machine learning(732A99) lab2 block2

Anubhav Dikshit(anudi287)

17 December 2018

Contents

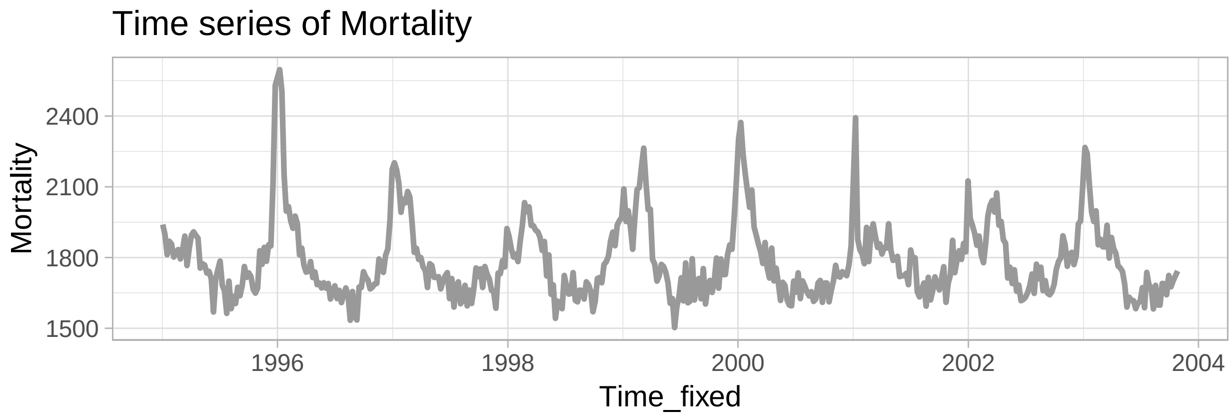
Assignment 1

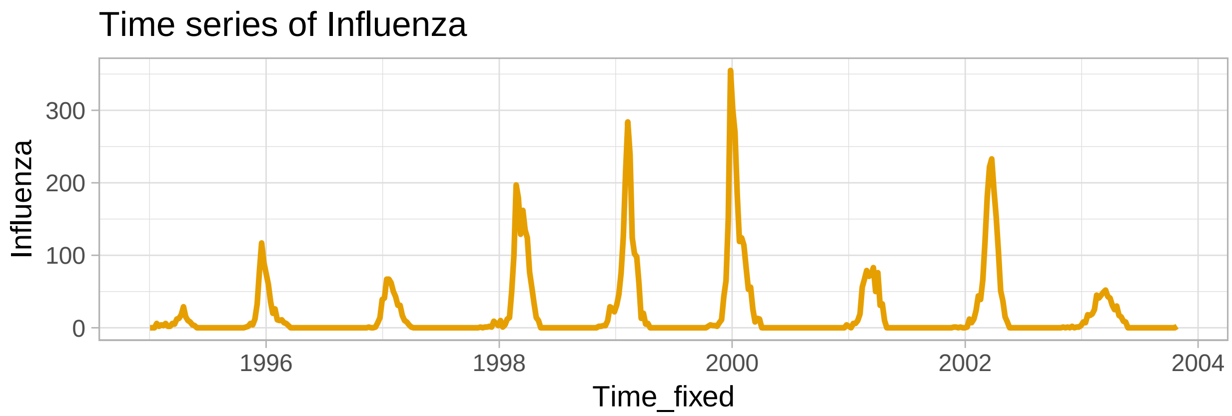
Loading The Libraries

```
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))
```

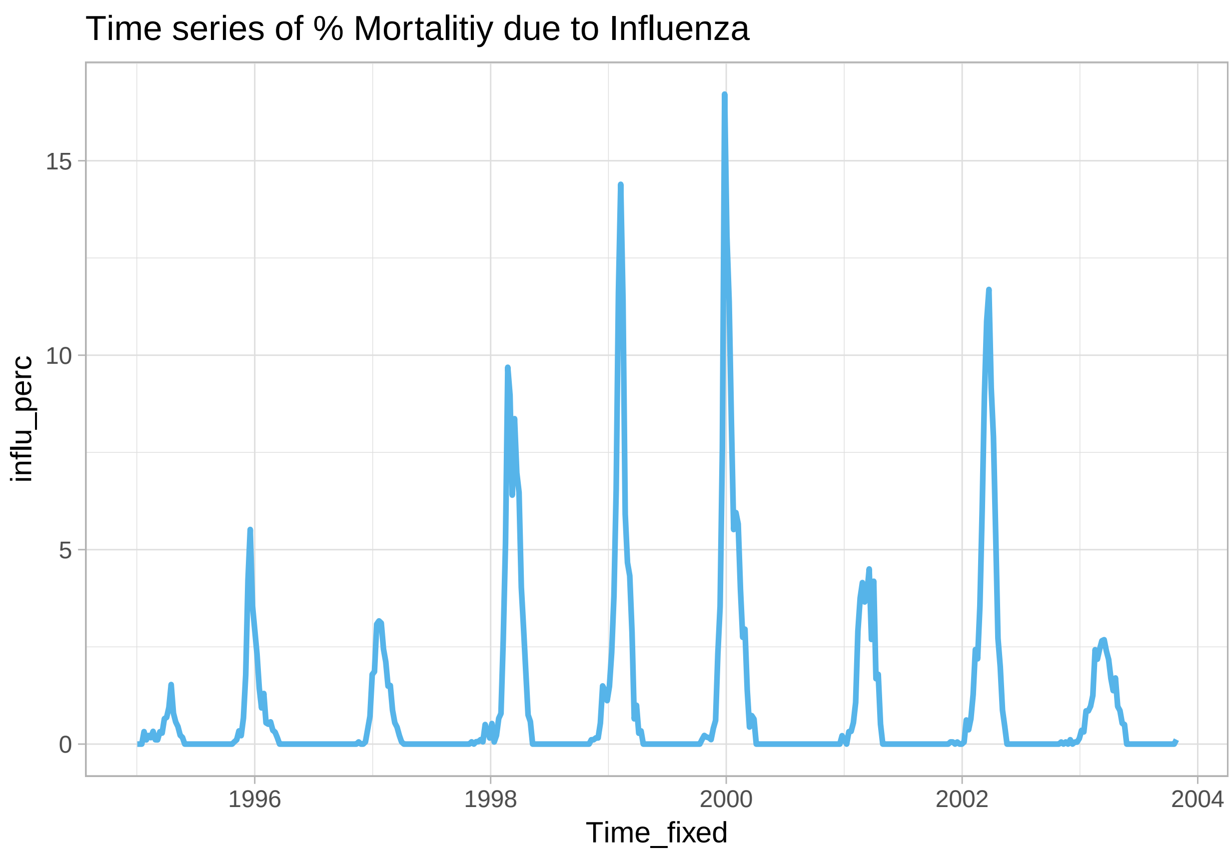
1. Use time series plots to visually inspect how the mortality and influenza number vary with time (use Time as X axis). By using this plot, comment how the amounts of influenza cases are related to mortality rates.

```
set.seed(12345)

# Importing data
flu_data = read.xlsx("influenza.xlsx", sheetName = "Raw data")
flu_data$Time_fixed <- as.Date(paste(flu_data$Year, flu_data$Week, 1, sep="-"), "%Y-%U-%u")
flu_data$influ_perc <- (flu_data$Influenza/flu_data$Mortality) * 100

# Plot

p1 <- ggplot(flu_data, aes(x=Time_fixed, y = Mortality)) +
  geom_line(color = "#999999", size = 1) +
    scale_fill_brewer() +
      theme_light() +
  ggtitle("Time series of Mortality")

p2 <- ggplot(flu_data, aes(x=Time_fixed, y = Influenza)) +
  geom_line(color = "#E69F00", size = 1) +
      scale_fill_brewer() +
      theme_light() +
  ggtitle("Time series of Influenza")

p3 <- ggplot(flu_data, aes(x=Time_fixed, y = influ_perc)) +
  geom_line(color = "#56B4E9", size = 1) +
      scale_fill_brewer() +
      theme_light() +
  ggtitle("Time series of % Mortalitiy due to Influenza")

gridExtra::grid.arrange(p1, p2, ncol=1)
```


Time series of Mortality
2400
2100
1800
1500
Mortality
1996
1998
2000
2002
2004
Time_fixed


Time series of Influenza
300
200
100
0
Influenza
1996
1998
2000
2002
2004
Time_fixed

```
p3
```



Analsis: From the plots is we can defintely see that Influenza and Mortalitiy in the given dataset are in sync, everytime Mortality peaks so does influenza, however the magnitiude of peaking is not in sync, that is the highest cases of mortaility were observed in '1996' while for influenza its in year '2000'.

From the third plot, we can see the percentage of mortalitiy due to influenza, here also the peaks match with the other plots, suggests that these two events are closely correleated.

