Indian Liver Patients

Sung Wook Choi

22/04/2020

The dataset which I have obtained from Kaggle was titled 'Indian Liver Patient Records." The original dataset (before wrangling) included features that were related to whether or not patients have liver diseases. The features (the columns of the original dataset) were the following:

Age: Age of the patient Gender: Gender of the patient Total_Bilirubin: Total Bilirubin Direct_Bilirubin: Direct Bilirubin Alkaline_Phosphotase:Alkaline Phosphotase Alamine_Aminotransferase: Alamine Aminotransferase Aspartate_Aminotransferase: Aspartate Aminotransferase Total_Protiens: Total Protiens Albumin: Albumin Albumin_and_Globulin_Ratio: Albumin and Globulin Ratio Dataset: field used to split the data into two sets (patient with liver disease (1), or no disease (2))

The frequencies of liver diseases have been increasing continuously affected by modern day lifestyles such as heavy drinking and harmful gases in the atmosphere. The goal of this project was to create a prediction algorithm so that with unknown feature datas, liver diseases within patients could be predicted in order to help the doctors diagnose liver diseases. The main steps that were performed were the following

1. Importing of the data
2. Data cleaning/wrangling
3. Data Exploration/Visualization
4. Preprocessing
5. Machine Learning

<Methods & Analysis> The following section will be explained along with the codes (comments are made both with and without hashtag symbols).

1. Importing Data/Installing Packages

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(matrixStats)
```

```
##
## Attaching package: 'matrixStats'
```

```
## The following object is masked from 'package:dplyr':
##
##     count
```

```
library(Matrix)
library(rpart)

if(!require("PerformanceAnalytics")){
    install.packages("PerformanceAnalytics")
    library(PerformanceAnalytics)
}
```

```
## Loading required package: PerformanceAnalytics
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
##
## Attaching package: 'PerformanceAnalytics'
```

```
## The following object is masked from 'package:graphics':
##
##     legend
```

```
if(!require("gam")){
    install.packages("gam")
    library(gam)
}
```

```
## Loading required package: gam
```

```
## Loading required package: splines
```

```
## Loading required package: foreach
```

```
## Loaded gam 1.16.1
```

```
if(!require("naivebayes")){
    install.packages("naivebayes")
    library(naivebayes)
}
```

```
## Loading required package: naivebayes
```

```
## naivebayes 0.9.7 loaded
```

```
if(!require("splines")){
    install.packages("splines")
    library(splines)
}

getwd()
setwd("/Users/st_woogie/Desktop") #the csv will be uploaded onto my github
dat <- read.csv("indian_liver_patient.csv")

nrow(dat) #11 rows
ncol(dat) #583 columns
head(dat)
```

2. Data Cleaning/Wrangling

```
sum(is.na(dat)) #4 rows with na as inputs
```

```
## [1] 4
```

```
newdat <- na.omit(dat) #omit all na's
Disease <- ifelse(newdat$Dataset == 1, 1, 0)

# Changing 1(has liver disease) and 2(doens't have liver disease), to 0(doesn't have liv
er disease) and 1(has liver disease)

Sex <- ifelse(newdat$Gender == 'Male', 0, 1) #Let male be 0 and female be 1 (binarizing
 the data)
newdat <- cbind(Sex, newdat, Disease)
newdat <- newdat[,-c(3,12)]
newdat$Sex <- as.factor(newdat$Sex)
newdat$Disease <- as.factor(newdat$Disease) #Turned the features 'Sex' and our response
 'Dataset' into factors)
head(newdat)
```

	Sex	...	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotrans
	<fctr>	<int>	<dbl>	<dbl>	<int>	
1	1	65	0.7	0.1	187	
2	0	62	10.9	5.5	699	
3	0	62	7.3	4.1	490	
4	0	58	1.0	0.4	182	
5	0	72	3.9	2.0	195	
6	0	46	1.8	0.7	208	

6 rows | 1-7 of 12 columns

3.Data Exploration/Preprocessing

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────────────────────
──────────── tidyverse 1.3.0 ──
```

```
## ✓ tibble  2.1.3     ✓ purrr   0.3.3
## ✓ tidyr   1.0.0     ✓ stringr 1.4.0
## ✓ readr   1.3.1     ✓ forcats 0.4.0
```

```
## ── Conflicts ──────────────────────────────────────────────────────────────────
─────── tidyverse_conflicts() ──
## x purrr::accumulate()  masks foreach::accumulate()
## x matrixStats::count() masks dplyr::count()
## x tidyr::expand()      masks Matrix::expand()
## x dplyr::filter()      masks stats::filter()
## x xts::first()         masks dplyr::first()
## x dplyr::lag()         masks stats::lag()
## x xts::last()          masks dplyr::last()
## x purrr::lift()        masks caret::lift()
## x tidyr::pack()        masks Matrix::pack()
## x tidyr::unpack()      masks Matrix::unpack()
## x purrr::when()        masks foreach::when()
```

```
nrow(newdat) #579 observations after omitting all the na values
```

```
## [1] 579
```

```
ncol(newdat) # two categorical features and the rest are numeric
```

```
## [1] 11
```

```
mean(Disease == 1) # Proportion of diseased in our dataset is 71.5 percent
```

```
## [1] 0.7150259
```

```
mean(Sex == 0) # Proportion of male seems higher
```

```
## [1] 0.7582038
```

```
newdat %>% group_by(Sex, Disease) %>% summarize(count = n())
```

