732A99_lab2_block2_A2

Anubhav Dikshit(anudi287), Lennart Schilling(lensc874), Thijs Quast(thiqu264)

17 December 2018

Contents


```
if (!require("pacman")) install.packages("pacman")
pacman::p_load(xlsx, ggplot2, tidyr, dplyr, reshape2, gridExtra,
               mgcv, rgl, akima, pamr, caret, glmnet, kernlab)
```

```r
set.seed(12345)
options("jtools-digits" = 2, scipen = 999)

# colours (colour blind friendly)
cbPalette <- c("#999999", "#E69F00", "#56B4E9", "#009E73", "#F0E442", "#0072B2",
               "#D55E00", "#CC79A7")

## Making title in the center
theme_update(plot.title = element_text(hjust = 0.5))
```

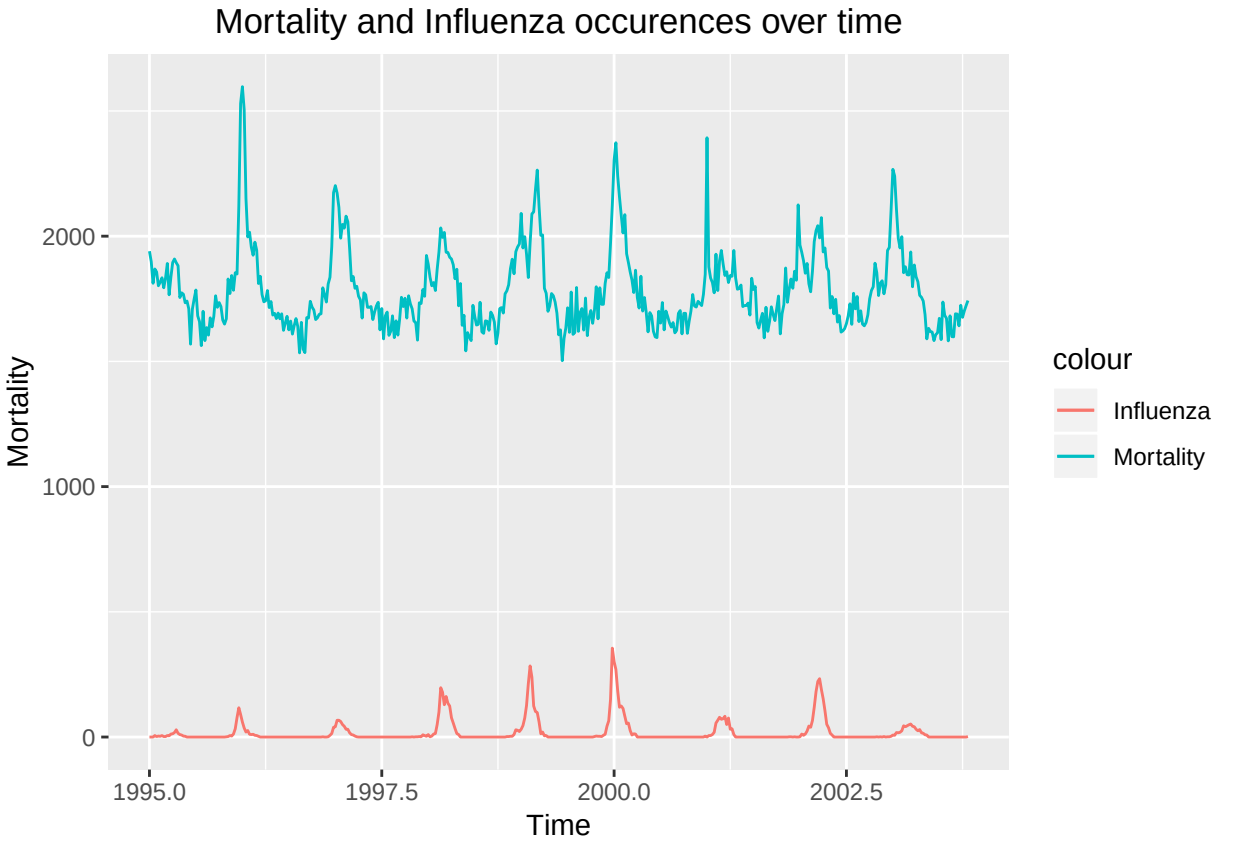
Contributions

During the lab, Thijs focused on assignment 1 while Lennart and Anubhav focused on assignment 2. All codes and analysis was indepedently done and is also reflected in the individual reports.

Assignment 1

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.

```
library(readxl)
options(scipen = 999)
influenza <- read_xlsx("influenza.xlsx")
influenza$Time_fixed <- as.Date(paste(influenza$Year, influenza$Week, 1, sep="-"), "%Y-%U-%u")

library(ggplot2)
plot <- ggplot(data = influenza, aes(x = Time, y = Mortality, color = "Mortality")) +
  geom_line() +
  geom_line(aes(y = Influenza, color = "Influenza")) +
  ggtitle("Mortality and Influenza occurences over time")

plot
```

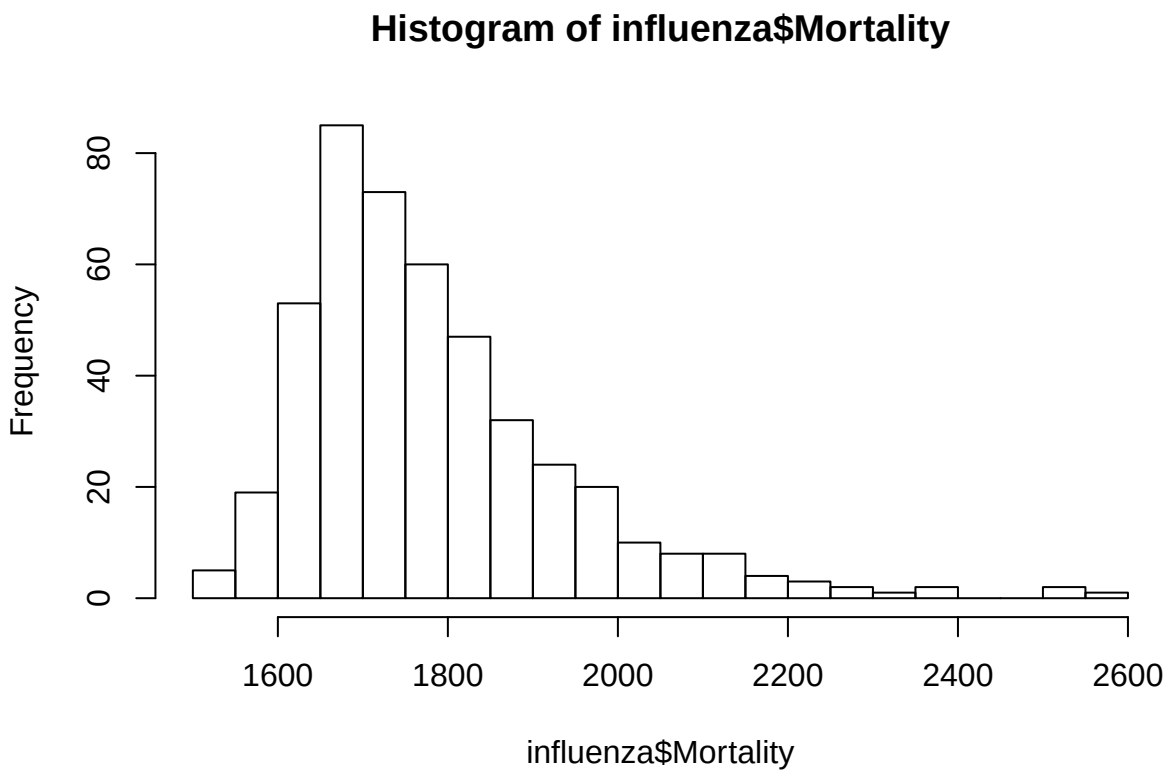


Analysis:

When looking at the plot of Mortality and Influenza cases over time, one can see a similarity in the patterns. When Influenze reaches a spike, so does the Mortality rate. From such a plot one is then tempted to argue that Influenza causes the mortality to rate to go up. Given that Influenza is a disease, I would say it is reasonable to argue that spikes in Influenza cases lead to spikes in the Mortality rate.