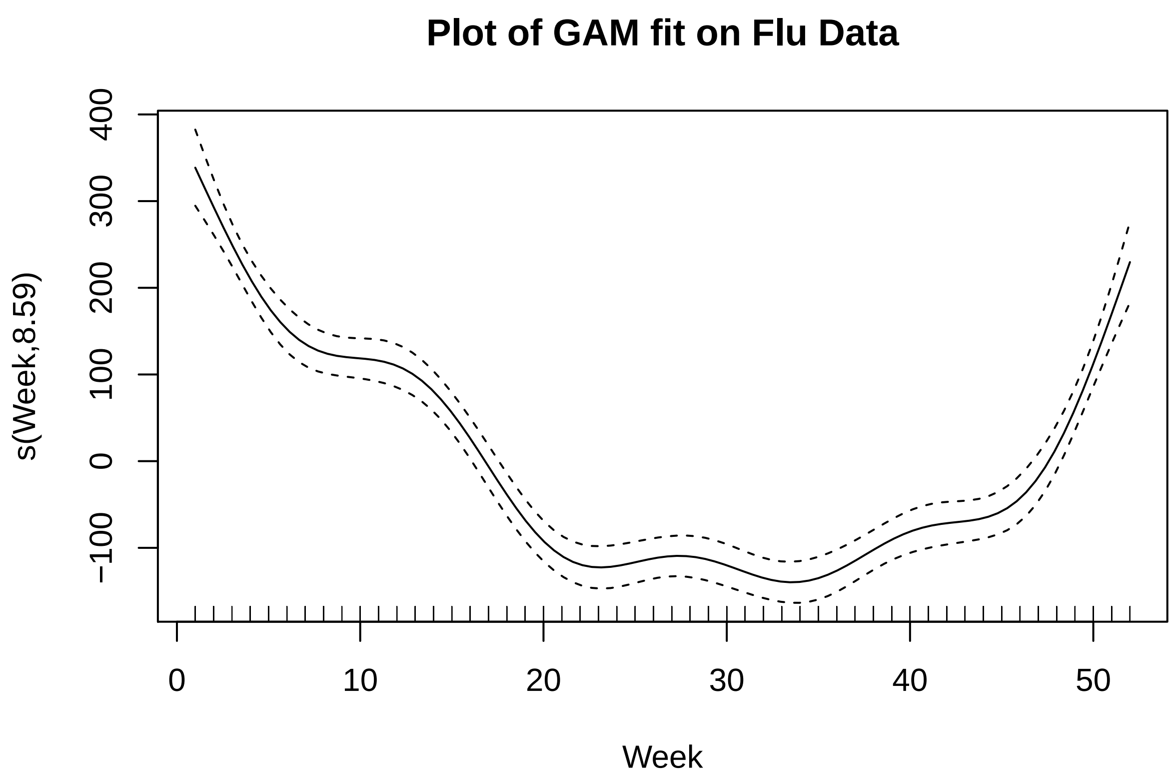
2. Use gam() function from mgcv package to fit a GAM model in which Mortality is normally distributed and modelled as a linear function of Year and spline function of Week, and make sure that the model parameters are selected by the generalized cross-validation. Report the underlying probabilistic model.

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

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ Year + s(Week)
##
## 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
```

```r
#plot the fit

p4 <- plot(gam_model, main= "Plot of GAM fit on Flu Data")
```




```r
gam_model$coefficients
```

```
##   (Intercept)          Year     s(Week).1     s(Week).2     s(Week).3
##   -652.059509      1.218569     -5.751113   -835.903619    -91.847456
##     s(Week).4     s(Week).5     s(Week).6     s(Week).7     s(Week).8
##    620.655661    212.284749   -643.469495   -147.338050  -1646.394212
##     s(Week).9
##    108.337112
```

```r
predict.gam(gam_model, type = "link")
```

```
##        1        2        3        4        5        6        7        8
## 2117.570 2072.305 2028.868 1989.539 1956.503 1931.229 1914.082 1904.197
##        9       10       11       12       13       14       15       16
```

```
## 1899.604 1897.552 1894.979 1889.036 1877.571 1859.493 1834.952 1805.298
##       17       18       19       20       21       22       23       24
## 1772.838 1740.425 1710.945 1686.823 1669.602 1659.710 1656.419 1658.043
##       25       26       27       28       29       30       31       32
## 1662.300 1666.788 1669.467 1669.058 1665.266 1658.793 1651.140 1644.230
##       33       34       35       36       37       38       39       40
## 1639.944 1639.680 1644.010 1652.547 1664.027 1676.617 1688.365 1697.707
##       41       42       43       44       45       46       47       48
## 1703.920 1707.394 1709.677 1713.252 1721.089 1736.055 1760.310 1794.823
##       49       50       51       52       53       54       55       56
## 1839.122 1891.378 1948.841 2008.627 2118.788 2073.523 2030.086 1990.757
##       57       58       59       60       61       62       63       64
## 1957.722 1932.448 1915.301 1905.416 1900.823 1898.770 1896.198 1890.255
##       65       66       67       68       69       70       71       72
## 1878.790 1860.712 1836.170 1806.516 1774.057 1741.643 1712.164 1688.041
##       73       74       75       76       77       78       79       80
## 1670.821 1660.928 1657.637 1659.261 1663.518 1668.006 1670.686 1670.277
##       81       82       83       84       85       86       87       88
## 1666.484 1660.011 1652.359 1645.448 1641.163 1640.898 1645.228 1653.765
##       89       90       91       92       93       94       95       96
## 1665.246 1677.836 1689.583 1698.926 1705.138 1708.613 1710.896 1714.471
##       97       98       99      100      101      102      103      104
## 1722.307 1737.273 1761.529 1796.042 1840.341 1892.596 1950.060 2009.846
##      105      106      107      108      109      110      111      112
## 2120.007 2074.742 2031.305 1991.976 1958.940 1933.666 1916.519 1906.634
##      113      114      115      116      117      118      119      120
## 1902.041 1899.989 1897.416 1891.473 1880.008 1861.931 1837.389 1807.735
##      121      122      123      124      125      126      127      128
## 1775.276 1742.862 1713.382 1689.260 1672.040 1662.147 1658.856 1660.480
##      129      130      131      132      133      134      135      136
## 1664.737 1669.225 1671.905 1671.496 1667.703 1661.230 1653.577 1646.667
##      137      138      139      140      141      142      143      144
## 1642.382 1642.117 1646.447 1654.984 1666.464 1679.054 1690.802 1700.144
##      145      146      147      148      149      150      151      152
## 1706.357 1709.831 1712.114 1715.689 1723.526 1738.492 1762.747 1797.260
##      153      154      155      156      157      158      159      160
## 1841.559 1893.815 1951.278 2011.064 2121.225 2075.960 2032.523 1993.194
##      161      162      163      164      165      166      167      168
## 1960.159 1934.885 1917.738 1907.853 1903.260 1901.208 1898.635 1892.692
##      169      170      171      172      173      174      175      176
## 1881.227 1863.149 1838.607 1808.953 1776.494 1744.080 1714.601 1690.478
##      177      178      179      180      181      182      183      184
## 1673.258 1663.365 1660.075 1661.698 1665.955 1670.443 1673.123 1672.714
##      185      186      187      188      189      190      191      192
## 1668.921 1662.449 1654.796 1647.885 1643.600 1643.335 1647.666 1656.202
##      193      194      195      196      197      198      199      200
## 1667.683 1680.273 1692.020 1701.363 1707.576 1711.050 1713.333 1716.908
##      201      202      203      204      205      206      207      208
## 1724.744 1739.710 1763.966 1798.479 1842.778 1895.033 1952.497 2012.283
##      209      210      211      212      213      214      215      216
## 2122.444 2077.179 2033.742 1994.413 1961.378 1936.104 1918.956 1909.072
##      217      218      219      220      221      222      223      224
## 1904.478 1902.426 1899.853 1893.911 1882.446 1864.368 1839.826 1810.172
##      225      226      227      228      229      230      231      232
```