Sex <fctr>	Disease <fctr>	count <int>
0	0	116
0	1	323
1	0	49
1	1	91

4 rows

```
(323/579)/0.715 # conditional prob that a person is diseased given the sex is male
```

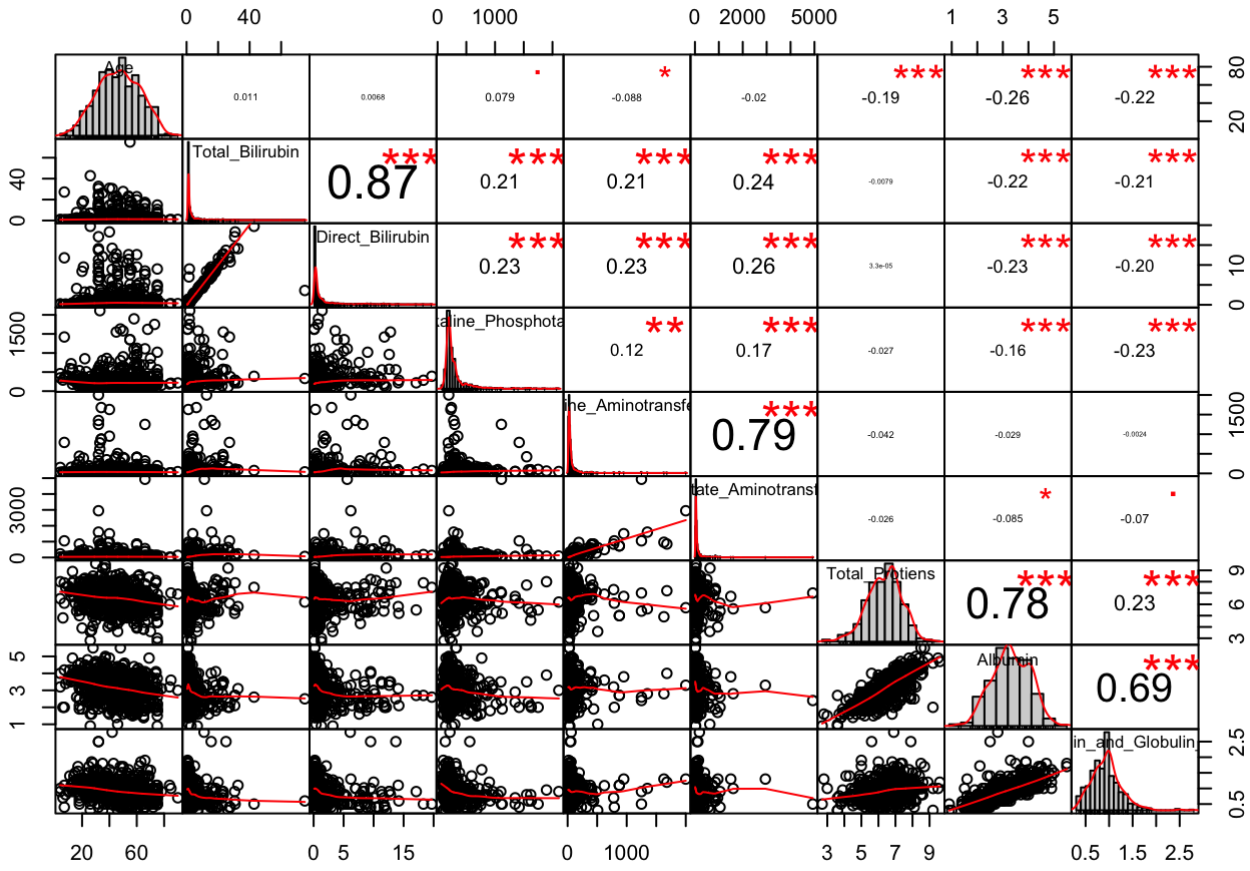
```
## [1] 0.7802215
```

```
(91/579)/0.285 # conditional prob that a person is diseased given the sex is female
```

```
## [1] 0.551465
```

```
# Taking a look at correlation between features (All the factor features were excluded i
n this plot)

if(!require("PerformanceAnalytics")){
    install.packages("PerformanceAnalytics")
    library("PerformanceAnalytics")
}
chart.Correlation(newdat[,-c(1,11)], histogram=TRUE, pch=19)
```




```
# In the middle shows the disribution of each variable
# The bottom shows the bivariate scatter plots with a fitted line
# This plot shows a sign of little colinearity between the features which is a good sign
```

