P8106_midterm

yimin chen yc4195

2023-03-26

```r
library(tidyverse)
library(caret)
library(mgcv)
library(patchwork)
library(rpart)
library(rpart.plot)
library(party)
library(randomForest)
library(ranger)
library(e1071)
library(pdp)
library(earth)
library(splines)
library(pdp)
library(ggplot2)
library(gridExtra)
library(glmnet)
library(MASS)
library(pROC)
library(vip)
library(AppliedPredictiveModeling)
library(klaR)
```

import and Data

```r
set.seed(4195)
load("recovery.Rdata")
dat <- dat[sample(1:10000, 2000),]
dat <- dat[,-1]%>%
    mutate(study = case_when( # from character variable to a numeric variable
    study == "A" ~ 1,
    study == "B" ~ 2,
    study == "C" ~ 3)) %>%
    mutate(
    gender = as.factor(gender),
    race = as.factor(race),
    smoking = as.factor(smoking),
    hypertension = as.factor(hypertension),
    diabetes = as.factor(diabetes),
    vaccine = as.factor(vaccine),
    severity = as.factor(severity),
    study = as.factor(study)
  )
skimr::skim_without_charts(dat)
```

Table 1: Data summary

Name	dat
Number of rows	2000
Number of columns	15
Column type frequency:	
factor	8
numeric	7
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
gender	0	1	FALSE	2	0: 1009, 1: 991
race	0	1	FALSE	4	1: 1246, 3: 439, 4: 221, 2: 94
smoking	0	1	FALSE	3	0: 1209, 1: 583, 2: 208
hypertension	0	1	FALSE	2	0: 1030, 1: 970
diabetes	0	1	FALSE	2	0: 1722, 1: 278
vaccine	0	1	FALSE	2	1: 1189, 0: 811
severity	0	1	FALSE	2	0: 1804, 1: 196
study	0	1	FALSE	3	2: 1152, 3: 432, 1: 416

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
age	0	1	60.08	4.58	45.0	57.00	60.0	63.0	77.0
height	0	1	170.10	5.89	151.4	166.20	170.3	174.1	189.3
weight	0	1	79.92	6.99	57.5	75.00	79.9	84.7	104.2
bmi	0	1	27.67	2.65	19.7	25.80	27.6	29.4	37.1
SBP	0	1	130.43	8.06	104.0	125.00	130.0	136.0	156.0
LDL	0	1	110.30	20.22	47.0	96.75	110.0	124.0	172.0
recovery_time	0	1	41.83	27.05	2.0	28.00	38.5	49.0	365.0

Data

This project uses the "recovery.RData" file, which consists of 10000 participants. A random sample of 2000 participants is used for this analysis using a seed set to my UNI number (2183).

Split the data into training (70%) and test (30%) sets

Create x and y matrices for modeling

```
ctrl <- trainControl(method = "repeatedcv",
                     repeats = 5,
                     number=10)
set.seed(4195)
traindata <- createDataPartition(dat$recovery_time, p = 0.7, list = FALSE)
traindataset <- dat[traindata,]
testdataset <- dat[-traindata,]
#train
x = model.matrix(recovery_time ~ ., dat)[traindata,-1]
y = traindataset$recovery_time

#test
x_test = model.matrix(recovery_time ~ ., dat)[-traindata,-1]
y_test = testdataset$recovery_time
```