```
## 1777.713 1745.299 1715.819 1691.697 1674.477 1664.584 1661.293 1662.917
##      233      234      235      236      237      238      239      240
## 1667.174 1671.662 1674.342 1673.933 1670.140 1663.667 1656.014 1649.104
##      241      242      243      244      245      246      247      248
## 1644.819 1644.554 1648.884 1657.421 1668.902 1681.492 1693.239 1702.581
##      249      250      251      252      253      254      255      256
## 1708.794 1712.268 1714.551 1718.126 1725.963 1740.929 1765.184 1799.697
##      257      258      259      260      261      262      263      264
## 1843.996 1896.252 1953.715 2013.502 2123.663 2078.397 2034.961 1995.631
##      265      266      267      268      269      270      271      272
## 1962.596 1937.322 1920.175 1910.290 1905.697 1903.645 1901.072 1895.129
##      273      274      275      276      277      278      279      280
## 1883.664 1865.586 1841.044 1811.390 1778.931 1746.518 1717.038 1692.915
##      281      282      283      284      285      286      287      288
## 1675.695 1665.803 1662.512 1664.135 1668.392 1672.880 1675.560 1675.151
##      289      290      291      292      293      294      295      296
## 1671.358 1664.886 1657.233 1650.322 1646.037 1645.773 1650.103 1658.639
##      297      298      299      300      301      302      303      304
## 1670.120 1682.710 1694.457 1703.800 1710.013 1713.487 1715.770 1719.345
##      305      306      307      308      309      310      311      312
## 1727.182 1742.147 1766.403 1800.916 1845.215 1897.470 1954.934 2014.720
##      313      314      315      316      317      318      319      320
## 2124.881 2079.616 2036.179 1996.850 1963.815 1938.541 1921.393 1911.509
##      321      322      323      324      325      326      327      328
## 1906.916 1904.863 1902.291 1896.348 1884.883 1866.805 1842.263 1812.609
##      329      330      331      332      333      334      335      336
## 1780.150 1747.736 1718.256 1694.134 1676.914 1667.021 1663.730 1665.354
##      337      338      339      340      341      342      343      344
## 1669.611 1674.099 1676.779 1676.370 1672.577 1666.104 1658.452 1651.541
##      345      346      347      348      349      350      351      352
## 1647.256 1646.991 1651.321 1659.858 1671.339 1683.929 1695.676 1705.019
##      353      354      355      356      357      358      359      360
## 1711.231 1714.705 1716.989 1720.564 1728.400 1743.366 1767.622 1802.135
##      361      362      363      364      365      366      367      368
## 1846.434 1898.689 1956.153 2015.939 2126.100 2080.835 2037.398 1998.069
##      369      370      371      372      373      374      375      376
## 1965.033 1939.759 1922.612 1912.727 1908.134 1906.082 1903.509 1897.566
##      377      378      379      380      381      382      383      384
## 1886.101 1868.023 1843.482 1813.828 1781.368 1748.955 1719.475 1695.352
##      385      386      387      388      389      390      391      392
## 1678.132 1668.240 1664.949 1666.573 1670.829 1675.318 1677.997 1677.588
##      393      394      395      396      397      398      399      400
## 1673.796 1667.323 1659.670 1652.760 1648.474 1648.210 1652.540 1661.077
##      401      402      403      404      405      406      407      408
## 1672.557 1685.147 1696.895 1706.237 1712.450 1715.924 1718.207 1721.782
##      409      410      411      412      413      414      415      416
## 1729.619 1744.584 1768.840 1803.353 1847.652 1899.908 1957.371 2017.157
##      417      418      419      420      421      422      423      424
## 2127.318 2082.053 2038.616 1999.287 1966.252 1940.978 1923.831 1913.946
##      425      426      427      428      429      430      431      432
## 1909.353 1907.300 1904.728 1898.785 1887.320 1869.242 1844.700 1815.046
##      433      434      435      436      437      438      439      440
## 1782.587 1750.173 1720.694 1696.571 1679.351 1669.458 1666.167 1667.791
##      441      442      443      444      445      446      447      448
```

```
## 1672.048 1676.536 1679.216 1678.807 1675.014 1668.541 1660.889 1653.978
##      449      450      451      452      453      454      455      456
## 1649.693 1649.428 1653.758 1662.295 1673.776 1686.366 1698.113 1707.456
##      457      458      459
## 1713.668 1717.142 1719.426
```

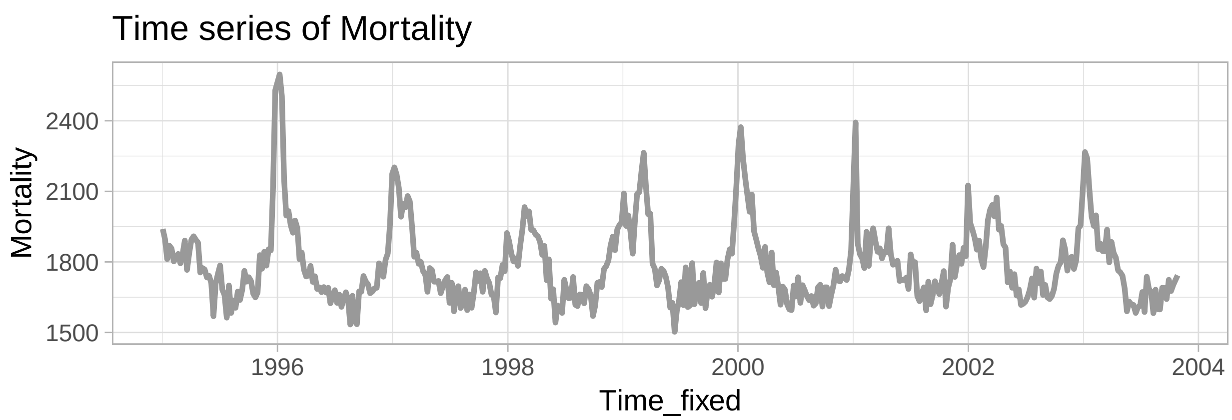
Analysis:

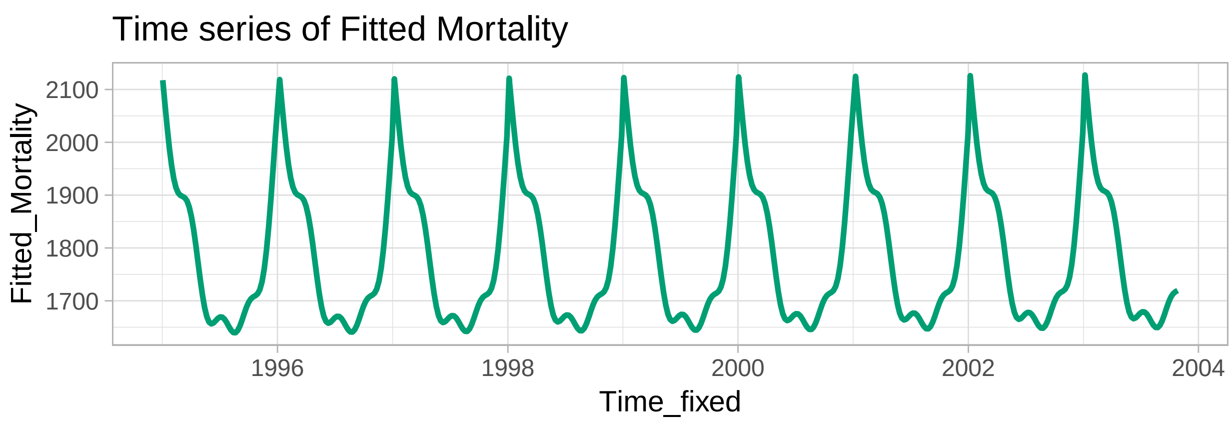
The underlying probablistic equation of the model is given by:

$$Mortality = N(\mu, \sigma^2)$$
$$g(\mu) = Intercept + Beta_{year} * Year + s(Week)$$

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.

```
temp <- flu_data
temp$Fitted_Mortality <- gam_model$fitted.values

p5 <- ggplot(data=temp, aes(x = Time_fixed, y = Fitted_Mortality)) +
   geom_line(color = "#009E73", size = 1) +
    scale_fill_brewer() +
      theme_light() +
  ggtitle("Time series of Fitted Mortality")

