         papers 1.116910e-10 5.251710e-07
## 2     submission 7.949969e-10 1.869038e-06
## 3       position 8.219362e-09 1.288248e-05
## 4      published 1.835157e-07 2.157227e-04
## 5      important 3.040833e-07 2.859600e-04
## 6           call 3.983540e-07 3.121767e-04
## 7     conference 5.091970e-07 3.420349e-04
## 8     candidates 8.612259e-07 5.061856e-04
## 9          paper 1.398619e-06 6.576305e-04
## 10         dates 1.398619e-06 7.307005e-04
## 11        topics 5.068373e-06 2.166499e-03
## 12       limited 7.907976e-06 3.098609e-03
## 13     candidate 1.190607e-05 4.306335e-03
## 14       authors 2.154461e-05 6.331422e-03
## 15         ready 2.099119e-05 6.580038e-03
## 16        camera 2.099119e-05 7.050040e-03
## 17      projects 3.499123e-05 9.140486e-03
## 18           org 3.742010e-05 9.260491e-03
## 19           phd 3.382671e-05 9.356069e-03
## 20        chairs 5.860175e-05 1.377727e-02
## 21      original 6.488781e-05 1.386829e-02
## 22  notification 6.882210e-05 1.406963e-02
## 23           due 6.488781e-05 1.452869e-02
## 24        salary 7.971981e-05 1.561844e-02
## 25        skills 9.090038e-05 1.643898e-02
## 26        record 9.090038e-05 1.709654e-02
## 27          held 1.529174e-04 2.663028e-02
## 28          team 1.757570e-04 2.951462e-02
## 29 international 2.295684e-04 3.084087e-02
## 30         apply 2.166414e-04 3.086811e-02
## 31        strong 2.246309e-04 3.106513e-02
## 32   proceedings 2.117020e-04 3.110696e-02
## 33      workshop 2.007353e-04 3.146191e-02
```

```
## 34     committee 2.117020e-04 3.211041e-02
## 35         pages 2.007353e-04 3.254681e-02
## 36     presented 3.765147e-04 4.539416e-02
## 37          post 3.762328e-04 4.655386e-02
## 38     excellent 3.762328e-04 4.781207e-02
## 39        degree 3.762328e-04 4.914019e-02
```

From the list above, after using BH to adjust the p-value, only 39 of 4702 features are suitable when constructing model.

Appendix

```
knitr::opts_chunk$set(echo = FALSE,
                      warning = FALSE,
                      message = FALSE,
                      fig.width = 7,
                      fig.height = 4,
                      fig.align = 'center')
library(xlsx)
library(ggplot2)
library(mgcv)
library(pamr)
library(knitr)
library(glmnet)
library(kernlab)
#### 1.1 ####
set.seed(12345)
# import data
# change the format of time and add new column to see the relationship between mo and influ
flu <- read.xlsx("/Users/darin/Desktop/ML/Lab/Influenza.xlsx", sheetName = "Raw data")
flu$new_time <- as.Date(paste(flu$Year, flu$Week, 1, sep="-"), "%Y-%U-%u")
flu$influ_perc <- (flu$Influenza/flu$Mortality)

# plot
p1 <- ggplot(flu, aes(x=new_time, y = Mortality)) +
  geom_line(col = "blue", size = 1) +
  labs(title = "Mortality",
       x = "Time",
       y = "Mortality")

p2 <- ggplot(flu, aes(x=new_time, y = Influenza)) +
  geom_line(col = "red", size = 1) +
  labs(title = "Influenza",
       x = "Time",
       y = "Influenza")

p3 <- ggplot(flu, aes(x=new_time, y = influ_perc)) +
  geom_line(col = 'yellow', size = 1) +
  labs(title = "Influenza / Mortality",
       x = "Time",
       y = "Influ_Perc",
       caption = "Source: Influenza")

gridExtra::grid.arrange(p1, p2, p3, ncol=1)
#### 1.2 ####
gam_model <- gam(data = flu, Mortality~Year+s(Week), family = gaussian(), method = "GCV.Cp")
#### 1.3 ####
# Plot predicted and observed mortality against time
flu$fitted <- gam_model$fitted.values

p4 <- ggplot(flu) +
  geom_line(aes(x=new_time, y = fitted, col = 'fitted'), size = 1.5) +
  geom_line(aes(x=new_time, y = Mortality, col = 'Mortality')) +
  theme_bw() +
```