It could be seen that there is a highest correlation of 0.87 (strong, postive and near 1 correlation) between the features Total_Bilirubin and Direct_Bilirubin. Through the process of variable selection I have decided to take out the feature “Total_Bilirubin.”Also, I decided to exclude the variable "Total_proteins as it had the least significance with other feature variables (these significances are calculated using p-values).

```
# Standardizing the numeric features since they range so vastly (by using the sweep func
tion, and matrix manipulation)
subset <- newdat[,-c(1,11)]
subset <- as.matrix(subset)
x_mean0 <- sweep(subset, 2, colMeans(subset))
x_standardized <- sweep(x_mean0, 2, colSds(subset), FUN = "/")
newdat <- cbind(x_standardized,newdat[,c(1,11)])
```

```
# Removing variables 'Total_Bilirubin' and 'Total_proteins'
head(newdat)
```

	Age <dbl>	Total_Bilirubin <dbl>	Direct_Bilirubin <dbl>	Alkaline_Phosphotase <dbl>	Alamine_Aminotr
1	1.24632497	-0.41995671	-0.4949862	-0.4284995	-0
2	1.06138848	1.21788279	1.4222880	1.6736358	-0
3	1.06138848	0.63982179	0.9252169	0.8155376	-0
4	0.81480651	-0.37178496	-0.3884709	-0.4490282	-0
5	1.67784343	0.09387529	0.1796103	-0.3956537	-0
6	0.07506058	-0.24332696	-0.2819557	-0.3422792	-0

6 rows | 1-6 of 12 columns

```
newdat <- newdat[,-c(2,7)]
head(newdat)
```

	Age <dbl>	Direct_Bilirubin <dbl>	Alkaline_Phosphotase <dbl>	Alamine_Aminotransferase <dbl>
1	1.24632497	-0.4949862	-0.4284995	-0.35552499
2	1.06138848	1.4222880	1.6736358	-0.09349172
3	1.06138848	0.9252169	0.8155376	-0.11532783
4	0.81480651	-0.3884709	-0.4490282	-0.36644304
5	1.67784343	0.1796103	-0.3956537	-0.29547570
6	0.07506058	-0.2819557	-0.3422792	-0.33914791

6 rows | 1-5 of 10 columns

4. Machine Learning

```
# Partitioning data (Used 0.1 to 0.9 instead half and half, the number of observations i
s too small to train algorithms with half of the data)
set.seed(1)
test_index <- createDataPartition(newdat$Disease, times = 1, p = 0.1, list = FALSE)
test <- newdat %>% slice(test_index)
train <- newdat %>% slice(-test_index)
```

```
# Random Forest
set.seed(8)
train_rf1 <- train(Disease ~ ., method = 'rf', tunegrid = data.frame(mtry = seq(1,7,1)),
data = train, ntree = 100)
train_rf1$bestTune # Best mtry was chosen to be 2 after tuning parameters
```

	mtry <dbl>
1	2

1 row

```
y_hat_rf1 <- predict(train_rf1, test)
mean(y_hat_rf1 == test$Disease)
```

```
## [1] 0.7966102
```

```
varImp(train_rf1) # Variable importance shows that sex:female is the least important var
iable
```

```
## rf variable importance
##
##                            Overall
## Alkaline_Phosphotase        100.00
## Aspartate_Aminotransferase   92.06
## Alamine_Aminotransferase     85.89
## Age                          82.95
## Albumin                      66.08
## Direct_Bilirubin             59.53
## Albumin_and_Globulin_Ratio   48.31
## Sex1                          0.00
```

```
y_hat_rf1 <- as.matrix(y_hat_rf1) # Turned this into a vector for the very final step at
the end when creating an ensemble
colnames(y_hat_rf1) <- 'rf1'
```

```
# Tried removing the feature Sex, and see how it affects the accuracy
new_train <- train[,-1]
head(new_train)
```