grid.arrange(p1, p5, nrow = 2)
```

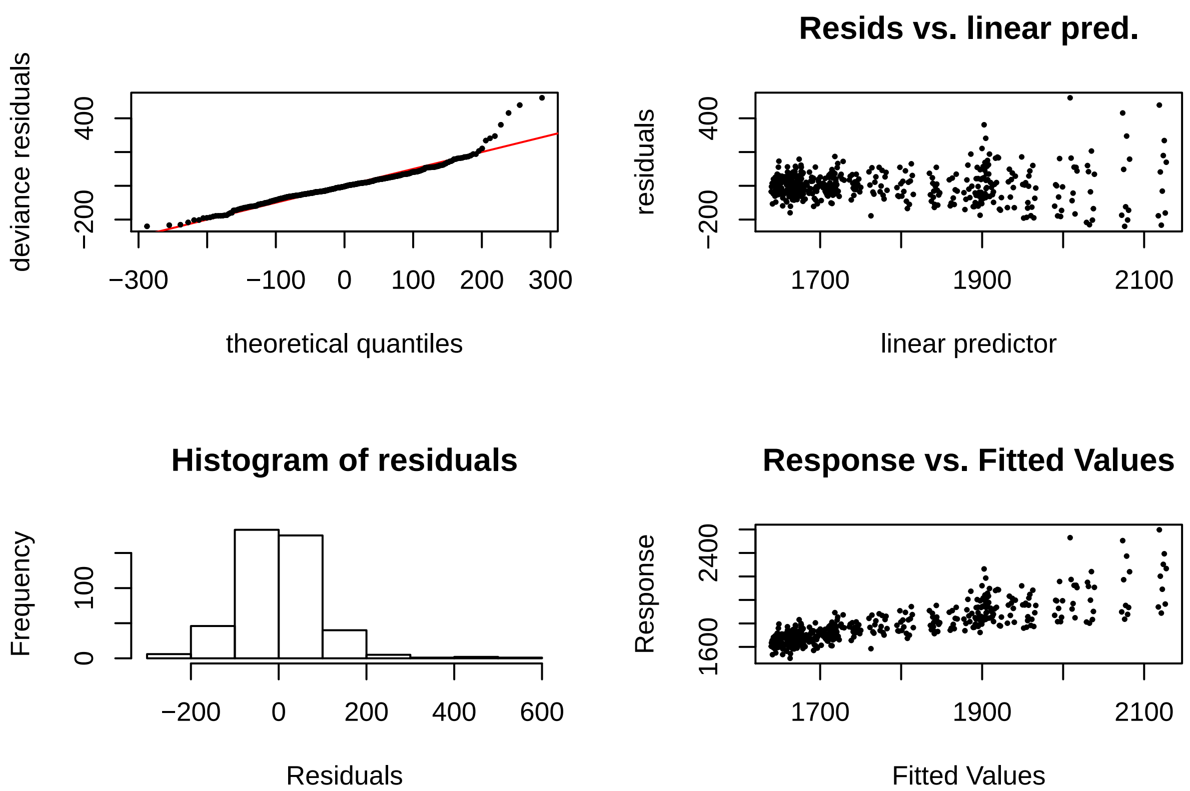





```
summary(gam_model)
```

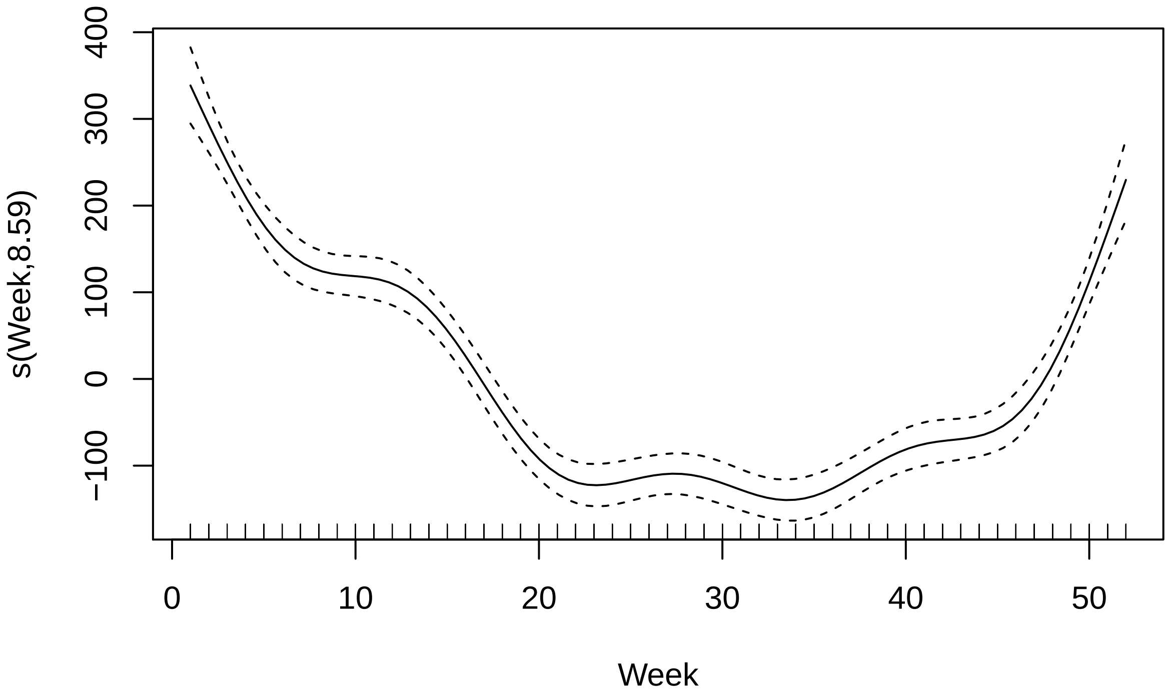
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ Year + s(Week)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -652.060   3448.379  -0.189     0.85
## Year           1.219      1.725   0.706     0.48
##
## Approximate 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
```

```
gam.check(gam_model,pch=19,cex=.3)
```




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 7 iterations.
## The RMS GCV score gradient at convergence was 0.114505 .
## 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    0.74
```