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.

```
library(mgcv)
hist(influenza$Mortality, breaks = 20)
```




```
gam <- gam(Mortality ~ s(Week) + Year, data = influenza, method = "GCV.Cp")
summary(gam)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ s(Week) + Year
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -652.060   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  8.951 100.3 <0.0000000000000002 ***
```

```
## ---
## 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
```

Analysis:

Using the default parameter settings within the *gam*-function implies that *Mortality* is normally distributed (*family=gaussian()*). Also, since *method = "GCV.Cp"*, this leads to the usage of GCV (*Generalized Cross Validation score*) related to the smoothing parameter estimation. The underlying probabilistic model can be written as:

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

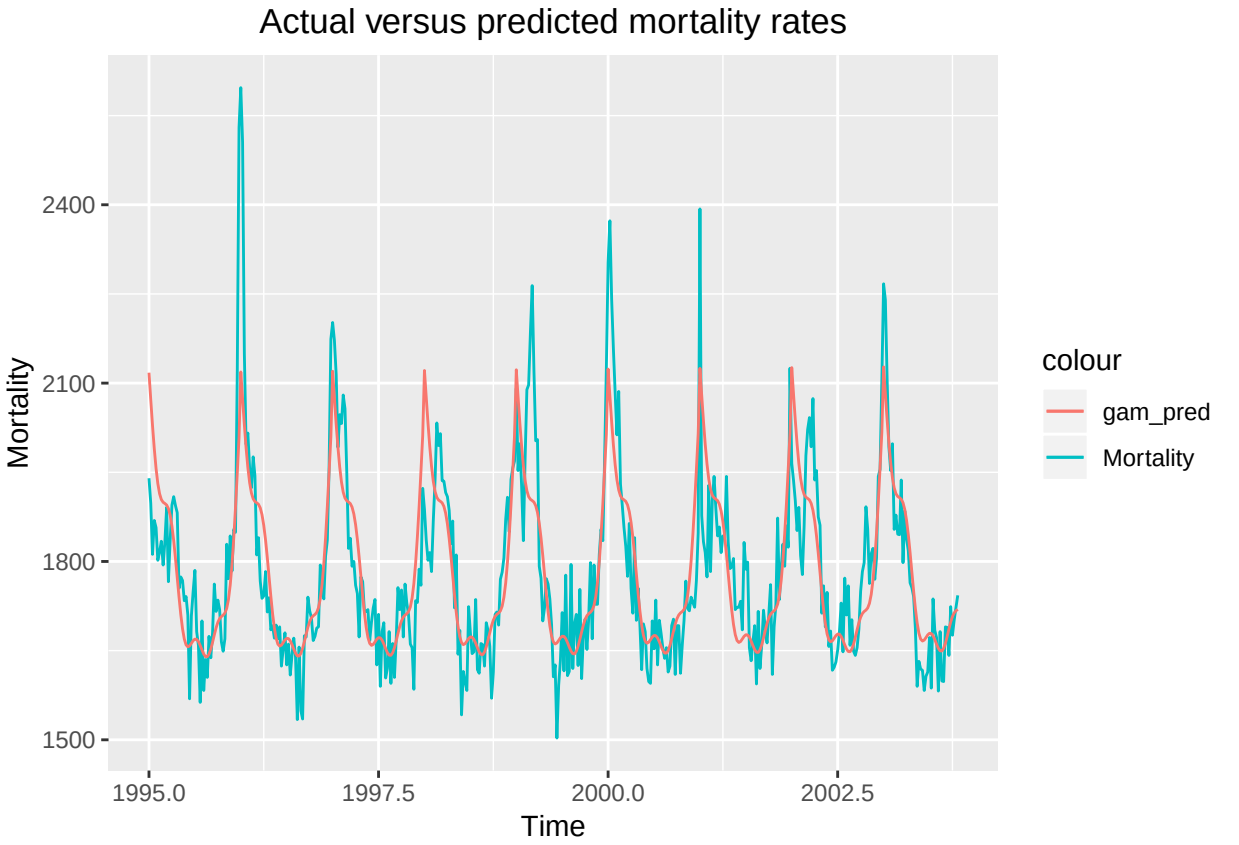
$$\hat{Mortality} = Intercept + \beta_1 Year + s(Week) + \epsilon$$

where

$$\epsilon = N(0, \sigma^2).$$

3. 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.

```r
gam_pred <- predict.gam(gam, newdata = influenza)
influenza <- cbind(influenza, gam_pred)

plot_gam <- ggplot(data = influenza, aes(x = Time, y = Mortality, color = "Mortality")) +
  geom_line() +
  geom_line(aes(y = gam_pred, color = "gam_pred")) +
  ggtitle("Actual versus predicted mortality rates")
plot_gam
```



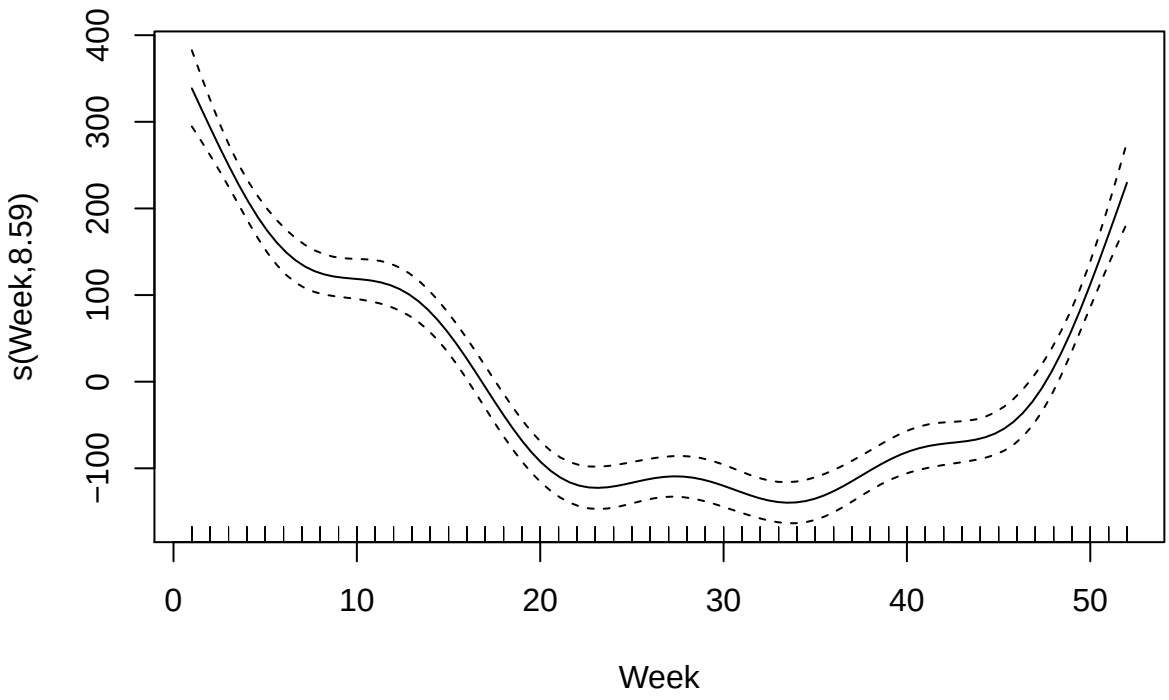
Analysis:

The predicted values for Mortality are shown in the red line, whereas actual values are shown in the blue line. The patterns of both line correspond, meaning the model estimates the dependent variable in a good way. Therefore I would say the fit is good. Still it has to be mentioned that the fitted values do not fully capture the extremes of the actual mortality rate.

Results from step 1.2 imply that the parametric coefficients are insignificantly different from zero, therefore we cannot assume the coefficients have an influence on the target variable. However, the smoothing terms result in a significant p value for the Week variable. Meaning, week has a significant influence on the target variable. Given the adjusted R-squared value, 66.1% of the variance is explained by this model.

The plot above show that Mortality rates peak each year. Therefore I would say there is not trend in mortality rate from one year to another. I would rather say, mortality rates show the same trend within each year, namely a peak at a certain time of the year.

```
plot(gam)
```



Analysis:

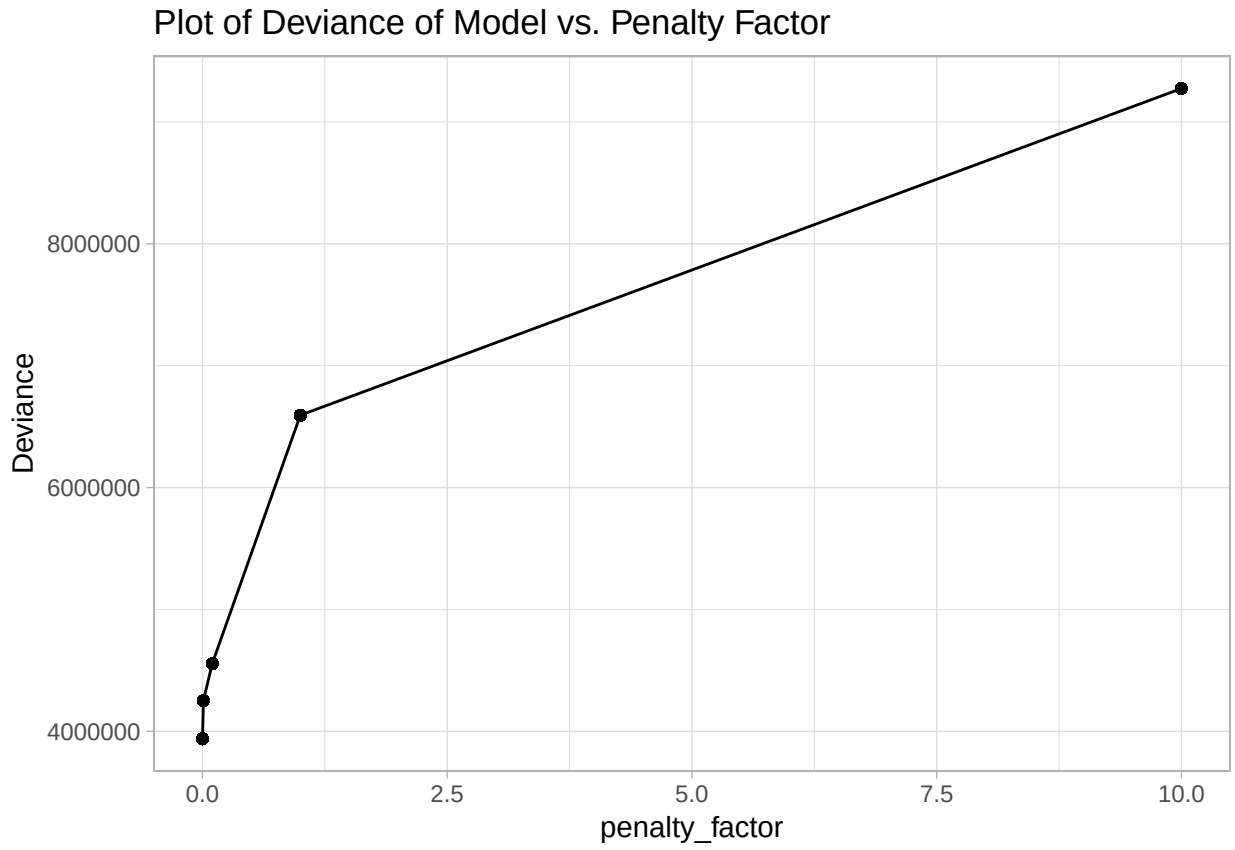
The plot of the spline component shows how the response variable (Mortality) varies with the weeks of the year. Clearly, at the beginning and end of the year mortality rates are very much higher than in the middel of the year. When one thinks of this, this makes sense. Most likely will people suffer from influenzia in winter periods, thus the beginning and end of the calendar year, whereas in summer, the middle of the calendar year, people suffer less from influenzia, and thus less people die.

The curves in the shape is due to the fact that smoothing factors were implented in the model, and is due to non-linearity in the data. Dotted lines around the line represent standard errors of the fit.

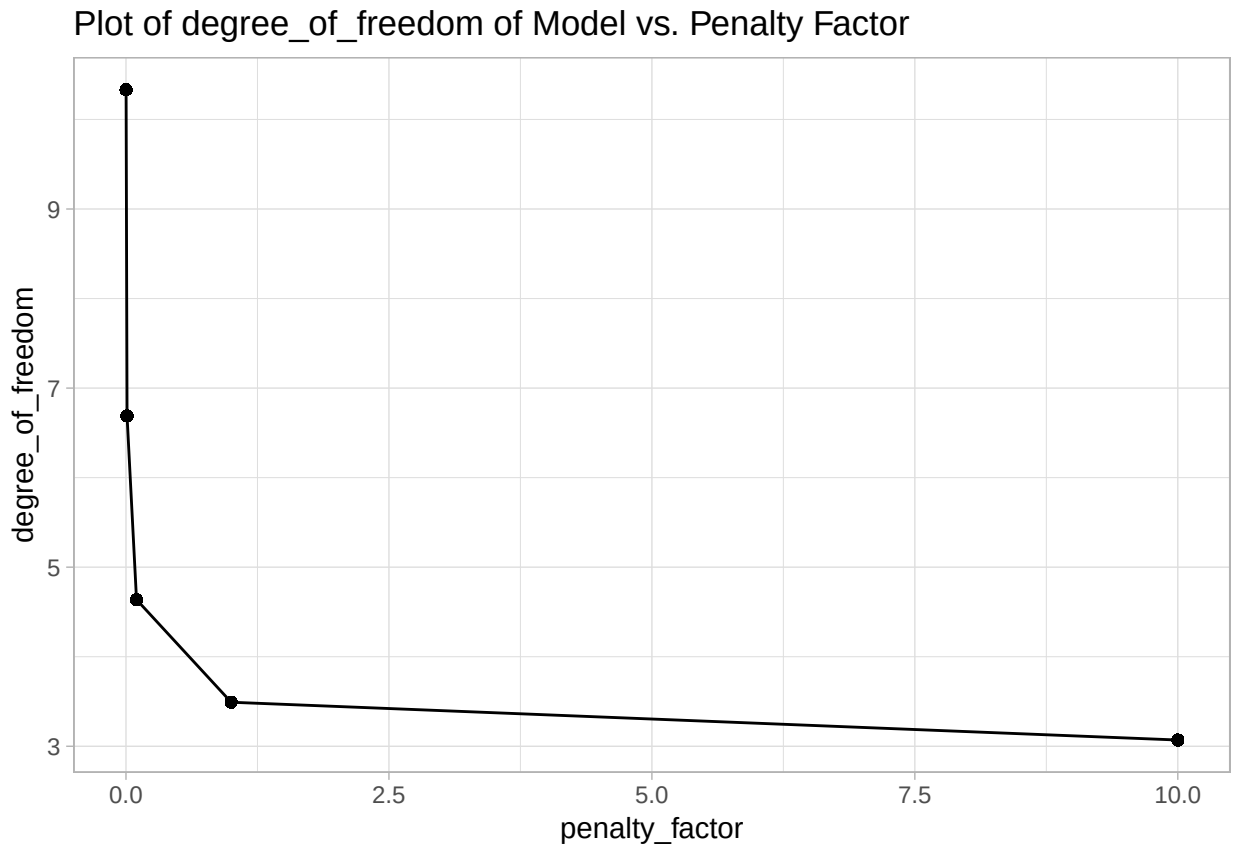
4. Examine how the penalty factor of the spline function in the GAM model from step 2 influences the estimated deviance of the model. Make plots of the predicted and observed mortality against time for cases of very high and very low penalty factors. What is the relation of the penalty factor to the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

```
model_deviance <- NULL
for(sp in c(0.001, 0.01, 0.1, 1, 10))
{
  k=length(unique(influenza$Week))

gam_model <- mgcv::gam(data = influenza, Mortality~Year+s(Week, k=k, sp=sp), method = "GCV.Cp")
temp <- cbind(gam_model$deviance, gam_model$fitted.values, gam_model$y, influenza$Time_fixed,
              sp, sum(influence(gam_model)))
```