	Direct_Bilirubin <dbl>	Alkaline_Phosphotase <dbl>	Alamine_Aminotransferase <dbl>	Aspartate_Amino
1	-0.4949862	-0.4284995	-0.35552499	-
2	1.4222880	1.6736358	-0.09349172	-
3	0.9252169	0.8155376	-0.11532783	-
4	-0.3884709	-0.4490282	-0.36644304	-
5	0.1796103	-0.3956537	-0.29547570	-
6	-0.2819557	-0.3422792	-0.33914791	-

6 rows | 1-5 of 9 columns

```
train_rf2 <- train(Disease ~ ., method = 'rf', tunegrid = data.frame(mtry = seq(1,7,1)),
data = new_train, ntree = 100)
y_hat_rf2 <- predict(train_rf2, test)
mean(y_hat_rf2 == test$Disease) # Can see that there is no change in accuracy, but hopef
ully getting rid of a feature would decrease the process time
```

```
## [1] 0.779661
```

```
y_hat_rf2 <- as.matrix(y_hat_rf2)
colnames(y_hat_rf2) <- 'rf2'
# Therefore I just decided to leave the feature sex and stick with rf1 instead of rf2
```

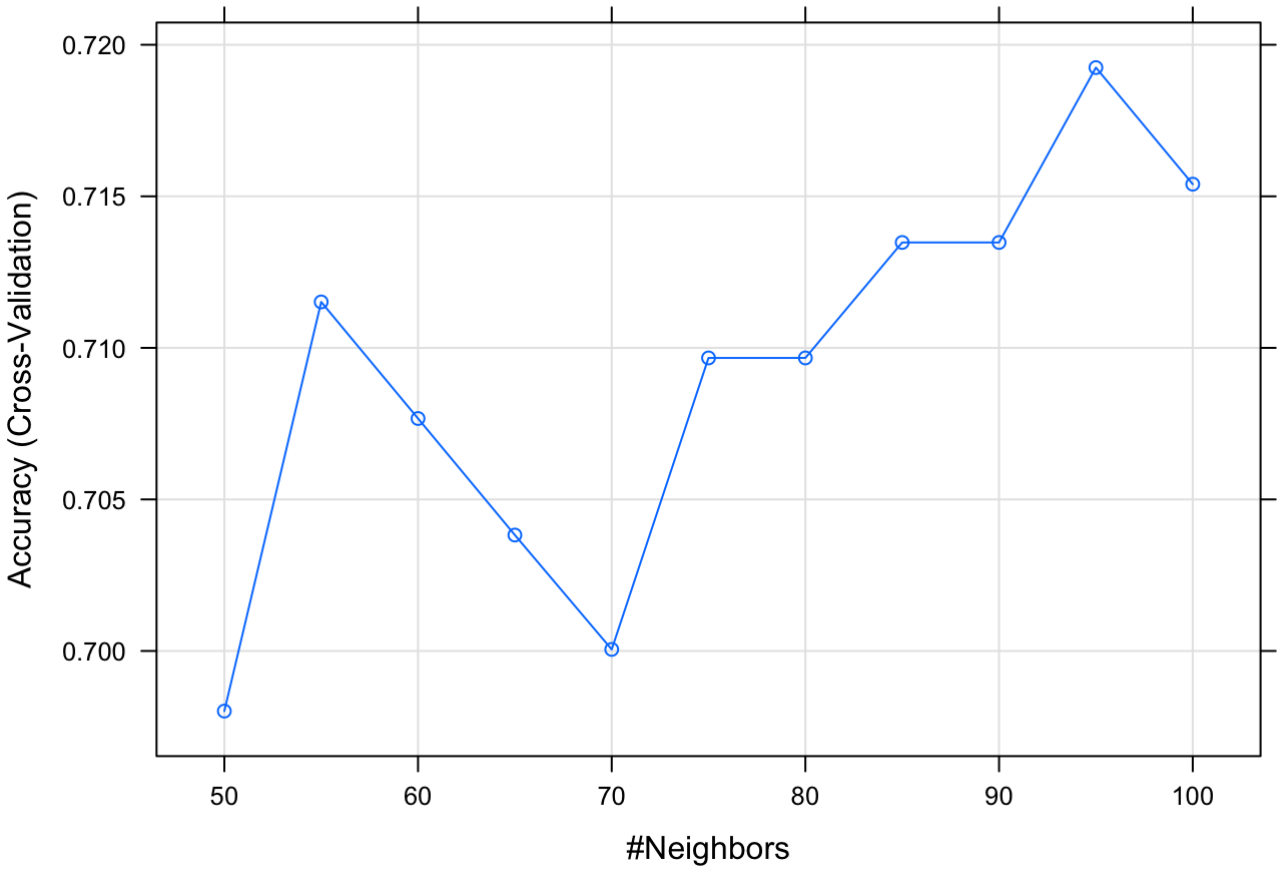
```
# Fit lda using train function in the caret package
set.seed(2)
train_lda <- train(Disease ~., method = "lda", data = new_train)
y_hat_lda <- predict(train_lda, test)
mean(y_hat_lda == test$Disease)
```

```
## [1] 0.7118644
```

```
y_hat_lda <- as.matrix(y_hat_lda)
colnames(y_hat_lda) <- "lda"
```

```
# Fit a new knn model with 10 fold cross validation, where each partition consists of 1
0% of the total data
set.seed(3)
train_knn <- train(Disease ~ .,
                   method = "knn",
                   data = new_train,
                   tuneGrid = data.frame(k = seq(50, 100, 5)), trControl = trainControl
(method = "cv", number = 10, p = 0.9))

plot(train_knn) # Plotted k against the accuracy for various tuning parameters
```




```
train_knn$bestTune # Shows the besttuned k (100)
```