```
plot(gam_model)
```



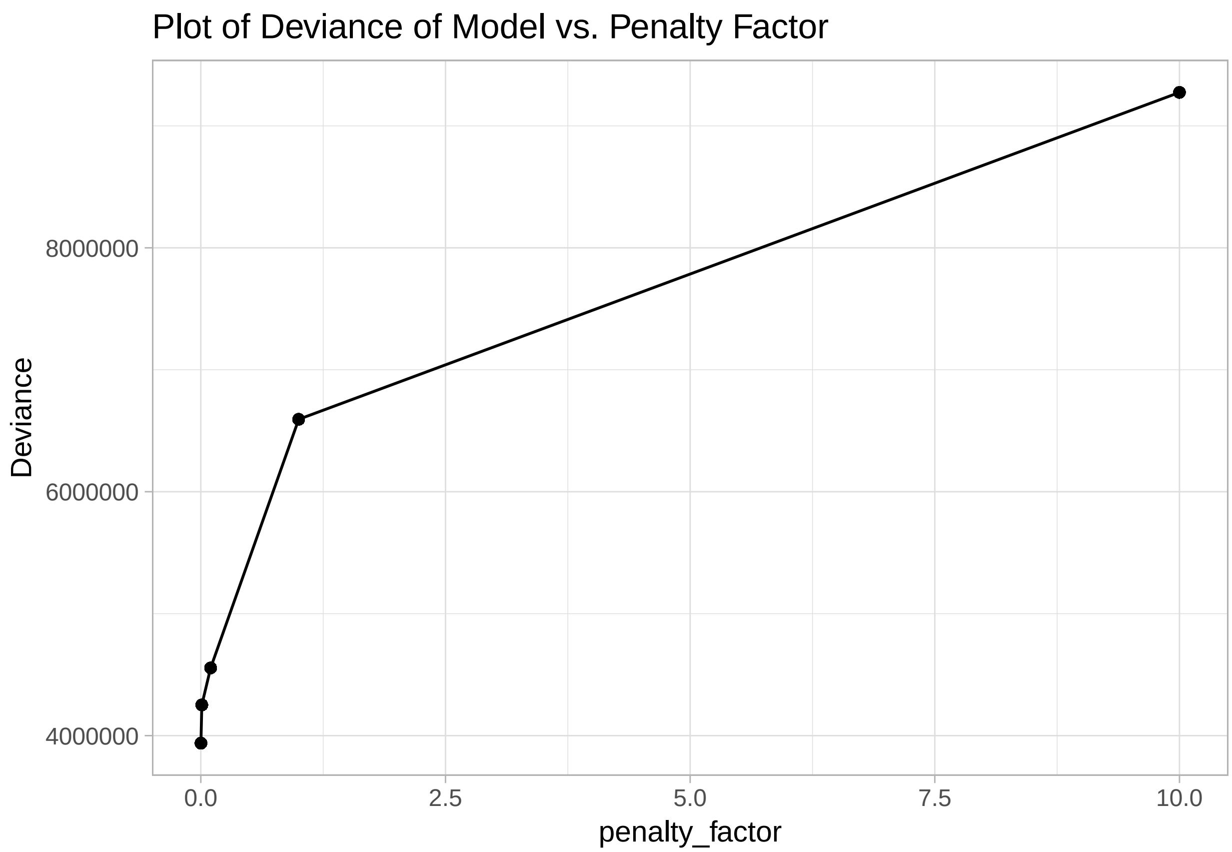

```
# s=interp(temp$Year,temp$Week, fitted(gam_model))
# persp3d(s$x, s$y, s$z, col="red")
```

Analysis: From the plot of residuals we can see that the resisuals are normally distributed. Thus this is a good fit.

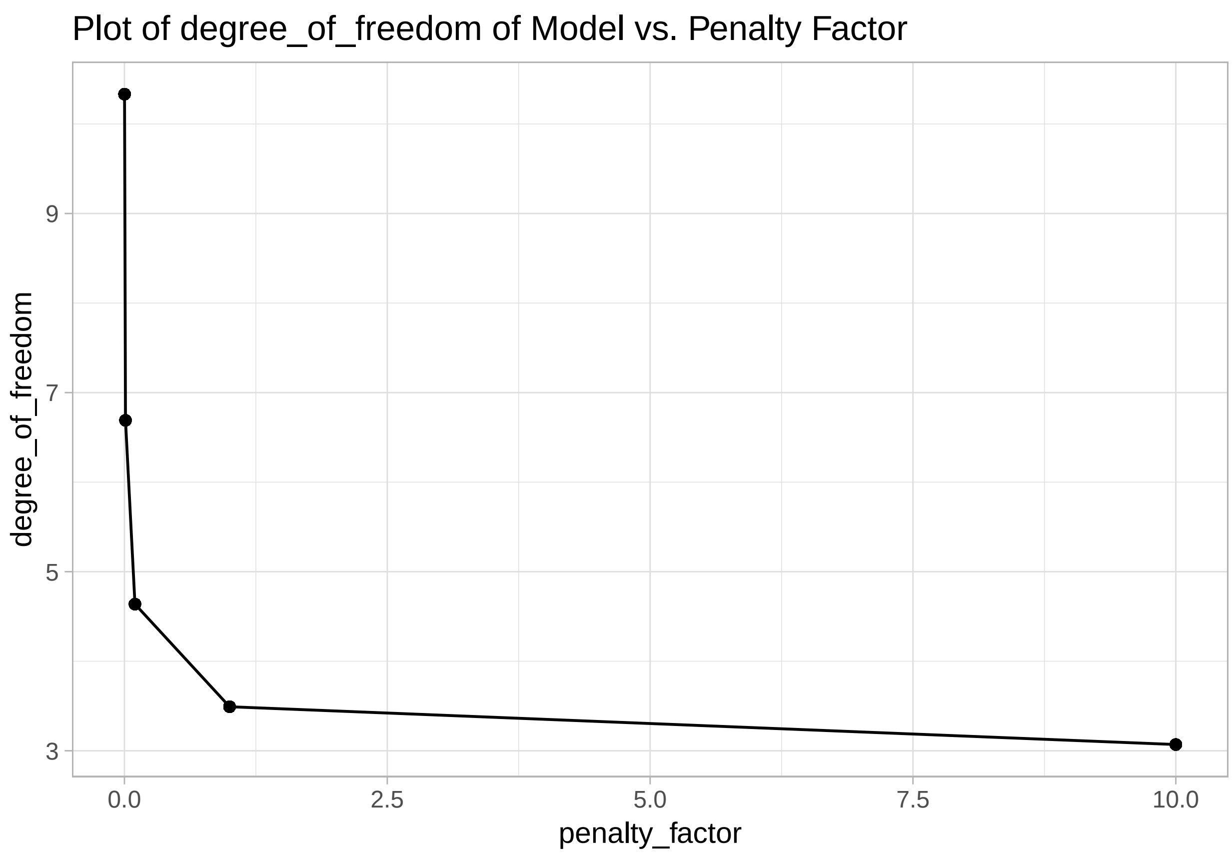
4. Examine how the penalty factor of the spline function in the GAM model from step 2 influences the estimated deviance of the model. Make plots of the predicted and observed mortality against time for cases of very high and very low penalty factors. What is the relation of the penalty factor to the 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(flu_data$Week))

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

model_deviance <- rbind(temp, model_deviance)
}
model_deviance <- as.data.frame(model_deviance)
```

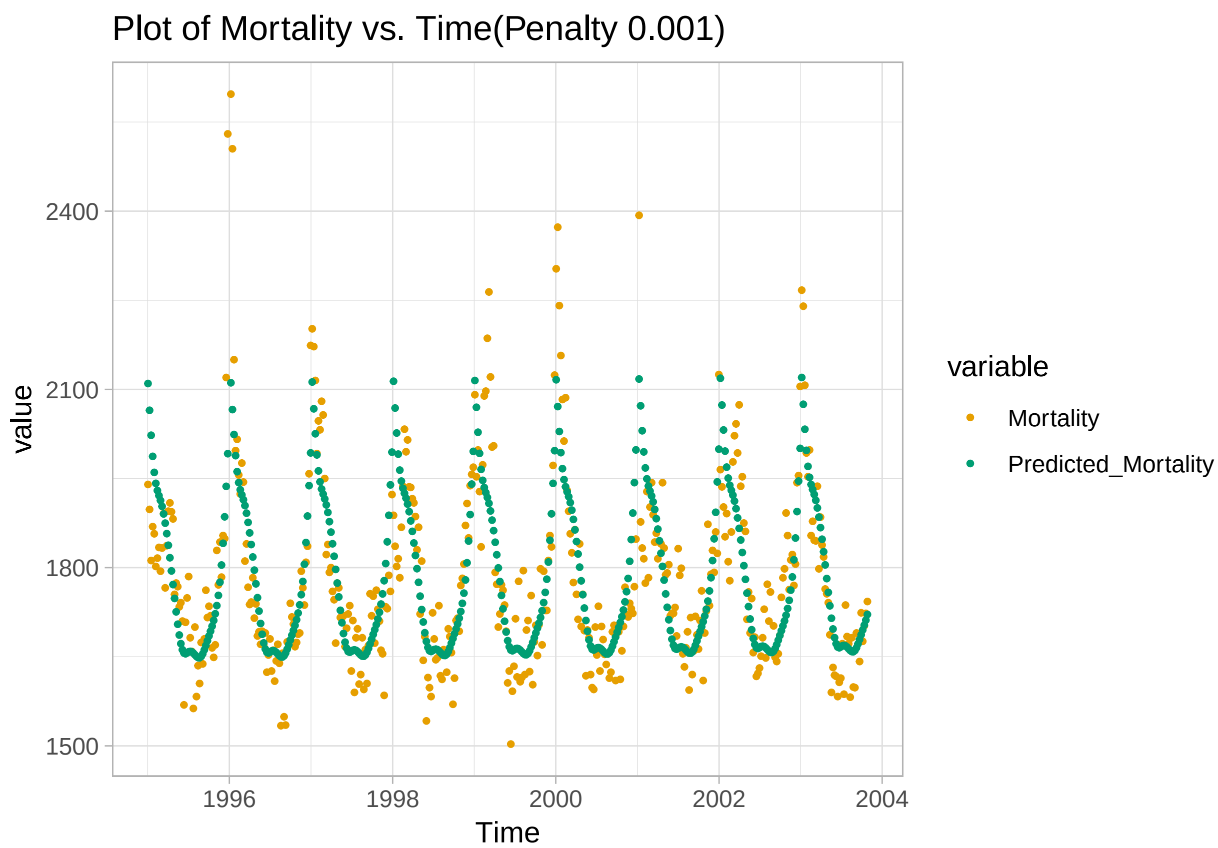
```r
colnames(model_deviance) <- c("Deviance", "Predicted_Mortality", "Mortality", "Time",
                              "penalty_factor", "degree_of_freedom")
model_deviance$Time <- as.Date(model_deviance$Time, origin = '1970-01-01')


# 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
```