```
  labs(title = "Time series plot",
       subtitle = "Mortality and Fitted Value vs Time", x = "Time",
       y = NULL,
       caption = "Source: Influenza.")
p4
# Investigate the output of the GAM model and report which terms appear to be significant
par(mfrow = c(2,2))
gam.check(gam_model)
summary(gam_model)
# plot the spline component
plot(gam_model)
#### 1.4 ####
set.seed(12345)
model_deviance <- NULL

a <- 0.0001
b <- seq(1:5)
c <- a * 10^b

for(sp in c){
  gam_model <- mgcv::gam(data = flu, Mortality~Year+s(Week, sp=sp), method = "GCV.Cp")
  temp <- cbind(gam_model$deviance,
                gam_model$fitted.values,
                gam_model$y,
                flu$new_time,
                sp,
                sum(influence(gam_model))
                )
  model_deviance <- rbind(temp, model_deviance)
}

model_deviance <- as.data.frame(model_deviance)
colnames(model_deviance) <- c("deviance", "pred_mortality", "mortality", "time",
                              "sp", "df")
model_deviance$time <- as.Date(model_deviance$time, origin = '1970-01-01')

# plot of deviance
p6 <- ggplot(data=model_deviance, aes(x = sp, y = deviance)) +
  geom_point() +
  geom_line() +
  theme_light() +
  ggtitle("Plot of Deviance of Model vs. Penalty Factor")
p6

# plot of predicted vs. observed mortality
sp0.001 <- model_deviance[model_deviance$sp == 0.001,]
p_sp0.001 <- ggplot(sp0.001) +
  geom_line(aes(x = time, y = mortality, col = "mortality")) +
  geom_line(aes(x = time, y = pred_mortality, col = 'pred_mortality'), size = 1) +
  theme_bw()
p_sp0.001

sp10 <- model_deviance[model_deviance$sp == 10,]
```

```r
p_sp10 <- ggplot(sp10) +
  geom_line(aes(x = time, y = mortality, col = 'mortality')) +
  geom_line(aes(x = time, y = pred_mortality, col = 'pred_mortality'), size = 1) +
  theme_bw()
p_sp10
# plot of degree of freedom
p7 <- ggplot(data=model_deviance, aes(x = sp, y = df)) +
  geom_point() +
  geom_line() +
  theme_light() +
  ggtitle("Plot of degree_of_freedom of Model vs. Penalty Factor")
p7
#### 1.5 ####
flu$residual <- gam_model$residuals
p10 <- ggplot(flu) +
  geom_line(aes(x = new_time, y = Influenza, col = "Influenza")) +
  geom_line(aes(x = new_time, y = residual, col = 'residual'))
p10
#### 1.6 ####
k1 = length(unique(flu$Year))
k2 = length(unique(flu$Week))
k3 = length(unique(flu$Influenza))
gam_model_additive <- gam(data = flu, Mortality ~ s(Year, k=k1) +
                                                  s(Week, k=k2) +
                                                  s(Influenza, k=k3), method = 'GCV.Cp')
summary(gam_model_additive)


flu$new_fit = gam_model_additive$fitted.values
p11 <- ggplot(data = flu) +
  geom_line(aes(x = new_time, y = Mortality, col = "Mortality")) +
  geom_line(aes(x = new_time, y = new_fit, col = "new_fit"))
p11
#### 2.1 ####
data <- read.csv(file = "/Users/darin/Desktop/ML/Lab/data.csv",
                 sep =';',
                 fileEncoding="ISO-8859-1")
data$Conference <- as.factor(data$Conference)