	k
	<dbl>
10	95

1 row

```
# Initially the tuning parameters were set as seq(3, 51, 3), but once I plotted the accu
racy(bootstrap), against all the k's I realized that the tuning parameter could be optim
ized even more, once it is set higher. Therefore I maximized the accuracy when I put k =
100.

y_hat_knn <- predict(train_knn, test)
mean(y_hat_knn == test$Disease)
```

```
## [1] 0.7118644
```

```
y_hat_knn <- as.matrix(y_hat_knn)
colnames(y_hat_knn) <- 'knn'
```

```
# Fit a Naive Bayes Model
set.seed(9)
library(naivebayes)
train_naive <- train(Disease ~., method = 'naive_bayes', data = new_train)
y_hat_naive <- predict(train_naive, test)
mean(y_hat_naive == test$Disease)
```

```
## [1] 0.7288136
```

```
y_hat_naive <- as.matrix(y_hat_naive)
colnames(y_hat_naive) <- 'Naive_Bayes'
```

```
# Fit a Gam Loess Model
set.seed(10)
train_gam <- train(Disease ~., method = 'gamLoess', data = new_train)
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.5112
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.4757
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6256
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.92572
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 5.7978
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 4.8935
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6258
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6258
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval 2.9725
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : upperlimit 2.2425
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval 2.3432
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : upperlimit 2.2425
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval 2.9725
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : upperlimit 2.2425
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 8.7283
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 8.4996
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 10.475
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 8.4996
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.93012
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 10.475
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 8.7739
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6256
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.92572
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 2.8253
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 2.9817
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 2.6688
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 2.6688
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 2.6688
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 2.4185
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 2.8253
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 5.7978
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 4.8591
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 2.3778
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.5112
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6258
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 5.5711
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 16.624
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 9.8336
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval 2.9725
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : upperlimit 2.369
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval 2.9725
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : upperlimit 2.369
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 5.7978
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 3.0068
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 4.8591
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 3.0068
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.5112
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.4757
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval 2.9725
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : upperlimit 2.369
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval 2.9725
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : upperlimit 2.369
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 8.7283
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 6.4148
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 8.4553
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 6.4148
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 10.475
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 6.4148
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 6.9268
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 6.4148
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 9.7829
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 4.8199
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 5.1392
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 4.8199
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 16.624
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 4.8199
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 6.5882
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.0231
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.0231
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.92273
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 5.7978
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 3.0068
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 4.8591
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 3.0068
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 10.475
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 8.7738
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.88834
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.85997
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.85997
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval -2.8177
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : lowerlimit -2.2142
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval -2.8177
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : lowerlimit -2.2142
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : eval -2.6919
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : lowerlimit -2.2142
```

```
## Warning in gam.lo(data[["lo(Albumin, span = 0.5, degree = 1)"]], z, w, span
## = 0.5, : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.5112
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.93012
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5363
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5363
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.88834
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.85997
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.85997
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 16.624
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 9.8336
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.93012
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5363
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5363
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6256
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.92572
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 16.624
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 9.8336
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6258
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 16.624
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 9.8336
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.88834
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.85558
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6252
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.85558
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.5112
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.88834
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.85997
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.85997