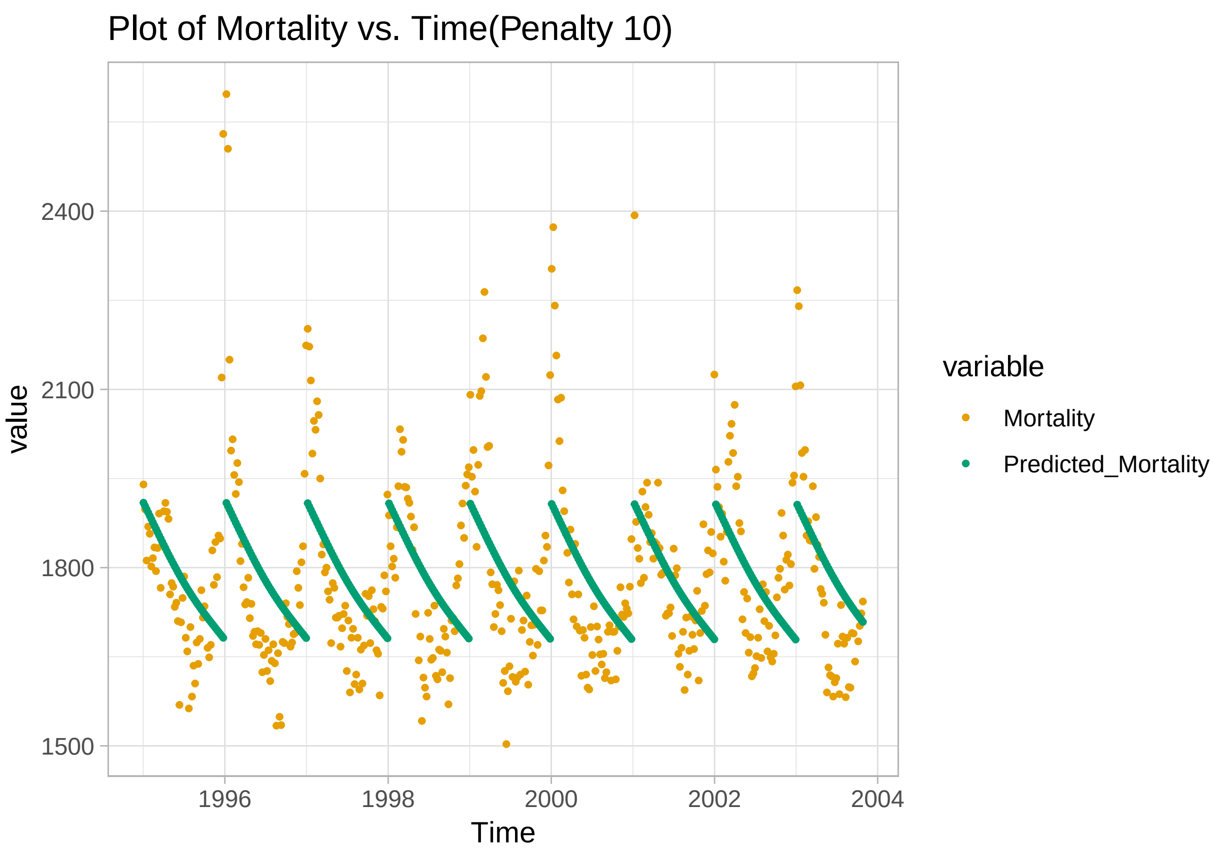

```r
# plot of degree of freedom
p7 <- ggplot(data=model_deviance, aes(x = penalty_factor, y = degree_of_freedom)) +
geom_point() +
  geom_line() +
      theme_light() +
ggtitle("Plot of degree_of_freedom of Model vs. Penalty Factor")
p7
```




```r
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) +
  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) +
  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

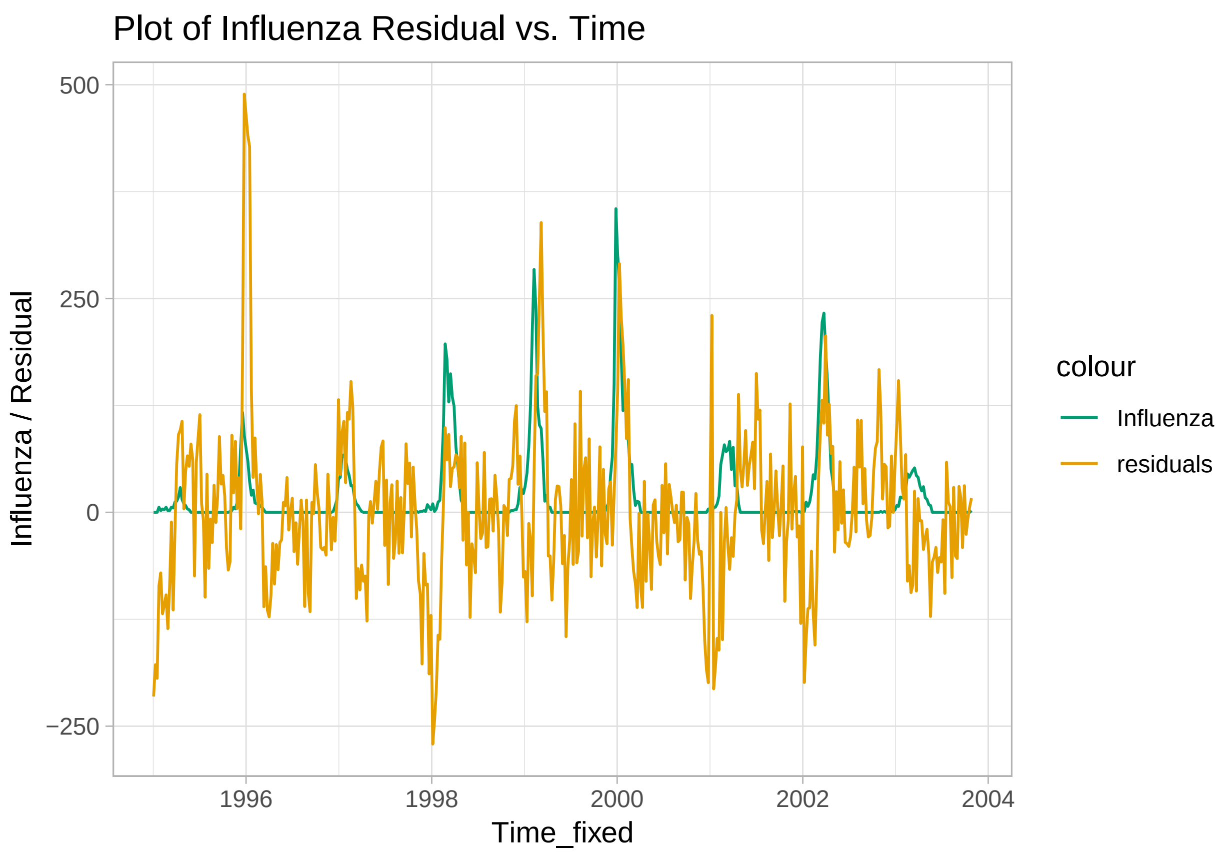
```
p9
```



Analysis: theortical maximum degree of freedom is k-1.

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?

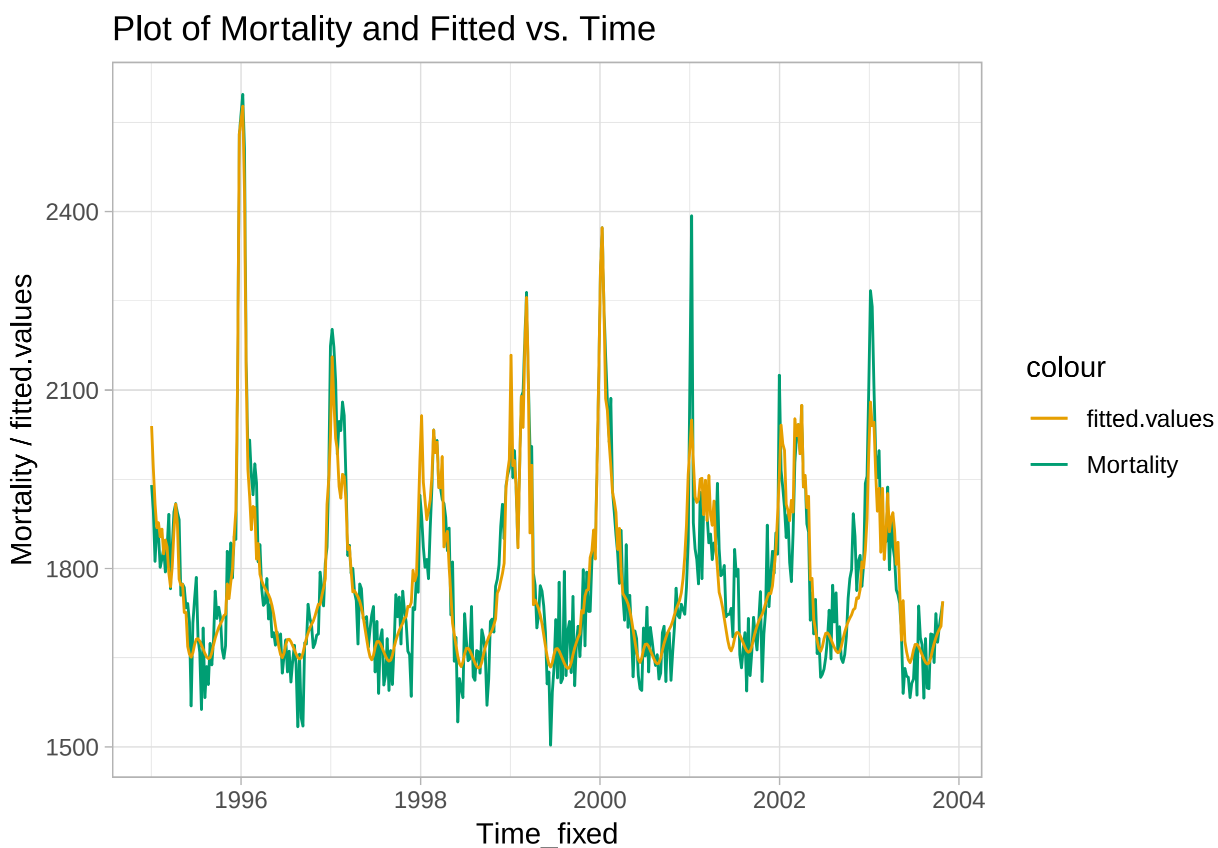
```
k=length(unique(flu_data$Week))
gam_model <- mgcv::gam(data = flu_data, Mortality~Year+s(Week, k=k), method = "GCV.Cp")

temp <- flu_data
temp <- cbind(temp, residuals = gam_model$residuals)


p10 <- ggplot(data = temp, aes(x = Time_fixed)) +
  geom_line(aes( y = Influenza, color = "Influenza")) +
  geom_line(aes(y = residuals, color = "residuals")) +
      theme_light() +
  scale_color_manual(values=c(Influenza = "#009E73", residuals = "#E69F00")) +
  labs(y = "Influenza / Residual") +
  ggtitle("Plot of Influenza Residual vs. Time")

p10
```



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.