# divide data into train/test (70/30)
set.seed(12345)
n <- dim(data)[1]
id=sample(1:n, floor(n*0.7))
train <- data[id,]
test <- data[-id,]

x_train <- t(as.matrix(train[,-4703]))
y_train <- as.factor(train[,4703])
xy_train <- list(x = x_train,
                 y = y_train,
                 geneids = 1:nrow(x_train),
                 genenames = rownames(x_train))
```

```r
x_test <- t(as.matrix(test[,-4703]))
y_test <- as.factor(test[,4703])
xy_test <- list(x = x_test,
                y = y_test,
                geneids = 1:nrow(x_test),
                genenames = rownames(x_test))

pam_train <- pamr.train(xy_train)
pam_results <- pamr.cv(pam_train,xy_train)
#Plot the cross-validated error curves
pamr.plotcv(pam_results)
import_feature <- pamr.listgenes(pam_train, xy_train, threshold = 1,genenames = TRUE)[1:10,]
import_feature
#Compute the confusion matrix for test dataset
pred <- pamr.predict(pam_train, newx = x_test, threshold = 1)

conf_scc <- table(y_test, pred)
names(dimnames(conf_scc)) <- c("Original", "Predict")
result_scc <- caret::confusionMatrix(conf_scc)
result_scc
#### 2.2 ####
x_train1 <- t(x_train)
x_test1 <- t(x_test)

set.seed(12345)
# get the contributing features
cvfit <- cv.glmnet(x = x_train1, y = y_train, alpha = 0.5, family = "binomial",nfolds = 10)
tmp_coeffs <- coef(cvfit)
elastic_variable <- data.frame(name = tmp_coeffs@Dimnames[[1]][tmp_coeffs@i + 1], coefficient = tmp_coe
knitr::kable(elastic_variable, caption = "Contributing features in the elastic model")
# predict
pred_elastic <- predict(cvfit, newx = x_test1, type = "class")
conf_elastic_net <- table(y_test, pred_elastic)
names(dimnames(conf_elastic_net)) <- c("Test_Original", "Test_Predict")
result_ela <- caret::confusionMatrix(conf_elastic_net)
result_ela
# svm
svm_fit <- ksvm(x_train1, y_train, kernel="vanilladot", scale = FALSE, type = "C-svc")
# predict
pred_svm <- predict(svm_fit, x_test1, type="response")
conf_svm <- table(y_test, pred_svm)
names(dimnames(conf_svm)) <- c("Actual Test", "Predicted SVM Test")
result_svm <- caret::confusionMatrix(conf_svm)
result_svm
# creating table
test_error <- as.data.frame(cbind((1-result_scc$overall[[1]]),
                                  (1-result_ela$overall[[1]]),
                                  (1-result_svm$overall[[1]])))
features_count <- as.character(cbind(nrow(import_feature),
                                     (nrow(elastic_variable)-1),
                                     (ncol(data)-1)))
final_result <- rbind(test_error, features_count)
```

```r
colnames(final_result) <- c("Nearest Shrunken Centroid Model",
                            "ElasticNet Model",
                            "SVM Model")
rownames(final_result) <- c("Test Error", "Number of Features")
knitr::kable(final_result, caption = "Comparsion of Models on Test dataset")
#### 2.3 ####
x <- as.matrix(data[,-4703])
y <- as.factor(data[,4703])
# compute p value for each feature
p_values <- data.frame(feature = '',P_value = 0,stringsAsFactors = FALSE)
for(i in 1:ncol(x)){
  res = t.test(x[,i]~y,
               data = data,
               alternative="two.sided",
               conf.level = 0.95)
  p_values[i,] <- c(colnames(x)[i],res$p.value)
}
p_values$P_value <- as.numeric(p_values$P_value)
# reorder the p value
p_new <- p_values[order(p_values$P_value,decreasing = F),]
# add the p rank column
p_new$p_rank <- 1:nrow(p_new)
# compute the BH p value
p_new$bh_p <- (p_new$P_value/p_new$p_rank) * nrow(p_new)
# get the most significant features
temp <- p_new[which(p_new$bh_p <= 0.05),]

temp1 <- temp[,-3]
final <- temp1[order(temp1$bh_p),]
rownames(final) <- NULL
final
```