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5363
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5363
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 10.475
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 8.7739
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 9.7829
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 5.1666
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 16.624
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 5.1666
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.5112
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.4757
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.0367
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 5.5711
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 10.475
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 8.7739
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 9.7829
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 5.1666
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 16.624
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 5.1666
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 5.5711
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame,
## bf.maxit, : lo.wam convergence not obtained in 30 iterations
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : eval 5.7978
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : upperlimit 4.8935
```

```
## Warning in gam.lo(data[["lo(Albumin_and_Globulin_Ratio, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 4.5112
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 5.5409
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## eval 6.464
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## upperlimit 4.5006
```

```
## Warning in gam.lo(data[["lo(Direct_Bilirubin, span = 0.5, degree = 1)"]], :
## extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 8.7283
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 8.4996
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 10.475
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 8.4996
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored

## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 8.7283
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 6.9633
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 8.4553
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 6.9633
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : eval 10.475
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : upperlimit 6.9633
```

```
## Warning in gam.lo(data[["lo(Alamine_Aminotransferase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 9.7829
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 5.1666
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : eval 16.624
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : upperlimit 5.1666
```

```
## Warning in gam.lo(data[["lo(Aspartate_Aminotransferase, span = 0.5, degree
## = 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.88834
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.83495
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval 7.4668
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : upperlimit 6.6251
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : eval -0.93761
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : lowerlimit -0.83495
```

```
## Warning in gam.lo(data[["lo(Alkaline_Phosphotase, span = 0.5, degree =
## 1)"]], : extrapolation not allowed with blending
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
y_hat_gam <- predict(train_gam, test)
y_hat_gam <- as.matrix(y_hat_gam)
colnames(y_hat_gam) <- 'Gam_Loess'
mean(y_hat_gam == test$Disease)
```

```
## [1] 0.7627119
```

```
#Fit a glm model
set.seed(1996)
train_glm <- train(Disease ~. ,method = 'glm', data = new_train)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
y_hat_glm <- predict(train_glm, test)
mean(y_hat_glm == test$Disease) #accuracy
```

```
## [1] 0.7627119
```

```
y_hat_glm <- as.matrix(y_hat_glm)
colnames(y_hat_glm) <- 'GLM'
```

```
# Train multinom, adaboost and svmlinear models at once
set.seed(11)
models <- c("multinom", "adaboost", "svmLinear")

ensemble <- lapply(models, function(model){
    print(model)
    train(Disease ~ ., method = model, data = train)
})
```

```
## [1] "multinom"
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 260.945125
## iter  20 value 254.346160
## iter  20 value 254.346160
## final  value 254.346160
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 261.649886
## final  value 255.951164
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 260.946321
## iter  20 value 254.347933
## iter  20 value 254.347933
## final  value 254.347933
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 268.213276
## final  value 265.019807
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 268.703241
## final  value 265.867797
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 268.213777
## final  value 265.020717
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 264.211543
## final  value 260.560796
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 264.819081
## final  value 261.601501
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 264.212161
## final  value 260.561902
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.583064
```

```
## final  value 264.036965
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 268.095633
## final  value 264.792210
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.583582
## final  value 264.037749
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 266.898372
## final  value 264.670225
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.354819
## final  value 265.327270
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 266.898842
## final  value 264.670916
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 258.660614
## iter  20 value 253.170367
## iter  20 value 253.170366
## final  value 253.170366
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 259.216916
## iter  20 value 254.221564
## iter  20 value 254.221564
## final  value 254.221564
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 258.661178
## iter  20 value 253.171506
## iter  20 value 253.171505
## final  value 253.171505
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 253.234065
## final  value 247.555080
## converged
## # weights:  10 (9 variable)
```

```
## initial  value 360.436534
## iter  10 value 254.321014
## final  value 249.009465
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 253.235204
## final  value 247.556669
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 280.851870
## final  value 277.633313
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 281.655166
## final  value 278.688189
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 280.852594
## final  value 277.634459
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.175875
## iter  20 value 261.779878
## iter  20 value 261.779878
## final  value 261.779878
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.812928
## final  value 262.878244
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.176520
## iter  20 value 261.781067
## iter  20 value 261.781067
## final  value 261.781067
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 246.706059
## iter  20 value 238.793571
## iter  20 value 238.793570
## final  value 238.793570
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 247.579147
## final  value 240.500266
```

```
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 246.706996
## iter  20 value 238.795501
## iter  20 value 238.795501
## final  value 238.795501
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 254.400120
## final  value 253.718400
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 255.266926
## final  value 254.634065
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 254.401012