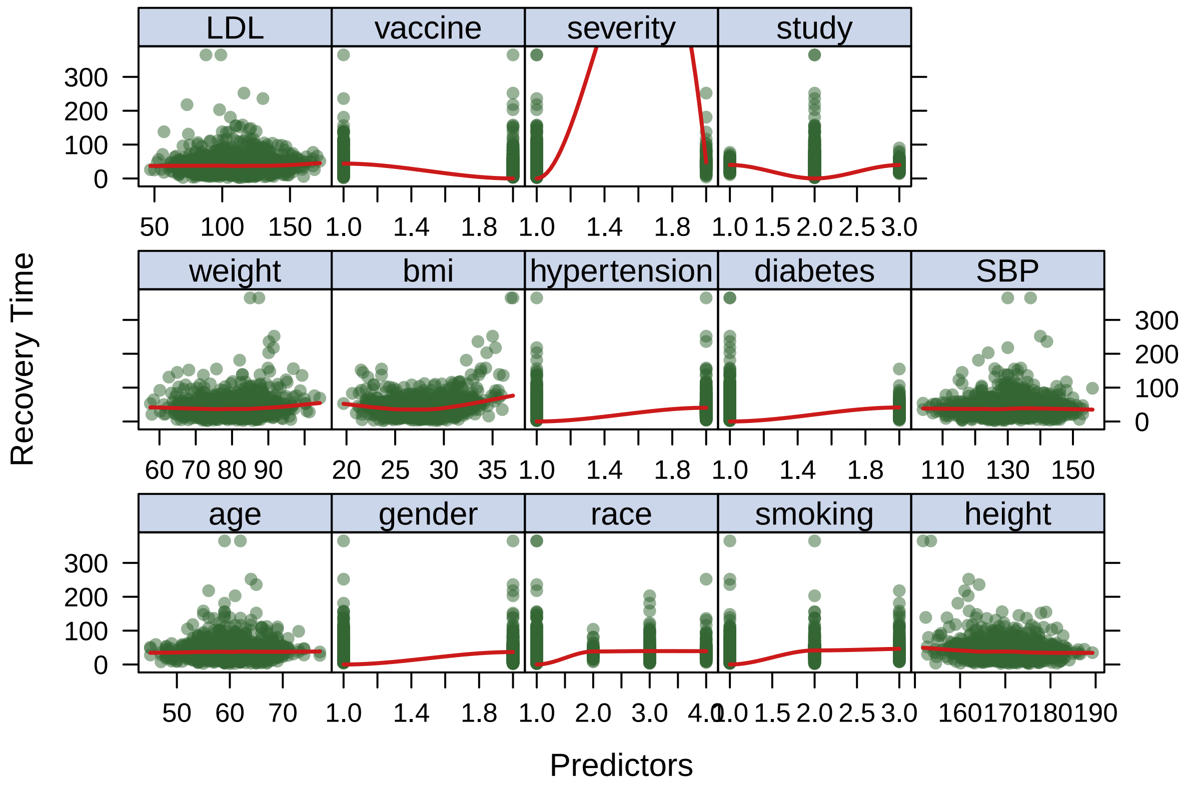
Exploratory analysis and data visualization:

In this section, use appropriate visualization techniques to explore the dataset and identify any patterns or relationships in the data.

```
# create dataset for exploratory analysis and data visualization
traindataset1 <- traindataset%>%
    mutate(
    gender = as.numeric(gender),
    race = as.numeric(race),
    smoking = as.numeric(smoking),
    hypertension = as.numeric(hypertension),
    diabetes = as.numeric(diabetes),
    vaccine = as.numeric(vaccine),
    severity = as.numeric(severity),
    study = as.numeric(study))
# Find the remaining non-numeric columns
#non_numeric_cols <- sapply(traindataset1, function(x) !is.numeric(x))
#non_numeric_cols
# Convert non-numeric columns to numeric
#traindataset1[, non_numeric_cols] <- lapply(traindataset1[, non_numeric_cols], as.numeric) # turn fact
```

```
theme1 = trellis.par.get()
theme1$plot.symbol$col = rgb(.2, .4, .2, .5)
theme1$plot.symbol$pch = 16
theme1$plot.line$col = rgb(.8, .1, .1, 1)
theme1$plot.line$lwd = 2
theme1$strip.background$col = rgb(.0, .2, .6, .2)
trellis.par.set(theme1)

## Exploratory analysis and visualization
featurePlot(x = traindataset1[ ,1:14],
            y = traindataset1[ ,15],
            plot = "scatter",
            span = .5,
            #layout=c(4,4),
            labels = c("Predictors", "Recovery Time"),
            type = c("p", "smooth"))
```



The following code creates lattice plots to visualize the the multivariate data. A plot is created for each of the 14 predictors in the dataset in order to visualize each predictor's association with the outcome, `recovery_time` (COVID-19 recovery time).

[Based on the lattice plots above, the following patterns in the data can be observed:]

Model training:

In this section, describe the models you used for predicting time to recovery from COVID-19. State the assumptions made by using the models. Provide a detailed description of the model training procedure and how you obtained the final model.