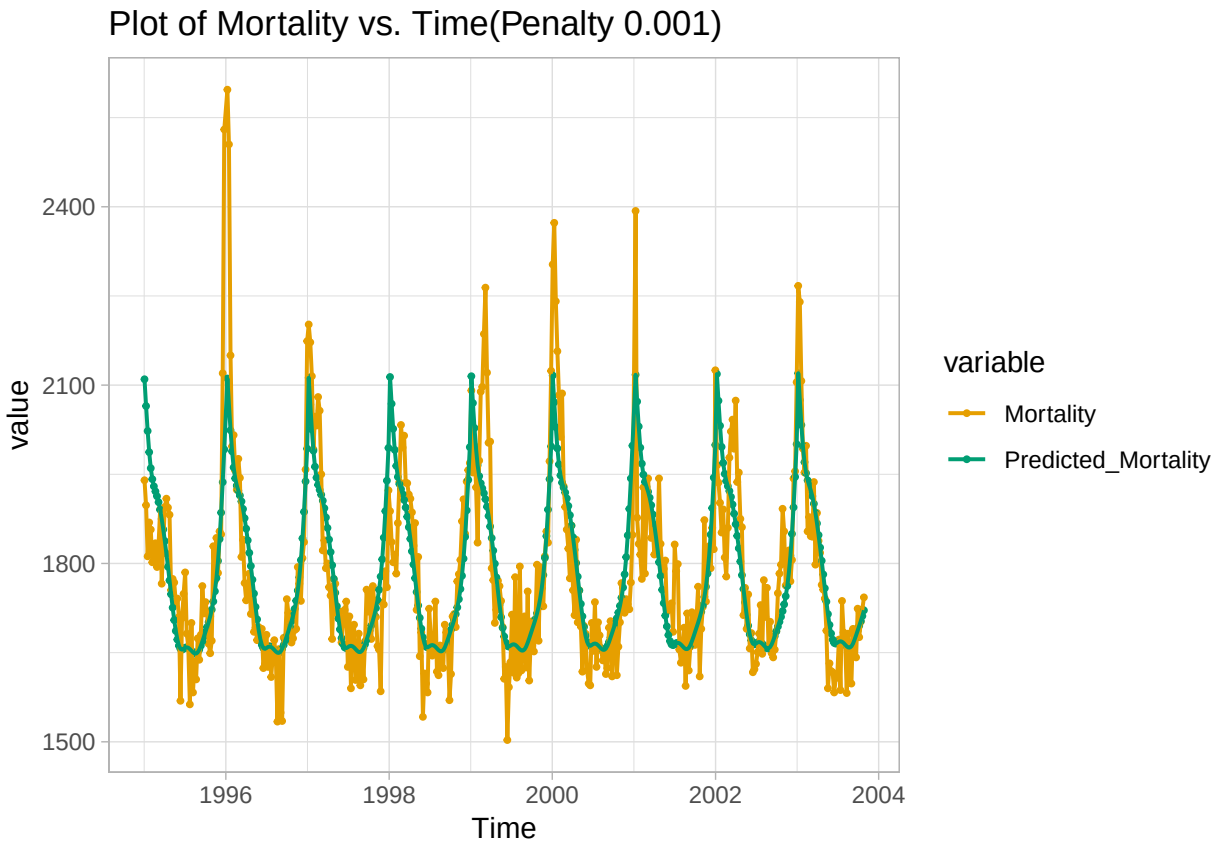
```
model_deviance <- rbind(temp, model_deviance)
}
model_deviance <- as.data.frame(model_deviance)
colnames(model_deviance) <- c("Deviance", "Predicted_Mortality", "Mortality", "Time",
                              "penalty_factor", "degree_of_freedom")
model_deviance$Time <- as.Date(model_deviance$Time, origin = '1970-01-01')


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



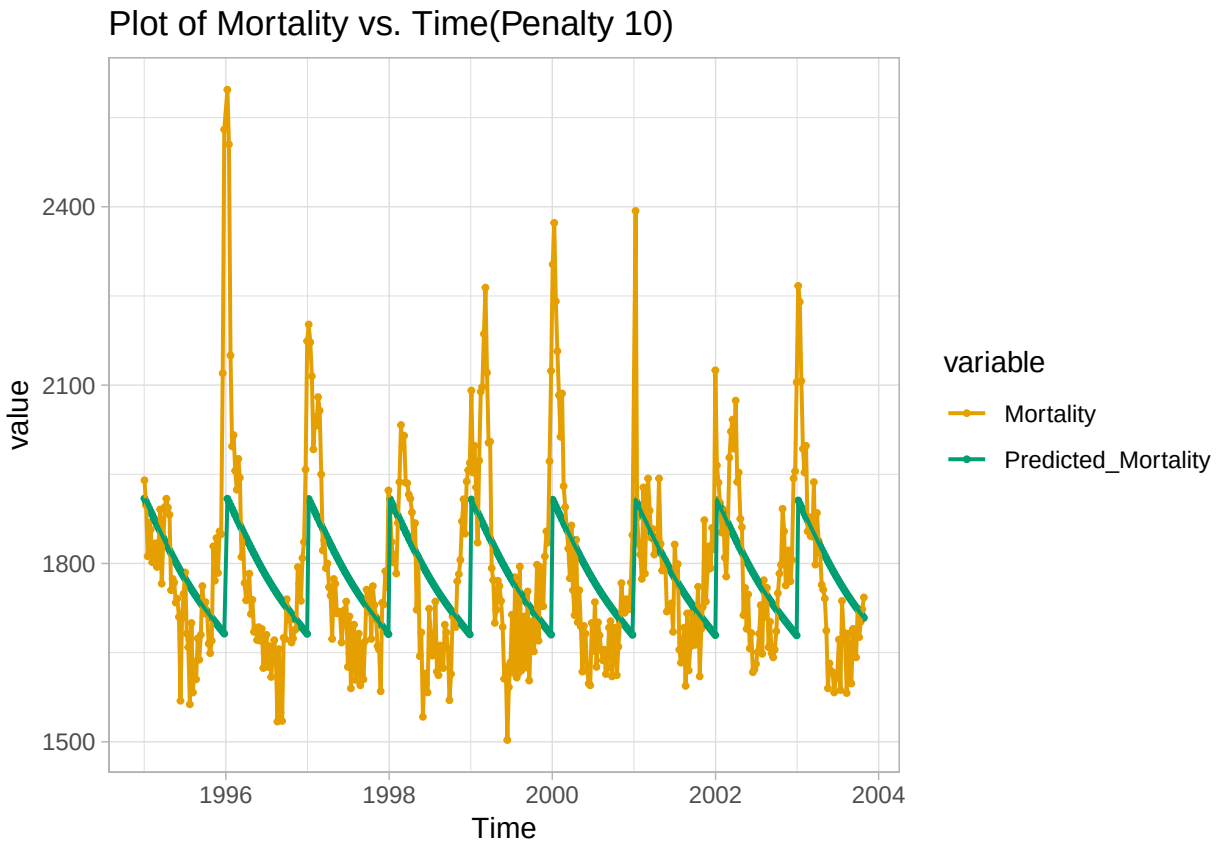

```
# plot of degree of freedom
p7 <- ggplot(data=model_deviance, aes(x = penalty_factor, y = degree_of_freedom)) +
geom_point() +
  geom_line() +
      theme_light() +
ggtitle("Plot of degree_of_freedom of Model vs. Penalty Factor")
p7
```




```
model_deviance_wide <- melt(model_deviance[,c("Time", "penalty_factor",
                                              "Mortality", "Predicted_Mortality")],
                            id.vars = c("Time", "penalty_factor"))

# plot of predicted vs. observed mortality
p8 <- ggplot(data=model_deviance_wide[model_deviance_wide$penalty_factor == 0.001,],
             aes(x= Time, y = value)) +
  geom_point(aes(color = variable), size=0.7) +
  geom_line(aes(color = variable), size=0.7) +
  scale_color_manual(values=c("#E69F00", "#009E73")) +
  theme_light() +
  ggtitle("Plot of Mortality vs. Time(Penalty 0.001)")

p9 <- ggplot(data=model_deviance_wide[model_deviance_wide$penalty_factor == 10,],
             aes(x= Time, y = value)) +
  geom_point(aes(color = variable), size=0.7) +
    geom_line(aes(color = variable), size=0.7) +
  scale_color_manual(values=c("#E69F00", "#009E73")) +
    theme_light() +
  ggtitle("Plot of Mortality vs. Time(Penalty 10)")

p8
```


Plot of Mortality vs. Time(Penalty 0.001)
value
2400
2100
1800
1500
1996
1998
2000
2002
2004
Time
variable
Mortality
Predicted_Mortality