```
#gam_model_additive <- mgcv::gam(data = flu_data, Mortality~s(Year)+s(Week), method = "GCV.Cp")

k1 = length(unique(flu_data$Year))
k2 = length(unique(flu_data$Week))
k3 = length(unique(flu_data$Influenza))

gam_model_additive <- gam(Mortality ~ s(Year, k=k1) +
                                     s(Week, k=k2) +
                                    s(Influenza, k=k3),
                          data = flu_data)


flu_data$fitted.values = gam_model_additive$fitted.values


p11 <- ggplot(data = flu_data, aes(x = Time_fixed)) +
```

```
  geom_line(aes( y = Mortality, color = "Mortality")) +
  geom_line(aes(y = fitted.values, color = "fitted.values")) +
      theme_light() +
  scale_color_manual(values=c(Mortality = "#009E73", fitted.values = "#E69F00")) +
  labs(y = "Mortality / fitted.values") +
  ggtitle("Plot of Mortality and Fitted vs. Time")

p11
```



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()
```

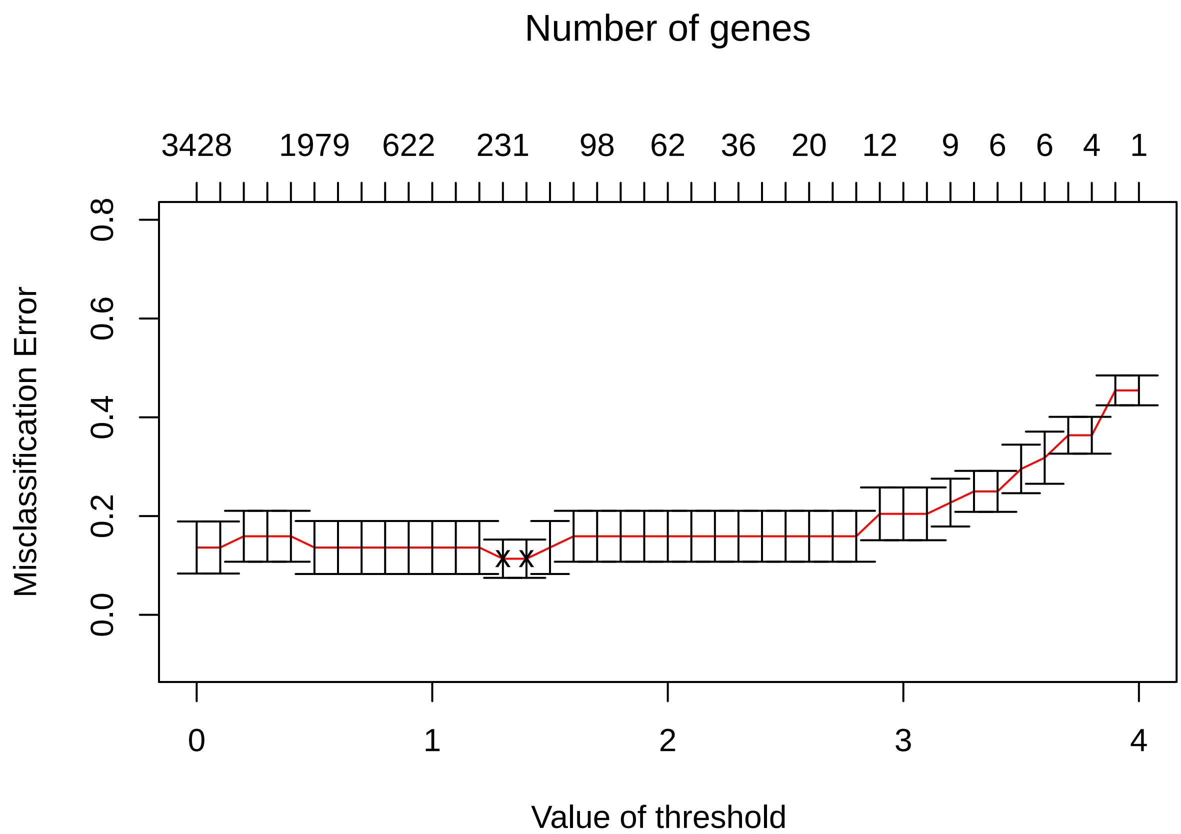
```
##           used  (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells 2898240 154.8    4746877 253.6  4746877 253.6
## Vcells 4749902  36.3   10146329  77.5  8388607  64.0
```

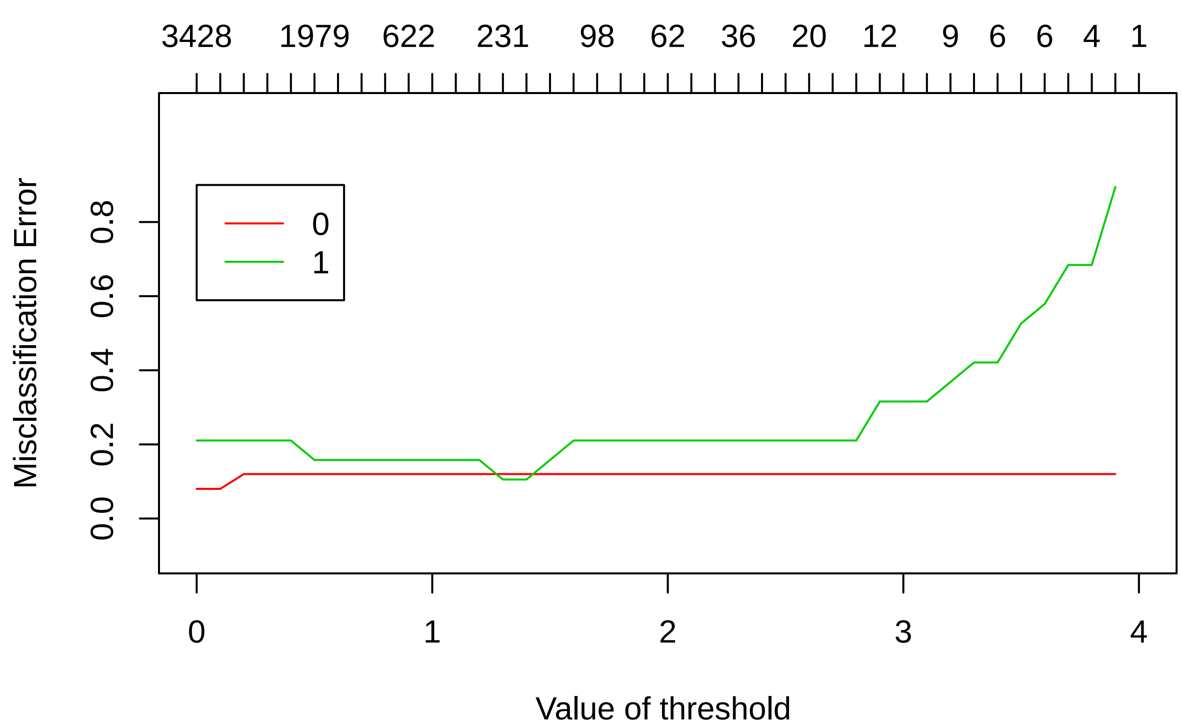
```r
data <- read.csv(file = "data.csv", sep = ";", header = TRUE)
```

```r
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)
```