Less flexible models:

Fit a linear model:

```
set.seed(4195)
# Fit a linear regression model using cross-validation on the training dataset
lm.fit <- train(recovery_time ~ age + gender + race + smoking + height + weight +
                 bmi + hypertension + diabetes + SBP + LDL + vaccine +
                 severity + study,
               data = traindataset,
               method = "lm",
               trControl = ctrl)
# model summary
summary(lm.fit$finalModel)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -82.006 -12.582  -0.872   9.012 235.079
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.580e+03  1.888e+02 -13.666  < 2e-16 ***
## age            2.584e-01  1.551e-01   1.666  0.09600 .
## gender1       -5.893e+00  1.245e+00  -4.733 2.44e-06 ***
## race2         -4.165e+00  2.869e+00  -1.452  0.14676
## race3          3.060e-01  1.556e+00   0.197  0.84412
## race4         -7.088e-01  2.041e+00  -0.347  0.72838
## smoking1       4.534e+00  1.398e+00   3.242  0.00122 **
## smoking2       1.021e+01  2.083e+00   4.900 1.07e-06 ***
## height         1.509e+01  1.111e+00  13.586  < 2e-16 ***
## weight        -1.639e+01  1.176e+00 -13.934  < 2e-16 ***
## bmi            4.892e+01  3.366e+00  14.534  < 2e-16 ***
## hypertension1  2.698e+00  2.066e+00   1.306  0.19168
## diabetes1     -3.749e-01  1.819e+00  -0.206  0.83674
## SBP            1.498e-02  1.356e-01   0.110  0.91205
## LDL           -4.738e-02  3.187e-02  -1.486  0.13739
## vaccine1      -6.875e+00  1.280e+00  -5.373 9.07e-08 ***
## severity1      8.467e+00  2.130e+00   3.975 7.40e-05 ***
## study2         2.775e+00  1.573e+00   1.764  0.07788 .
## study3        -9.477e-01  1.900e+00  -0.499  0.61805
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.25 on 1383 degrees of freedom
## Multiple R-squared:  0.2404, Adjusted R-squared:  0.2305
## F-statistic: 24.31 on 18 and 1383 DF,  p-value: < 2.2e-16
```

```
# RMSE
test_pred_lm <- predict(lm.fit, newdata = testdataset)
test_rmse_lm <- sqrt(mean((test_pred_lm -y_test)^2))
test_rmse_lm
```

```
## [1] 24.21453
```

/* We then use the train function from the caret package to fit a linear regression model to the training data using cross-validation with 10 folds. The trainControl function specifies the cross-validation method, and the method argument specifies the type of model to fit (in this case, linear regression). The resulting model object contains the final model and information about the cross-validation performance.

We can view the summary statistics for the final model using the summary function on the finalModel object within the model object. We can also use the predict function to generate predictions for the test set using the final model, and compute the root mean squared error (RMSE) between the predicted and actual recovery times on the test set.

This approach allows us to obtain a final model that has been trained using cross-validation, which can help to reduce overfitting and improve the generalization performance of the model on new, unseen data. */

Fit Ridge Regression

```
set.seed(4195)
ridge.fit = train(x, y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = 0,
                                         lambda = exp(seq(-1, 5, length = 100))),
                  trControl = ctrl)
ridge_pred = predict(ridge.fit$finalModel, newx = x_test, s = ridge.fit$bestTune$lambda, type = "respon
# model summary
summary(ridge.fit$finalModel)
```

```
##             Length Class      Mode
## a0           100   -none-     numeric
## beta        1900   dgCMatrix  S4
## df           100   -none-     numeric
## dim            2   -none-     numeric
## lambda       100   -none-     numeric
## dev.ratio    100   -none-     numeric
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## call           5   -none-     call
## nobs           1   -none-     numeric
## lambdaOpt      1   -none-     numeric
## xNames        19   -none-     character
## problemType    1   -none-     character
## tuneValue      2   data.frame list
## obsLevels      1   -none-     logical
## param          0   -none-     list
```

```
# RMSE
test_pred_ridge <- predict(ridge.fit, newdata = data.frame(x_test))
test_rmse_ridge <- sqrt(mean((test_pred_ridge -y_test)^2))
test_rmse_ridge
```

```
## [1] 26.41153
```