```
p9
```



Analysis:

A gamma model with a small penalty factor results in more degrees of freedom and higher percentage of deviance explained than the gamma model with a high penalty factor. Therefore the penalty factor negatively relates to deviance and degrees of freedom. The fact that this relationship holds can be seen from the plot above, in which a penalty factor of 10 shows a severly worse fit to the data.

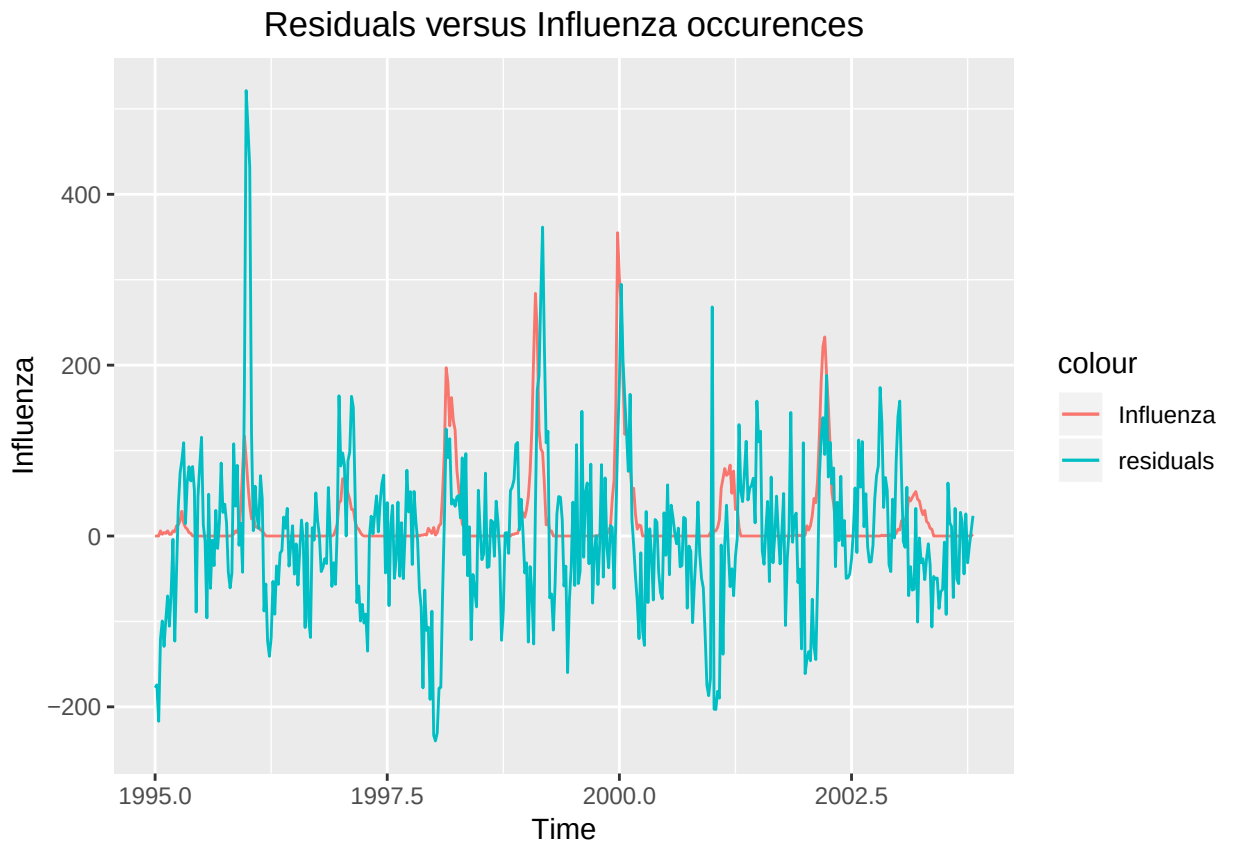
Another explaination is that penalty factor in the model determines the complexity of the model, higher the penalty factor the more the model will have bias and hence lesser the complexity. We can see that as the penalty factor increases the degree of freedom decreases.

From the plots of degree of freedom vs. penalty factor we see that our result to confirm our hypothesis.

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?

```
residuals <- influenza$Mortality - gam_pred
df2 <- data.frame(cbind(influenza$Time, influenza$Influenza, residuals))
colnames(df2) <- c("Time", "Influenza", "residuals")

residuals_plot <- ggplot(data = df2, aes(x = Time, y = Influenza, color = "Influenza")) +
  geom_line() +
  geom_line(aes(y = residuals, color = "residuals")) +
  ggtitle("Residuals versus Influenza occurences")
```

```
residuals_plot
```



Analysis:

Some of the peaks in Influenza outbreaks correspond to peaks in the residuals of the fitted model. Still, however, a lot of variance in the residuals is not correlated to Influenza outbreaks. Therefore, I would say that the Influenza outbreaks are not correlated to the residuals.

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.

```
additive_gam <- gam(Mortality ~ s(Year, k=length(unique(influenza$Year))) +
                      s(Week, k=length(unique(influenza$Week))) +
                      s(Influenza, k=length(unique(influenza$Influenza))), data = influenza)


summary(additive_gam)
```

```
##
## Family: gaussian
## Link function: identity
```

```
##
## Formula:
## Mortality ~ s(Year, k = length(unique(influenza$Year))) + s(Week,
##     k = length(unique(influenza$Week))) + s(Influenza, k = length(unique(influenza$Influenza)))
##
## Parametric coefficients:
##             Estimate Std. Error t value            Pr(>|t|)
## (Intercept)   1783.8        3.2   557.5 <0.0000000000000002 ***
## ---
## 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)       4.663  5.677  1.487               0.181
## s(Week)      14.641 18.248 18.533 <0.0000000000000002 ***
## s(Influenza) 69.737 72.854  5.599 <0.0000000000000002 ***
## ---
## 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
```

```r
additive_pred <- predict.gam(additive_gam, newdata = influenza)


influenza <- cbind(influenza, additive_pred)
plot_additive <- ggplot(data = influenza, aes(x = Time, y = Mortality, color = "Mortality")) +
  geom_line() +
  geom_line(aes(y = additive_pred, color = "additive_pred")) +
  ggtitle("Predicted mortality rate versus actual mortality rate over time")
plot_additive
```

Analysis:

The additive GAM model clearly has the best fit. Much of the variance of the data is captured by the model, given the R-squared statistic of 0.819. Given that the GAM models in step 2 and step 4 do not include the influenza variable from the dataset, and the the model above does, one can say that most likely mortality is influenced by the outbreaks of influenza.

Assignment 2

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.

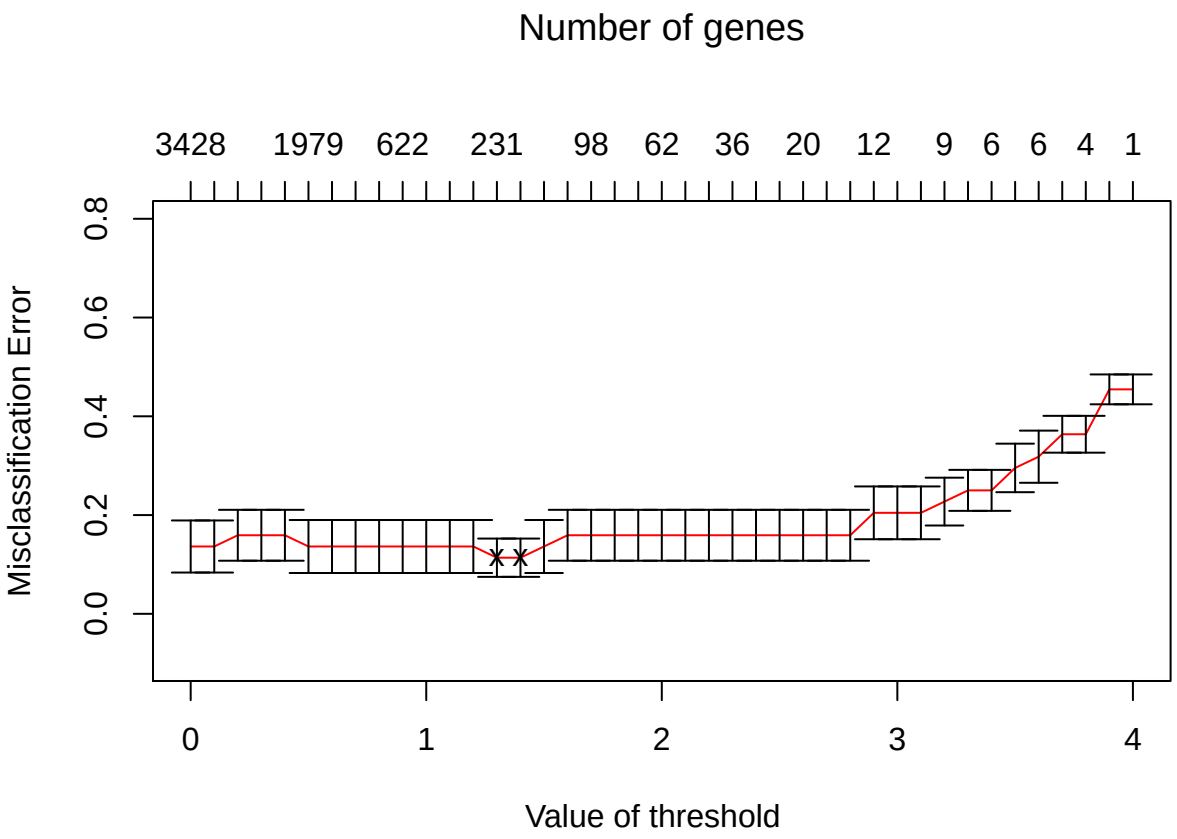
```
rm(list=ls())
gc()
data <- read.csv(file = "data.csv", sep = ";", header = TRUE)
```

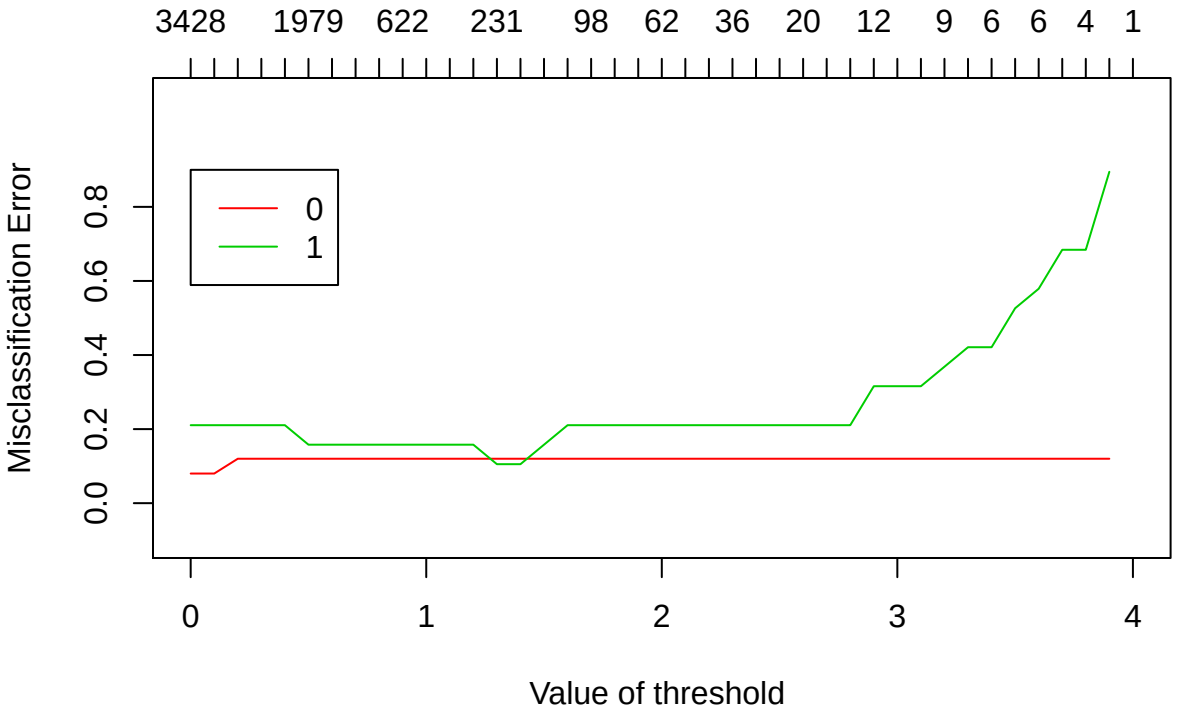
```
n=NROW(data)
data$Conference <- as.factor(data$Conference)
set.seed(12345)
```