## final  value 253.719378
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 264.769884
## final  value 259.175119
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 265.530624
## final  value 260.628803
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 264.770687
## final  value 259.176745
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 271.570123
## iter  20 value 263.299329
## iter  30 value 263.297000
## final  value 263.296996
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 272.172869
## iter  20 value 264.409453
## iter  30 value 264.408647
## final  value 264.408639
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
```

```
## iter  10 value 271.570763
## iter  20 value 263.300521
## iter  30 value 263.298195
## final  value 263.298191
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 249.148263
## iter  20 value 243.848185
## iter  30 value 243.846273
## final  value 243.846270
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 250.458194
## iter  20 value 245.384570
## iter  30 value 245.383891
## final  value 245.383886
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 249.149516
## iter  20 value 243.849950
## iter  30 value 243.848041
## final  value 243.848038
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 273.754947
## iter  20 value 267.970729
## final  value 267.970713
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 274.478527
## iter  20 value 269.117238
## iter  20 value 269.117238
## final  value 269.117238
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 273.755731
## iter  20 value 267.971957
## final  value 267.971942
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 253.129586
## final  value 244.992335
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 253.610011
## final  value 246.510116
```

```
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 253.130074
## final  value 244.994037
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 261.437087
## final  value 256.851041
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 261.987744
## final  value 257.877009
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 261.437658
## final  value 256.852148
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 258.914583
## final  value 257.515987
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 259.646619
## final  value 258.496741
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 258.915327
## final  value 257.517059
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.446758
## final  value 263.725530
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 268.067544
## final  value 264.573855
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.447375
## final  value 263.726435
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 275.427000
```

```
## final  value 267.230187
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 275.983643
## final  value 269.284781
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 275.427571
## final  value 267.232752
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 242.316769
## final  value 239.081472
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 243.123422
## final  value 240.307641
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 242.317587
## final  value 239.082791
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 263.389153
## final  value 257.738929
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 264.093626
## final  value 259.182288
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 263.389908
## final  value 257.740569
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 262.057691
## final  value 259.141502
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 262.770654
## final  value 260.286756
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
```

```
## iter  10 value 262.058418
## final  value 259.142733
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 266.051076
## final  value 265.653583
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 267.792262
## final  value 267.189299
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 266.053168
## final  value 265.655520
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 264.492908
## iter  20 value 257.793373
## final  value 257.793367
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 262.929347
## iter  20 value 258.910127
## iter  20 value 258.910126
## final  value 258.910126
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 264.494230
## iter  20 value 257.794608
## final  value 257.794601
## converged
## # weights:  10 (9 variable)
## initial  value 360.436534
## iter  10 value 261.804764
## final  value 259.801183
## converged
## [1] "adaboost"
## [1] "svmLinear"
```

```
names(ensemble) <- models # Wrote a function that trains multiple models at once

#create a prediction matrix of those three trained models
predict_all <- function(model){
  predict(model, test)
}
pred <- sapply(ensemble, predict_all)
pred_matrix <- as.matrix(pred)
pred_matrix
```

```
##       multinom adaboost svmLinear
##  [1,] "1"      "1"      "1"
##  [2,] "1"      "1"      "1"
##  [3,] "1"      "1"      "1"
##  [4,] "1"      "1"      "1"
##  [5,] "1"      "1"      "1"
##  [6,] "1"      "1"      "1"
##  [7,] "1"      "1"      "1"
##  [8,] "1"      "1"      "1"
##  [9,] "1"      "1"      "1"
## [10,] "1"      "0"      "1"
## [11,] "1"      "0"      "1"
## [12,] "1"      "1"      "1"
## [13,] "1"      "1"      "1"
## [14,] "1"      "1"      "1"
## [15,] "0"      "1"      "1"
## [16,] "1"      "1"      "1"
## [17,] "1"      "1"      "1"
## [18,] "1"      "1"      "1"
## [19,] "1"      "0"      "1"
## [20,] "1"      "1"      "1"
## [21,] "1"      "1"      "1"
## [22,] "0"      "1"      "1"
## [23,] "0"      "0"      "1"
## [24,] "1"      "1"      "1"
## [25,] "1"      "1"      "1"
## [26,] "1"      "1"      "1"
## [27,] "0"      "0"      "1"
## [28,] "1"      "1"      "1"
## [29,] "1"      "1"      "1"
## [30,] "1"      "1"      "1"
## [31,] "1"      "1"      "1"
## [32,] "1"      "0"      "1"
## [33,] "1"      "1"      "1"
## [34,] "0"      "1"      "1"
## [35,] "1"      "1"      "1"
## [36,] "1"      "1"      "1"
## [37,] "0"      "1"      "1"
## [38,] "0"      "1"      "1"
## [39,] "1"      "1"      "1"
## [40,] "1"      "1"      "1"
## [41,] "1"      "1"      "1"
## [42,] "1"      "1"      "1"
## [43,] "0"      "1"      "1"
## [44,] "1"      "1"      "1"
## [45,] "1"      "1"      "1"
## [46,] "1"      "1"      "1"
## [47,] "1"      "0"      "1"
## [48,] "1"      "1"      "1"
## [49,] "1"      "1"      "1"
## [50,] "1"      "1"      "1"
## [51,] "1"      "1"      "1"
## [52,] "1"      "1"      "1"
```

```
## [53,] "1"      "1"      "1"
## [54,] "1"      "1"      "1"
## [55,] "1"      "1"      "1"
## [56,] "1"      "1"      "1"
## [57,] "1"      "0"      "1"
## [58,] "1"      "1"      "1"
## [59,] "1"      "1"      "1"
```

```
#Show accuracy for each ML algorithm
colMeans(pred_matrix == test$Disease)
```

```
##  multinom  adaboost svmLinear
## 0.8135593 0.7457627 0.7118644
```

```
#Combine all the good models
final_ensemble <- cbind(pred_matrix[,2], y_hat_gam, y_hat_rf2, y_hat_glm) #ensemble mode
l was designed with the models that obtained top 4 highest accuracies

#I included all the models that I have created in this project at first, in the final en
semble and created many different subsets of ensemble which yielded me the highest accur
acy, which happened to be models with top 4 accuracies

one <- rowMeans(final_ensemble == 1)
y_hat_final_ensemble <- ifelse(one > 0.5, 1, 0)
#If the proportion of 1's in a row (each observation) comparing the predictions of all t
he models in the ensemble go over 0.5, then we say that the ensemble model has predicted
the patient to have a liver disease

mean(y_hat_final_ensemble == test$Disease) #calculating the accuracy of my ensemble mode
l
```

```
## [1] 0.7966102
```