RMSE = 26.4115291

Lasso model:

```r
set.seed(4195)
lasso.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 1,
                                          lambda = exp(seq(-1, 5, length = 100))),
                   trControl = ctrl)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```r
# view the model summary
summary(lasso.fit$finalModel)
```

```
##             Length Class      Mode
## a0            93   -none-     numeric
## beta        1767   dgCMatrix  S4
## df            93   -none-     numeric
## dim            2   -none-     numeric
## lambda        93   -none-     numeric
## dev.ratio     93   -none-     numeric
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## call           5   -none-     call
## nobs           1   -none-     numeric
## lambdaOpt      1   -none-     numeric
## xNames        19   -none-     character
## problemType    1   -none-     character
## tuneValue      2   data.frame list
## obsLevels      1   -none-     logical
## param          0   -none-     list
```

```r
# view performance on the test set (RMSE)
lasso_pred <- predict(lasso.fit, newdata = data.frame(x_test))
test_rmse_lasso<- sqrt(mean((lasso_pred - y_test)^2))
test_rmse_lasso
```

```
## [1] 26.54542
```

RMSE = 26.5454163

Elastic net model:

```r
set.seed(4195)
enet.fit <- train(x, y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = seq(0, 1, length = 21),
                                         lambda = exp(seq(2, -2, length = 50))),
                  trControl = ctrl)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
# view the model summary
summary(enet.fit$finalModel)
```

```
##             Length Class      Mode
## a0            93   -none-     numeric
## beta        1767   dgCMatrix  S4
## df            93   -none-     numeric
## dim            2   -none-     numeric
## lambda        93   -none-     numeric
## dev.ratio     93   -none-     numeric
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## call           5   -none-     call
## nobs           1   -none-     numeric
## lambdaOpt      1   -none-     numeric
## xNames        19   -none-     character
## problemType    1   -none-     character
## tuneValue      2   data.frame list
## obsLevels      1   -none-     logical
## param          0   -none-     list
```

```
# view performance on the test set (RMSE)
enet_pred <- predict(enet.fit, data.frame(x_test))
test_rmse_enet <- sqrt(mean((enet_pred - y_test)^2))
test_rmse_enet
```

```
## [1] 25.34275
```

```
# calculate RMSE
sqrt(mean((enet_pred - y_test)^2))
```

```
## [1] 25.34275
```

RMSE = 25.3427474

Partial least squares model:

```
set.seed(4195)
pls.fit <- train(x, y,
                 method = "pls",
                 tuneGrid = data.frame(ncomp = 1:15),
                 trControl = ctrl,
                 preProcess = c("center", "scale"))
# view the model summary
summary(pls.fit$finalModel)
```

```
## Data:    X dimension: 1402 18
##  Y dimension: 1402 1
## Fit method: oscorespls
## Number of components considered: 12
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X           8.308    14.97    22.46    31.92    36.72    41.27    47.00
## .outcome   12.019    13.09    13.19    13.28    13.81    14.93    16.75
##           8 comps  9 comps  10 comps  11 comps  12 comps
## X           50.70    54.04     57.10     62.31     67.54
## .outcome    19.99    22.55     23.94     24.03     24.04
```

```
# view performance on the test set (RMSE)
pls_pred <- predict(pls.fit, data.frame(x_test))
test_rmse_pls <- sqrt(mean((pls_pred - y_test))^2)
test_rmse_pls
```

```
## [1] 1.030057
```

RMSE = 1.0300573

More flexible models:

Generalized additive model (GAM):

```
set.seed(4195)
# fit GAM using all predictors
gamall.fit <- train(x, y, # test dataset
                 method = "gam",
                 trControl = ctrl, # 10-fold CV
                 control = gam.control(maxit = 200)) # Adjusted due to failure to converge at default s
# fit GAM using selection specification
gamselect.fit <- train(x, y, # test dataset
                 method = "gam",
                 tuneGrid = data.frame(method = "GCV.Cp", select = c(TRUE)),
                 trControl = ctrl, # 10-fold CV
                 control = gam.control(maxit = 200))  # Adjusted due to failure to converge at default 

# view the model summary
summary(gamall.fit$finalModel)
```

```
## 
## Family: gaussian 
## Link function: identity 
## 
## Formula:
## .outcome ~ gender1 + race2 + race3 + race4 + smoking1 + smoking2 + 
##     hypertension1 + diabetes1 + vaccine1 + severity1 + study2 + 
##     study3 + s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) + 
##     s(weight) + s(id)
## 
## Parametric coefficients:
##                 Estimate Std. Error t value Pr(>|t|)    
## (Intercept)     43.29962    3.43307  12.613  < 2e-16 ***
## gender1         -6.10715    1.11574  -5.474 5.24e-08 ***
## race2         -2.34087    2.58851  -0.904  0.36598
## race3          1.03128    1.39772   0.738  0.46074
## race4         -1.35282    1.83575  -0.737  0.46129
## smoking1       5.10488    1.25560   4.066 5.06e-05 ***
## smoking2      11.22155    1.87219   5.994 2.62e-09 ***
## hypertension1  3.58200    1.12337   3.189  0.00146 **
## diabetes1      0.03979    1.63114   0.024  0.98054
## vaccine1      -7.01109    1.15269  -6.082 1.53e-09 ***
## severity1      9.98092    1.90908   5.228 1.98e-07 ***
## study2         0.95589    3.81305   0.251  0.80209
## study3        -0.36937    5.49720  -0.067  0.94644
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                 edf Ref.df      F p-value
## s(age)    3.345e-07      9  0.000  0.3837
## s(SBP)    1.250e-07      9  0.000  0.5334
## s(LDL)    3.389e-07      9  0.000  0.3758
## s(bmi)    8.912e+00      9 71.448  <2e-16 ***
## s(height) 6.042e-08      9  0.000  0.8946
## s(weight) 5.307e+00      9  1.286  0.0363 *
## s(id)     5.032e+00      9  0.762  0.2167
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.385   Deviance explained = 39.9%
## GCV = 442.21  Scale est. = 432.04    n = 1402
```

```
gamall.fit$bestTune
```

```
##   select method
## 2   TRUE GCV.Cp
```

```
gamall.fit$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender1 + race2 + race3 + race4 + smoking1 + smoking2 +
##     hypertension1 + diabetes1 + vaccine1 + severity1 + study2 +
##     study3 + s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) +
##     s(weight) + s(id)
##
## Estimated degrees of freedom:
## 0.00 0.00 0.00 8.91 0.00 5.31 5.03
##  total = 32.25
##
## GCV score: 442.2128
```

```r
# view performance on the test set (RMSE)
gamall_pred <- predict(gamall.fit, data.frame(x_test))
test_rmse_gamall <- sqrt(mean((gamall_pred - y_test))^2)
test_rmse_gamall
```

```
## [1] 0.4312508
```

```r
# calculate RMSE
sqrt(mean((gamall_pred - y_test))^2)
```

```
## [1] 0.4312508
```

```r
# view the model summary
summary(gamselect.fit$finalModel)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender1 + race2 + race3 + race4 + smoking1 + smoking2 +
##     hypertension1 + diabetes1 + vaccine1 + severity1 + study2 +
##     study3 + s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) +
##     s(weight) + s(id)
##
## Parametric coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    43.29962    3.43307  12.613  < 2e-16 ***
## gender1        -6.10715    1.11574  -5.474 5.24e-08 ***
## race2          -2.34087    2.58851  -0.904  0.36598
## race3           1.03128    1.39772   0.738  0.46074
## race4          -1.35282    1.83575  -0.737  0.46129
## smoking1        5.10488    1.25560   4.066 5.06e-05 ***
## smoking2       11.22155    1.87219   5.994 2.62e-09 ***
## hypertension1   3.58200    1.12337   3.189  0.00146 **
## diabetes1       0.03979    1.63114   0.024  0.98054
## vaccine1       -7.01109    1.15269  -6.082 1.53e-09 ***
## severity1       9.98092    1.90908   5.228 1.98e-07 ***
## study2          0.95589    3.81305   0.251  0.80209
## study3         -0.36937    5.49720  -0.067  0.94644
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                 edf Ref.df      F p-value
## s(age)    3.345e-07      9  0.000  0.3837
## s(SBP)    1.250e-07      9  0.000  0.5334
## s(LDL)    3.389e-07      9  0.000  0.3758
## s(bmi)    8.912e+00      9 71.448  <2e-16 ***
## s(height) 6.042e-08      9  0.000  0.8946
## s(weight) 5.307e+00      9  1.286  0.0363 *
## s(id)     5.032e+00      9  0.762  0.2167
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.385   Deviance explained = 39.9%
## GCV = 442.21  Scale est. = 432.04    n = 1402
```

```
gamselect.fit$bestTune
```

```
##   select method
## 1   TRUE GCV.Cp
```

```
gamselect.fit$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender1 + race2 + race3 + race4 + smoking1 + smoking2 +
##     hypertension1 + diabetes1 + vaccine1 + severity1 + study2 +
##     study3 + s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) +
##     s(weight) + s(id)
##
## Estimated degrees of freedom:
## 0.00 0.00 0.00 8.91 0.00 5.31 5.03
##  total = 32.25
##
## GCV score: 442.2128
```

```
# view performance on the test set (RMSE)
gamselect_pred <- predict(gamselect.fit, data.frame(x_test))
test_rmse_gamselect <- sqrt(mean((gamselect_pred - y_test))^2)
test_rmse_gamselect
```

```
## [1] 0.4312508
```