```
id=sample(1:n, floor(n*0.7))
train=data[id,]
test = data[-id,]

rownames(train)=1:nrow(train)
x=t(train[,-4703])
y=train[[4703]]

rownames(test)=1:nrow(test)
x_test=t(test[,-4703])
y_test=test[[4703]]

mydata = list(x=x,y=as.factor(y),geneid=as.character(1:nrow(x)), genenames=rownames(x))
mydata_test = list(x=x_test,y=as.factor(y_test),geneid=as.character(1:nrow(x)),
                   genenames=rownames(x))
model = pamr.train(mydata,threshold=seq(0, 4, 0.1))

cvmodel=pamr.cv(model, mydata)
important_gen <- as.data.frame(pamr.listgenes(model, mydata, threshold = 1.3))
predicted_scc_test <- pamr.predict(model, newx = x_test, threshold = 1.3)
```

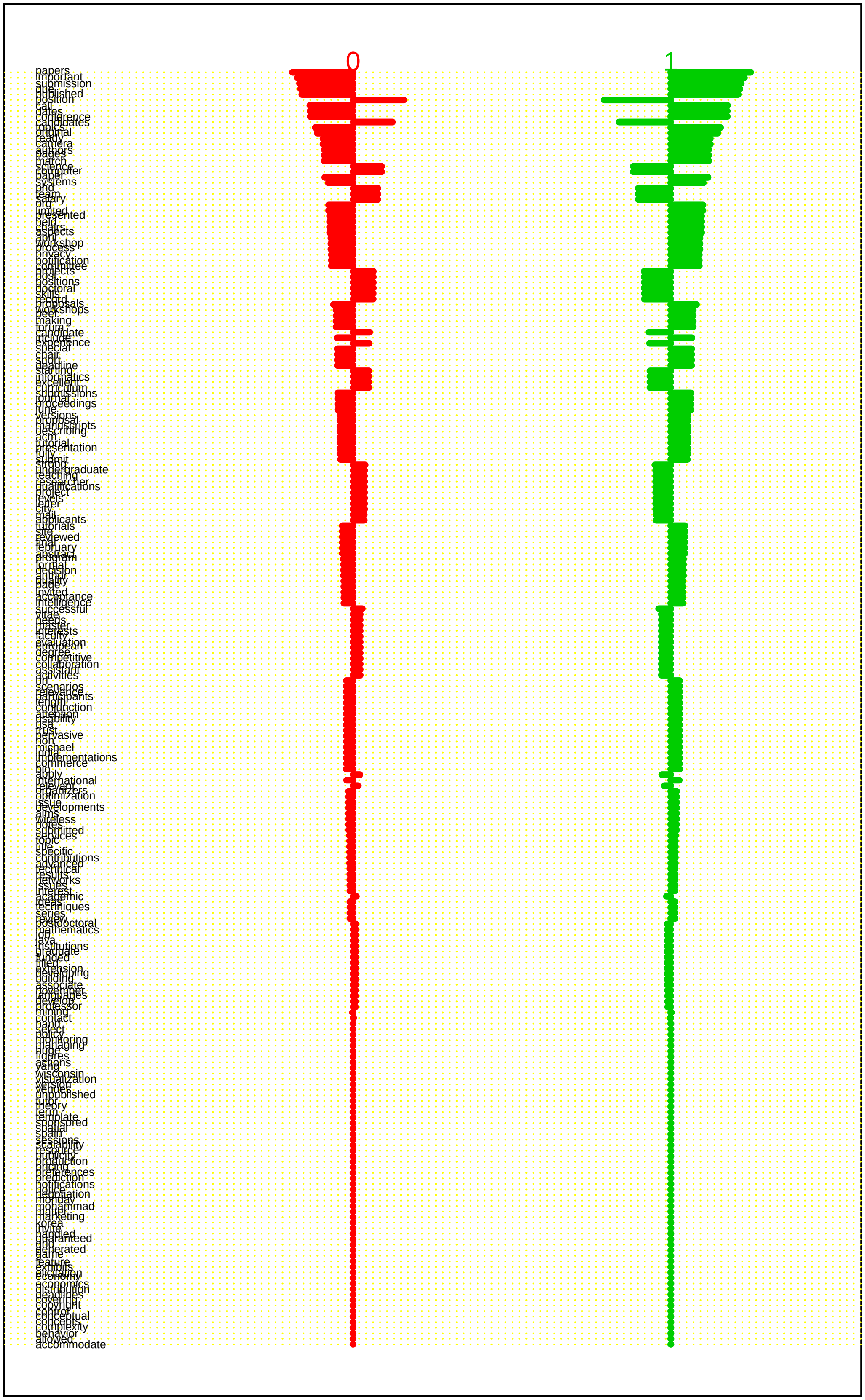
plots

```
pamr.plotcv(cvmodel)
```


Number of genes
3428 1979 622 231 98 62 36 20 12 9 6 6 4 1
Misclassification Error
0.0 0.2 0.4 0.6 0.8
0 1 2 3 4
Value of threshold


3428 1979 622 231 98 62 36 20 12 9 6 6 4 1
Misclassification Error
0.0 0.2 0.4 0.6 0.8
0
1
0 1 2 3 4
Value of threshold

```
pamr.plotcen(model, mydata, threshold = 1.3)
```


0
1

important features

```
## List the significant genes
temp <- colnames(data) %>% as.data.frame()
colnames(temp) <- "col_name"
temp$index <- row.names(temp)

df <- merge(x = important_gen, y = temp, by.x = "id", by.y = "index", all.x = TRUE)
df <- df[order(df[,3], decreasing = TRUE ),]

knitr::kable(head(df[,4],10),
             caption = "Important feaures selected by Nearest Shrunken Centroids ")
```

Table 1: Important feaures selected by Nearest Shrunken Centroids

x
papers
important
submission
due
published
call
dates
conference
topics
original

confusion table

```
conf_scc <- table(y_test, predicted_scc_test)
names(dimnames(conf_scc)) <- c("Actual Test", "Predicted Srunken Centroid Test")
result_scc <- caret::confusionMatrix(conf_scc)
caret::confusionMatrix(conf_scc)
```

```
## Confusion Matrix and Statistics
##
##            Predicted Srunken Centroid Test
## Actual Test  0  1
##           0 10  0
##           1  2  8
##
##                Accuracy : 0.9
##                  95% CI : (0.683, 0.9877)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.003611
##
##                   Kappa : 0.8
##  Mcnemar's Test P-Value : 0.479500
##
##             Sensitivity : 0.8333
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.8000
```

```
##              Prevalence : 0.6000
##          Detection Rate : 0.5000
##    Detection Prevalence : 0.5000
##       Balanced Accuracy : 0.9167
##
##        'Positive' Class : 0
##
```

Analysis:

From the plot of thershold vs. misclassification error we can see that for the thershold value of 1.3, the class error is lowest.

231 features were selected by this model as the most important features. The top ten features of the model are given by the table above, from this table we can see that the features selected are logical in nature, example "conference", "papers" etc.

The test error is just 10% (accuracy is 90%) and the ability of our model to classify non-conference is 100%, while its ability to classifiy conference mail is 80%, the accuracy along with low number of samples hints that our model may very well be overfitted.

2. Compute the test error and the number of the contributing features for the following methods fitted to the training data: a. Elastic net with the binomial response and alpha = 0.5 in which penalty is selected by the cross-validation. b. Support vector machine with "vanilladot" kernel. 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?