In terms of the data set, before the models were built two variables “Total Protien,” and “Total Bilirubins” were excluded through the preprocessing procedure using correlations between the features. Specifically predictors that caused colinearity were taken out even before the data partitions were created. In terms of the model performance, individual model which has performed the best was rf2, which was trained by using random forest method. After modelling my first random forest model, I used the varImp function in order to take a look at variables of less importance when it came to effects on our dependent variable, which turned out to be sex:female. Although after removing the sex feature, the accuracy still stayed the same for my second random forest model, I proceeded since I decided that removing one more feature would shorten the time that it takes to train models. Once all the models were trained, an ensemble model consisting of glm, random forest, gam loess, and adaboost models (these models had allowed me to obtain the highest accuracies) were created. The decision making threshold was set to 0.5 in terms of the predictions. The highest accuracy was obtained which was around 80 percent. This was the highest accuracy I could obtain among any individual model or any subset of the ensembles.

Throughout this edx project, I had the opportunity to explore through an actual data off the edx learning platform without anyone’s help. It was a great practice to brush up on skills such as data wrangling, exploration, critical thinking and builind models that could predict outcomes. I have attempted to come up with an algorithm, given

traits and features of patients in India, which could predict whether a patient has a liver disease or not. The limitation that existed during the course of the project was the small size of the dataset. If there were more datas and more observations to train my models on, I think that I could have obtained models with higher accuracies. This project could have potentially a positive impact on doctors, in a sense that this algorithm (there are still room for improvement in terms of accuracy) could lift the burdens off the doctors' shoulders and expedit their processes of diagnosing patients with liver diseases. Instead of limiting to patients in India, with improvement, this algorithm could be applied to patients globally and has the potential to have a positive impact in the field of medical sciences.