RMSE for gam_all= 0.4312508 RMSE for gam_select= 0.4312508

Multivariate adaptive regression spline (MARS) model:

```
set.seed(4195)
# create grid of all possible pairs that can take degree and nprune values
mars_grid <- expand.grid(degree = 1:3, # number of possible product hinge functions in 1 term
                         nprune = 2:20) # Upper bound of number of terms in model
mars.fit <- train(x,
                  y,# training dataset
                  method = "earth",
                  tuneGrid = mars_grid,
                  trControl = ctrl)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
# view the model summary
summary(mars.fit$finalModel)
```

```
## Call: earth(x=matrix[1402,18], y=c(30,118,49,41,...), keepxy=TRUE, degree=1,
##             nprune=11)
##
##             coefficients
## (Intercept)   -34.958407
## gender1        -5.877791
## smoking2        9.425191
## vaccine1       -6.573231
## severity1       9.512094
## h(bmi-23.9)     9.037401
## h(bmi-28.6)     5.225776
## h(32-bmi)       9.704445
## h(bmi-32)       7.002485
## h(bmi-34.3)   -51.894845
## h(bmi-35.2)   156.582406
##
## Selected 11 of 23 terms, and 5 of 18 predictors (nprune=11)
## Termination condition: Reached nk 37
## Importance: bmi, vaccine1, gender1, smoking2, severity1, age-unused, ...
## Number of terms at each degree of interaction: 1 10 (additive model)
## GCV 447.0747    RSS 608162.2    GRSq 0.3642685    RSq 0.3822897
```

```
# view performance on the test set (RMSE)
mars_pred <- predict(mars.fit, data.frame(x_test))
test_rmse_mars <- sqrt(mean((mars_pred - y_test))^2)
test_rmse_mars
```

```
## [1] 0.498946
```

RMSE = 0.498946

```
## Perform LDA using the training data
#set.seed(4195)
#lda.fit = lda(recovery_time~., data = dat,
               #subset = traindata)

#Plot the linear discriminants in LDA
#plot(lda.fit)
#lda.pred <- predict(lda.fit, newdata = testdataset)
#head(lda.pred$posterior)
#lda.fit$scaling
```

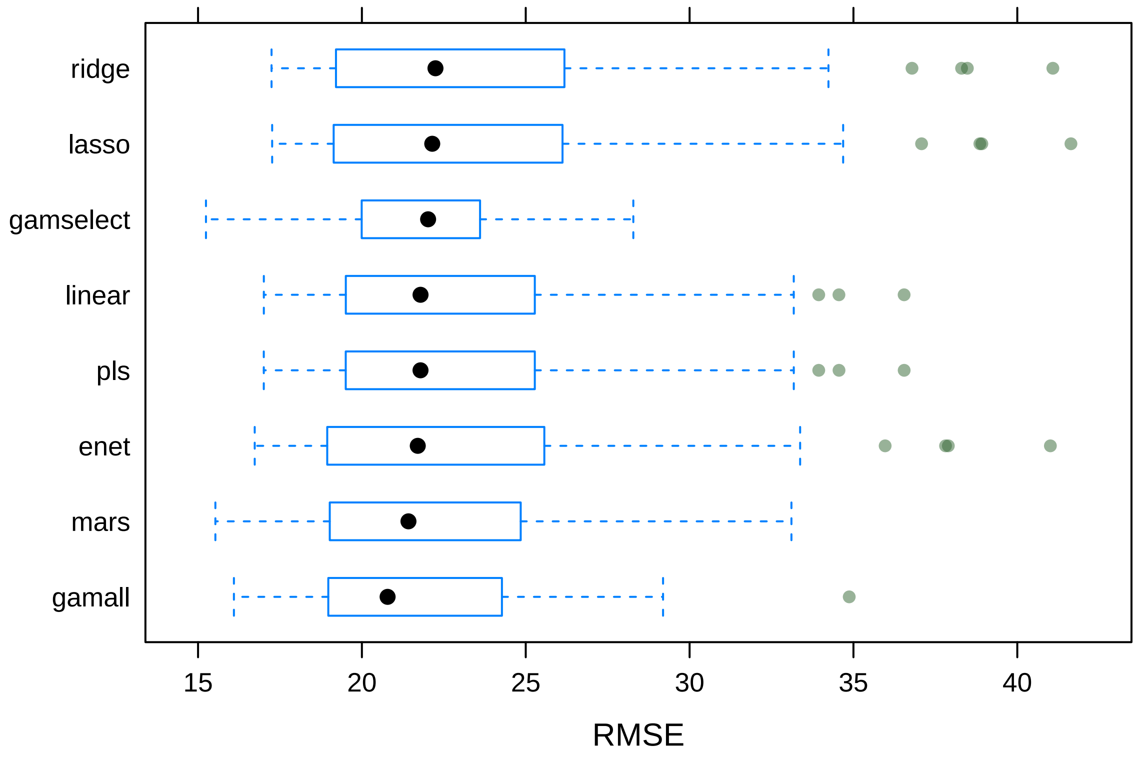
Model comparison:

Boxplot

```r
set.seed(4195)
resamp = resamples(list(linear=lm.fit,
                        ridge = ridge.fit,
                        lasso = lasso.fit,
                        enet = enet.fit,
                        pls = pls.fit,
                        gamall = gamall.fit,
                        gamselect = gamselect.fit,
                        mars = mars.fit))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: linear, ridge, lasso, enet, pls, gamall, gamselect, mars
## Number of resamples: 50
##
## MAE
##                Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## linear    12.45434 14.36326 15.46439 15.39960 16.07453 18.33422    0
## ridge     12.45054 14.18708 15.11421 15.26498 16.14132 18.84117    0
## lasso     12.33089 14.02611 15.12363 15.17466 16.07731 18.83506    0
## enet      12.22066 13.86494 14.89321 14.99975 15.88766 18.50572    0
## pls       12.45441 14.36526 15.46680 15.40030 16.07406 18.34113    0
## gamall    11.79423 13.81971 14.67363 14.61351 15.32388 17.21988    0
## gamselect 11.47791 13.65084 14.62087 14.55817 15.37338 17.19242    0
## mars      10.94788 13.88167 14.64942 14.54511 15.38748 17.06443    0
##
## RMSE
##                Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## linear    17.00778 19.53498 21.78932 23.17550 25.20443 36.54660    0
## ridge     17.24267 19.25478 22.24542 24.10238 26.04962 41.08614    0
## lasso     17.26112 19.17044 22.14641 24.19022 26.05641 41.63750    0
## enet      16.72887 19.02395 21.70502 23.57535 25.43814 41.01036    0
## pls       17.00605 19.53420 21.78871 23.17539 25.20385 36.54796    0
## gamall    16.09422 19.09326 20.78462 21.85334 24.26278 34.87183    0
## gamselect 15.24178 19.99991 22.02063 21.70571 23.57556 28.28298    0
## mars      15.52845 19.02786 21.42099 22.27648 24.80028 33.10796    0
##
## Rsquared
##                 Min.    1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## linear    0.06296364 0.14590446 0.1926016 0.2129520 0.2526136 0.4770537    0
## ridge     0.02287357 0.09666906 0.1271230 0.1341483 0.1662364 0.3152086    0
## lasso     0.01818920 0.08726159 0.1219181 0.1275187 0.1619502 0.3083602    0
## enet      0.04295509 0.13535499 0.1662425 0.1754240 0.2218866 0.3589363    0
## pls       0.06300174 0.14592884 0.1926212 0.2129513 0.2525843 0.4769383    0
## gamall    0.03495209 0.17587580 0.2515160 0.2930509 0.3521605 0.6627712    0
## gamselect 0.03604956 0.17793847 0.2608607 0.3064895 0.3591188 0.6680629    0
## mars      0.03294532 0.14345855 0.2403353 0.2747823 0.3541213 0.6495820    0
```

```r
bwplot(resamp, metric = "RMSE")
```



Results:

In this section, report the final model that you built for predicting time to recovery from COVID-19. Interpret the results. Assess the model's training/test performance.

Conclusions:

In this section, summarize your findings from the model analysis and discuss the insights gained into predicting time to recovery from COVID-19.