```
x = train[,-4703] %>% as.matrix()
y = train[,4703]

x_test = test[,-4703] %>% as.matrix()
y_test = test[,4703]

cvfit = cv.glmnet(x=x, y=y, alpha = 0.5, family =   "binomial")
predicted_elastic_test <- predict.cv.glmnet(cvfit, newx = x_test, s = "lambda.min", type = "class")
tmp_coeffs <- coef(cvfit, s = "lambda.min")
elastic_variable <- data.frame(name = tmp_coeffs@Dimnames[[1]][tmp_coeffs@i + 1],
                               coefficient = tmp_coeffs@x)
knitr::kable(elastic_variable, caption = "Contributing features in the elastic model")
```

Table 2: Contributing features in the elastic model

name	coefficient
(Intercept)	-1.0189313
abstracts	-0.3011264
aspects	0.0736776
bio	0.0228765
call	0.3319900
candidates	-0.1878311
computer	-0.2832065
conceptual	0.0380844
conference	0.1965330

name	coefficient
dates	0.2416630
due	0.5211725
evaluation	-0.1796401
exhibits	0.3782700
important	0.3924275
languages	-0.0258470
making	0.1892394
manuscripts	0.0325584
original	0.0558205
papers	0.3853810
peer	0.0967211
position	-0.3750830
process	0.0016238
projects	-0.1904080
proposals	0.0553554
published	0.2818206
queries	-0.3002459
record	-0.1162514
relevant	-0.1135564
scenarios	0.0053470
spatial	0.1925007
submission	0.2803519
team	-0.1291278
versions	0.1545749

```
conf_elastic_net <- table(y_test, predicted_elastic_test)
names(dimnames(conf_elastic_net)) <- c("Actual Test", "Predicted ElasticNet Test")
result_elastic_net <- caret::confusionMatrix(conf_elastic_net)
caret::confusionMatrix(conf_elastic_net)
```

```
## Confusion Matrix and Statistics
##
##            Predicted ElasticNet Test
## Actual Test  0  1
##           0 10  0
##           1  2  8
##
##                Accuracy : 0.9
##                  95% CI : (0.683, 0.9877)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.003611
##
##                   Kappa : 0.8
##  Mcnemar's Test P-Value : 0.479500
##
##             Sensitivity : 0.8333
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.8000
##              Prevalence : 0.6000
##          Detection Rate : 0.5000
##    Detection Prevalence : 0.5000
```

```
##       Balanced Accuracy : 0.9167
##
##        'Positive' Class : 0
##
```

```
# svm
svm_fit <- kernlab::ksvm(x, y, kernel="vanilladot", scale = FALSE, type = "C-svc")
```

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

```
predicted_svm_test <- predict(svm_fit, x_test, type="response")


conf_svm_tree <- table(y_test, predicted_svm_test)
names(dimnames(conf_svm_tree)) <- c("Actual Test", "Predicted SVM Test")
result_svm <- caret::confusionMatrix(conf_svm_tree)
caret::confusionMatrix(conf_svm_tree)
```

```
## Confusion Matrix and Statistics
##
##            Predicted SVM Test
## Actual Test  0  1
##           0 10  0
##           1  1  9
##
##                Accuracy : 0.95
##                  95% CI : (0.7513, 0.9987)
##     No Information Rate : 0.55
##     P-Value [Acc > NIR] : 0.0001114
##
##                   Kappa : 0.9
##  Mcnemar's Test P-Value : 1.0000000
##
##             Sensitivity : 0.9091
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9000
##              Prevalence : 0.5500
##          Detection Rate : 0.5000
##    Detection Prevalence : 0.5000
##       Balanced Accuracy : 0.9545
##
##        'Positive' Class : 0
##
```

```
# creating table
final_result <- cbind(result_scc$overall[[1]]*100,
                      result_elastic_net$overall[[1]]*100,
                      result_svm$overall[[1]] *100) %>% as.data.frame()

features_count <- cbind(NROW(important_gen), NROW(elastic_variable), NCOL(data))

final_result <- rbind(final_result, features_count)

colnames(final_result) <- c("Nearest Shrunken Centroid Model",
                            "ElasticNet Model", "SVM Model")
```

```r
rownames(final_result) <- c("Accuracy", "Number of Features")

knitr::kable(final_result, caption = "Comparsion of Models on Test dataset")
```

Table 3: Comparsion of Models on Test dataset

	Nearest Shrunken Centroid Model	ElasticNet Model	SVM Model
Accuracy	90	90	95
Number of Features	231	33	4703

Analysis:

33 variables were selected by the elastic net model as the features for classifying the mails as conference, while the svm model selects 4703 features to classify the mails.

From the model comparsion we see that overall choosing SVM gives us the best accuracy, while Nearest Centroid Model and Elastic Net model both have the same accuracy, however this is not a strong point given the low number of samples. From the coefficents of the elastic net we can see that the features choosen from the elastic net are far more reasonable than the once choosen by Nearest Centroid model, thus Elastic Net features selection is superior to Nearest Centroid model in quality and quantity too.

For SVM even though the model has good accuracy the sheer number of features used makes choosing this hard, although it should be noted that choosing SVM works very well when we are dealing with a sparse dataset.

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.

```r
p_value <- c()
for (i in 1:4702){
  x <- data[,i]
  res <- t.test(x ~ Conference, data = data, alternative = "two.sided")
  p <- res$p.value
  p_value[i] <- p
}
p_value <- as.data.frame(p_value)
p_value$reject_flag <- as.factor(ifelse(p_value$p_value <0.05, "Retain", "Drop"))
p_value$column_index <- row.names(p_value)

keep <- ifelse(p_value$reject_flag == "Retain", as.numeric(p_value$column_index), NA)
keep <- na.omit(keep)
keep <- colnames(data[,keep])
keep
```

```
##   [1] "abstract"       "academic"       "acceptance"     "accepted"       "access"         "acm
##  [28] "bio"            "call"           "calls"          "camera"         "canada"         "can
##  [55] "contributions"  "copyright"      "covering"       "cross"          "curriculum"     "dat
##  [82] "expected"       "experience"     "extension"      "feature"        "february"       "fig
## [109] "include"        "included"       "india"          "infrastructures" "initially"     "ins
## [136] "letter"         "levels"         "limited"        "liu"            "looking"        "mad
## [163] "ontologies"     "opportunity"    "optimization"   "org"            "organizers"     "org
```