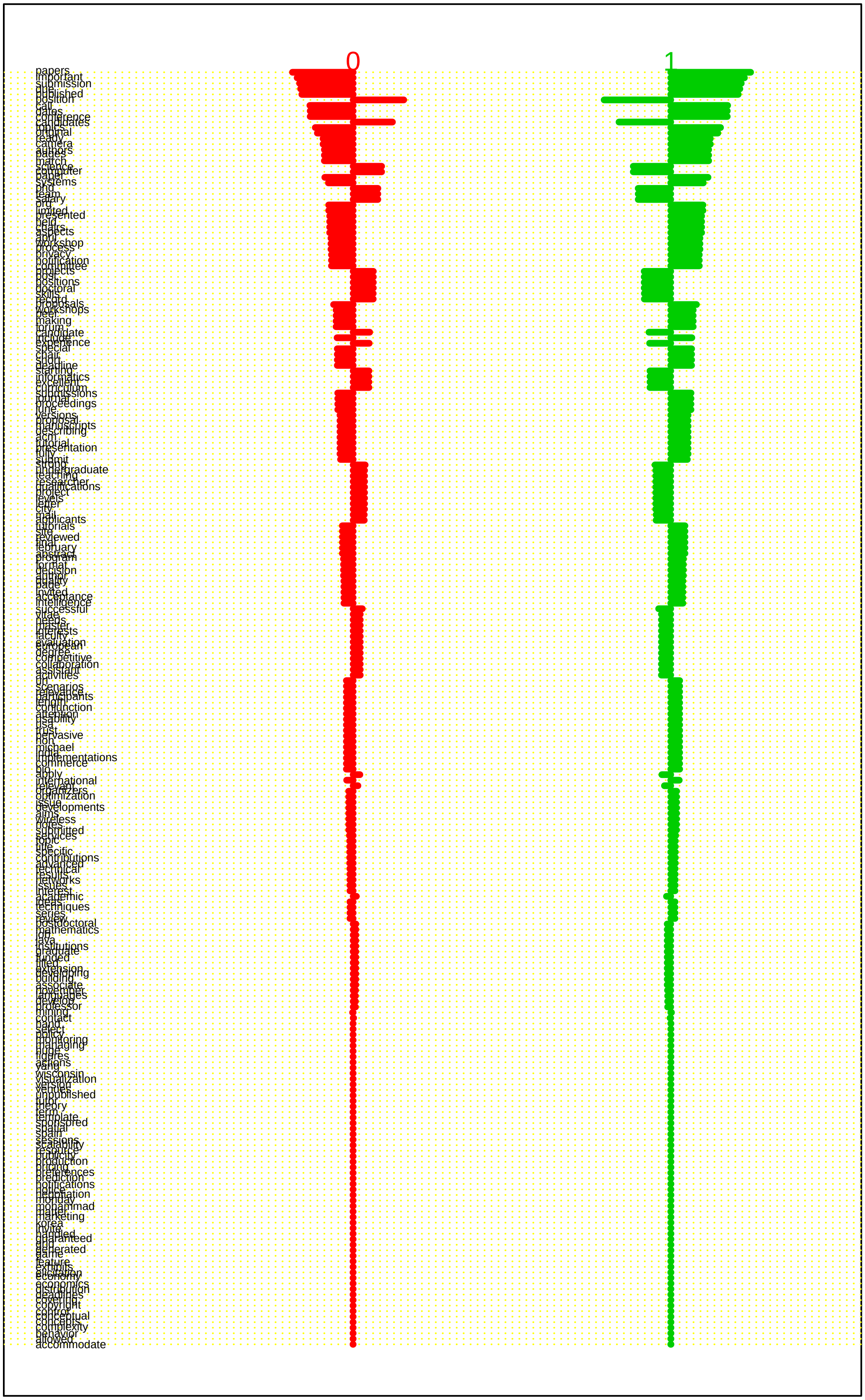
plots

```r
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

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


0
1

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
##
```

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_coe
```

```r
elastic_variable
```

```
##           name  coefficient
## 1  (Intercept) -1.018931295
## 2    abstracts -0.301126433
## 3      aspects  0.073677580
## 4          bio  0.022876514
## 5         call  0.331990016
## 6   candidates -0.187831077
## 7     computer -0.283206491
## 8   conceptual  0.038084357
## 9   conference  0.196532966
## 10       dates  0.241663004
## 11         due  0.521172495
## 12  evaluation -0.179640082
## 13    exhibits  0.378269987
## 14   important  0.392427522
## 15   languages -0.025846994
## 16      making  0.189239367
## 17 manuscripts  0.032558442
## 18    original  0.055820470
## 19      papers  0.385380979
## 20        peer  0.096721108
## 21    position -0.375082994
## 22     process  0.001623837
## 23    projects -0.190407998
## 24   proposals  0.055355377
## 25   published  0.281820589
## 26     queries -0.300245879
## 27      record -0.116251400
## 28    relevant -0.113556406
## 29   scenarios  0.005346950
## 30     spatial  0.192500683
## 31  submission  0.280351935
## 32        team -0.129127761
## 33    versions  0.154574908
```

```r
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()

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

knitr::kable(final_result, caption = "Accuracy of Model on Test dataset")
```

Table 1: Accuracy of Model on Test dataset

Accuracy of Nearest Shrunken Centroid Model	Accuracy of ElasticNet	Accuracy SVM Model
90	90	95

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.

```
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)
colnames(data[,keep])
```

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

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
if (!require("pacman")) install.packages("pacman")
pacman::p_load(xlsx, ggplot2, tidyr, dplyr, reshape2, gridExtra,
```

```r
               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))
set.seed(12345)

# Importing data
flu_data = read.xlsx("influenza.xlsx", sheetName = "Raw data")
flu_data$Time_fixed <- as.Date(paste(flu_data$Year, flu_data$Week, 1, sep="-"), "%Y-%U-%u")
flu_data$influ_perc <- (flu_data$Influenza/flu_data$Mortality) * 100

# Plot

p1 <- ggplot(flu_data, aes(x=Time_fixed, y = Mortality)) +
  geom_line(color = "#999999", size = 1) +
    scale_fill_brewer() +
      theme_light() +
  ggtitle("Time series of Mortality")

p2 <- ggplot(flu_data, aes(x=Time_fixed, y = Influenza)) +
  geom_line(color = "#E69F00", size = 1) +
      scale_fill_brewer() +
      theme_light() +
  ggtitle("Time series of Influenza")

p3 <- ggplot(flu_data, aes(x=Time_fixed, y = influ_perc)) +
  geom_line(color = "#56B4E9", size = 1) +
      scale_fill_brewer() +
      theme_light() +
  ggtitle("Time series of % Mortalitiy due to Influenza")

gridExtra::grid.arrange(p1, p2, ncol=1)
p3

gam_model <- mgcv::gam(data = flu_data, Mortality~Year+s(Week), method = "GCV.Cp")
summary(gam_model)
#plot the fit

p4 <- plot(gam_model, main= "Plot of GAM fit on Flu Data")
gam_model$coefficients
predict.gam(gam_model, type = "link")
temp <- flu_data
temp$Fitted_Mortality <- gam_model$fitted.values

p5 <- ggplot(data=temp, aes(x = Time_fixed, y = Fitted_Mortality)) +
   geom_line(color = "#009E73", size = 1) +
```

```r
    scale_fill_brewer() +
      theme_light() +
  ggtitle("Time series of Fitted Mortality")

grid.arrange(p1, p5, nrow = 2)

summary(gam_model)
gam.check(gam_model,pch=19,cex=.3)

plot(gam_model)

# s=interp(temp$Year,temp$Week, fitted(gam_model))
# persp3d(s$x, s$y, s$z, col="red")
model_deviance <- NULL
for(sp in c(0.001, 0.01, 0.1, 1, 10))
{
  k=length(unique(flu_data$Week))

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

```r
  geom_point(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) +
  scale_color_manual(values=c("#E69F00", "#009E73")) +
    theme_light() +
  ggtitle("Plot of Mortality vs. Time(Penalty 10)")

p8
p9


k=length(unique(flu_data$Week))
gam_model <- mgcv::gam(data = flu_data, Mortality~Year+s(Week, k=k), method = "GCV.Cp")

temp <- flu_data
temp <- cbind(temp, residuals = gam_model$residuals)


p10 <- ggplot(data = temp, aes(x = Time_fixed)) +
  geom_line(aes( y = Influenza, color = "Influenza")) +
  geom_line(aes(y = residuals, color = "residuals")) +
      theme_light() +
  scale_color_manual(values=c(Influenza = "#009E73", residuals = "#E69F00")) +
  labs(y = "Influenza / Residual") +
  ggtitle("Plot of Influenza Residual vs. Time")

p10

#gam_model_additive <- mgcv::gam(data = flu_data, Mortality~s(Year)+s(Week), method = "GCV.Cp")

k1 = length(unique(flu_data$Year))
k2 = length(unique(flu_data$Week))
k3 = length(unique(flu_data$Influenza))

gam_model_additive <- gam(Mortality ~ s(Year, k=k1) +
                                      s(Week, k=k2) +
                                     s(Influenza, k=k3),
                          data = flu_data)


flu_data$fitted.values = gam_model_additive$fitted.values


p11 <- ggplot(data = flu_data, aes(x = Time_fixed)) +
  geom_line(aes( y = Mortality, color = "Mortality")) +
  geom_line(aes(y = fitted.values, color = "fitted.values")) +
      theme_light() +
  scale_color_manual(values=c(Mortality = "#009E73", fitted.values = "#E69F00")) +
```

```
  labs(y = "Mortality / fitted.values") +
  ggtitle("Plot of Mortality and Fitted vs. Time")

p11

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)
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_coe
elastic_variable


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)
```

```r
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)

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

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

knitr::kable(final_result, caption = "Accuracy of Model 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)
colnames(data[,keep])
```