```
## [190] "privacy"         "proceedings"     "process"         "professor"       "proficiency"     "prog
## [217] "scalability"     "scenarios"       "science"         "scope"           "security"        "seri
## [244] "taiwan"          "takes"           "tasks"           "teaching"        "team"            "tech
## [271] "versions"        "vienna"          "visualization"   "vitae"           "wang"            "wire
```

Analysis:

From the above table we can see that 281 features had significant p-values (more than 0.05), some of the features do make sense to in their ability to distinguish mails pertaining to conferences, such as 'committee', 'conference', 'international', 'keynote', 'manuscripts' etc.

Thus we see that even a simple and time tested techniques like t test can be used to get a sense of the important features for model building. Although this method does help us its still selects far too many features than the other methods that we have seen and implemented uptill now.

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
if (!require("pacman")) install.packages("pacman")
pacman::p_load(xlsx, ggplot2, tidyr, dplyr, reshape2, gridExtra,
               mgcv, rgl, akima, pamr, caret, glmnet, kernlab)

set.seed(12345)
options("jtools-digits" = 2, scipen = 999)

# colours (colour blind friendly)
cbPalette <- c("#999999", "#E69F00", "#56B4E9", "#009E73", "#F0E442", "#0072B2",
               "#D55E00", "#CC79A7")

## Making title in the center
theme_update(plot.title = element_text(hjust = 0.5))
library(readxl)
options(scipen = 999)
influenza <- read_xlsx("influenza.xlsx")
influenza$Time_fixed <- as.Date(paste(influenza$Year, influenza$Week, 1, sep="-"), "%Y-%U-%u")

library(ggplot2)
plot <- ggplot(data = influenza, aes(x = Time, y = Mortality, color = "Mortality")) +
  geom_line() +
  geom_line(aes(y = Influenza, color = "Influenza")) +
  ggtitle("Mortality and Influenza occurences over time")

plot
library(mgcv)
hist(influenza$Mortality, breaks = 20)
gam <- gam(Mortality ~ s(Week) + Year, data = influenza, method = "GCV.Cp")
summary(gam)
gam_pred <- predict.gam(gam, newdata = influenza)
influenza <- cbind(influenza, gam_pred)

plot_gam <- ggplot(data = influenza, aes(x = Time, y = Mortality, color = "Mortality")) +
  geom_line() +
  geom_line(aes(y = gam_pred, color = "gam_pred")) +
```

```
  ggtitle("Actual versus predicted mortality rates")
plot_gam
plot(gam)


model_deviance <- NULL
for(sp in c(0.001, 0.01, 0.1, 1, 10))
{
  k=length(unique(influenza$Week))

gam_model <- mgcv::gam(data = influenza, Mortality~Year+s(Week, k=k, sp=sp), method = "GCV.Cp")
temp <- cbind(gam_model$deviance, gam_model$fitted.values, gam_model$y, influenza$Time_fixed,
              sp, sum(influence(gam_model)))

model_deviance <- rbind(temp, model_deviance)
}
model_deviance <- as.data.frame(model_deviance)
colnames(model_deviance) <- c("Deviance", "Predicted_Mortality", "Mortality", "Time",
                              "penalty_factor", "degree_of_freedom")
model_deviance$Time <- as.Date(model_deviance$Time, origin = '1970-01-01')


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

# plot of degree of freedom
p7 <- ggplot(data=model_deviance, aes(x = penalty_factor, y = degree_of_freedom)) +
geom_point() +
  geom_line() +
      theme_light() +
ggtitle("Plot of degree_of_freedom of Model vs. Penalty Factor")
p7

model_deviance_wide <- melt(model_deviance[,c("Time", "penalty_factor",
                                              "Mortality", "Predicted_Mortality")],
                            id.vars = c("Time", "penalty_factor"))

# plot of predicted vs. observed mortality
p8 <- ggplot(data=model_deviance_wide[model_deviance_wide$penalty_factor == 0.001,],
             aes(x= Time, y = value)) +
  geom_point(aes(color = variable), size=0.7) +
  geom_line(aes(color = variable), size=0.7) +
  scale_color_manual(values=c("#E69F00", "#009E73")) +
  theme_light() +
  ggtitle("Plot of Mortality vs. Time(Penalty 0.001)")

p9 <- ggplot(data=model_deviance_wide[model_deviance_wide$penalty_factor == 10,],
             aes(x= Time, y = value)) +
```

```
  geom_point(aes(color = variable), size=0.7) +
    geom_line(aes(color = variable), size=0.7) +
  scale_color_manual(values=c("#E69F00", "#009E73")) +
    theme_light() +
  ggtitle("Plot of Mortality vs. Time(Penalty 10)")

p8
p9
residuals <- influenza$Mortality - gam_pred
df2 <- data.frame(cbind(influenza$Time, influenza$Influenza, residuals))
colnames(df2) <- c("Time", "Influenza", "residuals")

residuals_plot <- ggplot(data = df2, aes(x = Time, y = Influenza, color = "Influenza")) +
  geom_line() +
  geom_line(aes(y = residuals, color = "residuals")) +
  ggtitle("Residuals versus Influenza occurences")

residuals_plot
additive_gam <- gam(Mortality ~ s(Year, k=length(unique(influenza$Year))) +
                      s(Week, k=length(unique(influenza$Week))) +
                      s(Influenza, k=length(unique(influenza$Influenza))), data = influenza)


summary(additive_gam)

additive_pred <- predict.gam(additive_gam, newdata = influenza)


influenza <- cbind(influenza, additive_pred)
plot_additive <- ggplot(data = influenza, aes(x = Time, y = Mortality, color = "Mortality")) +
  geom_line() +
  geom_line(aes(y = additive_pred, color = "additive_pred")) +
  ggtitle("Predicted mortality rate versus actual mortality rate over time")
plot_additive
rm(list=ls())
gc()
data <- read.csv(file = "data.csv", sep = ";", header = TRUE)
n=NROW(data)
data$Conference <- as.factor(data$Conference)
set.seed(12345)
id=sample(1:n, floor(n*0.7))
train=data[id,]
test = data[-id,]

rownames(train)=1:nrow(train)
x=t(train[,-4703])
y=train[[4703]]

rownames(test)=1:nrow(test)
x_test=t(test[,-4703])
y_test=test[[4703]]

mydata = list(x=x,y=as.factor(y),geneid=as.character(1:nrow(x)), genenames=rownames(x))
```

```
mydata_test = list(x=x_test,y=as.factor(y_test),geneid=as.character(1:nrow(x)),
                   genenames=rownames(x))
model = pamr.train(mydata,threshold=seq(0, 4, 0.1))

cvmodel=pamr.cv(model, mydata)
important_gen <- as.data.frame(pamr.listgenes(model, mydata, threshold = 1.3))
predicted_scc_test <- pamr.predict(model, newx = x_test, threshold = 1.3)
pamr.plotcv(cvmodel)
pamr.plotcen(model, mydata, threshold = 1.3)
## List the significant genes
temp <- colnames(data) %>% as.data.frame()
colnames(temp) <- "col_name"
temp$index <- row.names(temp)

df <- merge(x = important_gen, y = temp, by.x = "id", by.y = "index", all.x = TRUE)
df <- df[order(df[,3], decreasing = TRUE ),]

knitr::kable(head(df[,4],10),
             caption = "Important feaures selected by Nearest Shrunken Centroids ")

conf_scc <- table(y_test, predicted_scc_test)
names(dimnames(conf_scc)) <- c("Actual Test", "Predicted Srunken Centroid Test")
result_scc <- caret::confusionMatrix(conf_scc)
caret::confusionMatrix(conf_scc)

x = train[,-4703] %>% as.matrix()
y = train[,4703]

x_test = test[,-4703] %>% as.matrix()
y_test = test[,4703]

cvfit = cv.glmnet(x=x, y=y, alpha = 0.5, family =   "binomial")
predicted_elastic_test <- predict.cv.glmnet(cvfit, newx = x_test, s = "lambda.min", type = "class")
tmp_coeffs <- coef(cvfit, s = "lambda.min")
elastic_variable <- data.frame(name = tmp_coeffs@Dimnames[[1]][tmp_coeffs@i + 1],
                               coefficient = tmp_coeffs@x)
knitr::kable(elastic_variable, caption = "Contributing features in the elastic model")

conf_elastic_net <- table(y_test, predicted_elastic_test)
names(dimnames(conf_elastic_net)) <- c("Actual Test", "Predicted ElasticNet Test")
result_elastic_net <- caret::confusionMatrix(conf_elastic_net)
caret::confusionMatrix(conf_elastic_net)

# svm
svm_fit <- kernlab::ksvm(x, y, kernel="vanilladot", scale = FALSE, type = "C-svc")
predicted_svm_test <- predict(svm_fit, x_test, type="response")


conf_svm_tree <- table(y_test, predicted_svm_test)
names(dimnames(conf_svm_tree)) <- c("Actual Test", "Predicted SVM Test")
result_svm <- caret::confusionMatrix(conf_svm_tree)
caret::confusionMatrix(conf_svm_tree)
```

```r
# creating table
final_result <- cbind(result_scc$overall[[1]]*100,
                      result_elastic_net$overall[[1]]*100,
                      result_svm$overall[[1]] *100) %>% as.data.frame()

features_count <- cbind(NROW(important_gen), NROW(elastic_variable), NCOL(data))

final_result <- rbind(final_result, features_count)

colnames(final_result) <- c("Nearest Shrunken Centroid Model",
                            "ElasticNet Model", "SVM Model")

rownames(final_result) <- c("Accuracy", "Number of Features")

knitr::kable(final_result, caption = "Comparsion of Models on Test dataset")

p_value <- c()
for (i in 1:4702){
  x <- data[,i]
  res <- t.test(x ~ Conference, data = data, alternative = "two.sided")
  p <- res$p.value
  p_value[i] <- p
}
p_value <- as.data.frame(p_value)
p_value$reject_flag <- as.factor(ifelse(p_value$p_value <0.05, "Retain", "Drop"))
p_value$column_index <- row.names(p_value)

keep <- ifelse(p_value$reject_flag == "Retain", as.numeric(p_value$column_index), NA)
keep <- na.omit(keep)
keep <- colnames(data[,keep])
keep
```