models_total_Candice

Candice Yu

2024-03-26

Contents

```r
library(caret)
library(earth)
library(tidyverse)
library(gridExtra)
```

Load the training/test set & control method

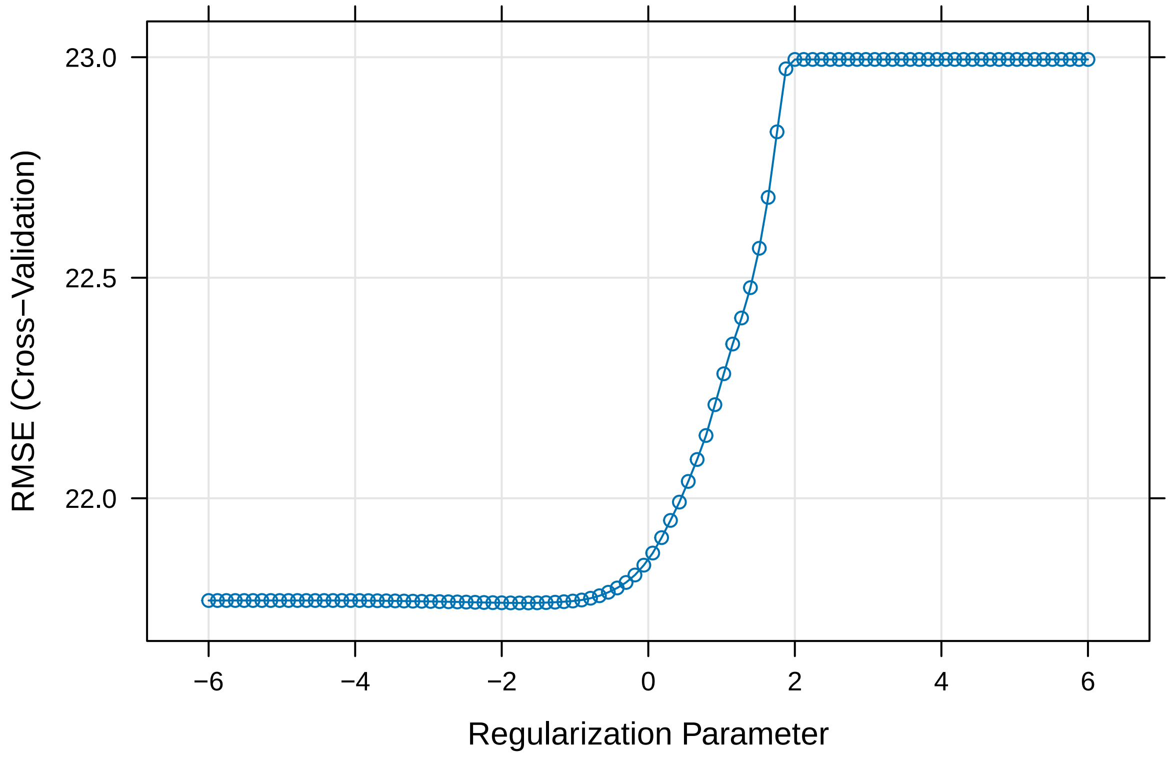
```r
# Load the training and test sets
train_data <- read.csv("./Data/train_data.csv")
test_data <- read.csv("./Data/test_data.csv")

# Load the control method
ctrl1 <- readRDS("./Data/train_control.rds")

# change variables to be factors again
train_data <- train_data %>%
  mutate(gender = as_factor(gender),
         diabetes = as_factor(diabetes),
         hypertension = as_factor(hypertension),
         vaccine = as_factor(vaccine),
         severity = as_factor(severity))

test_data <- test_data %>%
  mutate(gender = as_factor(gender),
         diabetes = as_factor(diabetes),
         hypertension = as_factor(hypertension),
         vaccine = as_factor(vaccine),
         severity = as_factor(severity))

# matrix of predictors
x <- train_data %>% select(-recovery_time)
y <- train_data$recovery_time

x_test <- test_data %>% select(-recovery_time)
y_test <- test_data$recovery_time
```

Model Training: Linear models

Lasso Regression Model

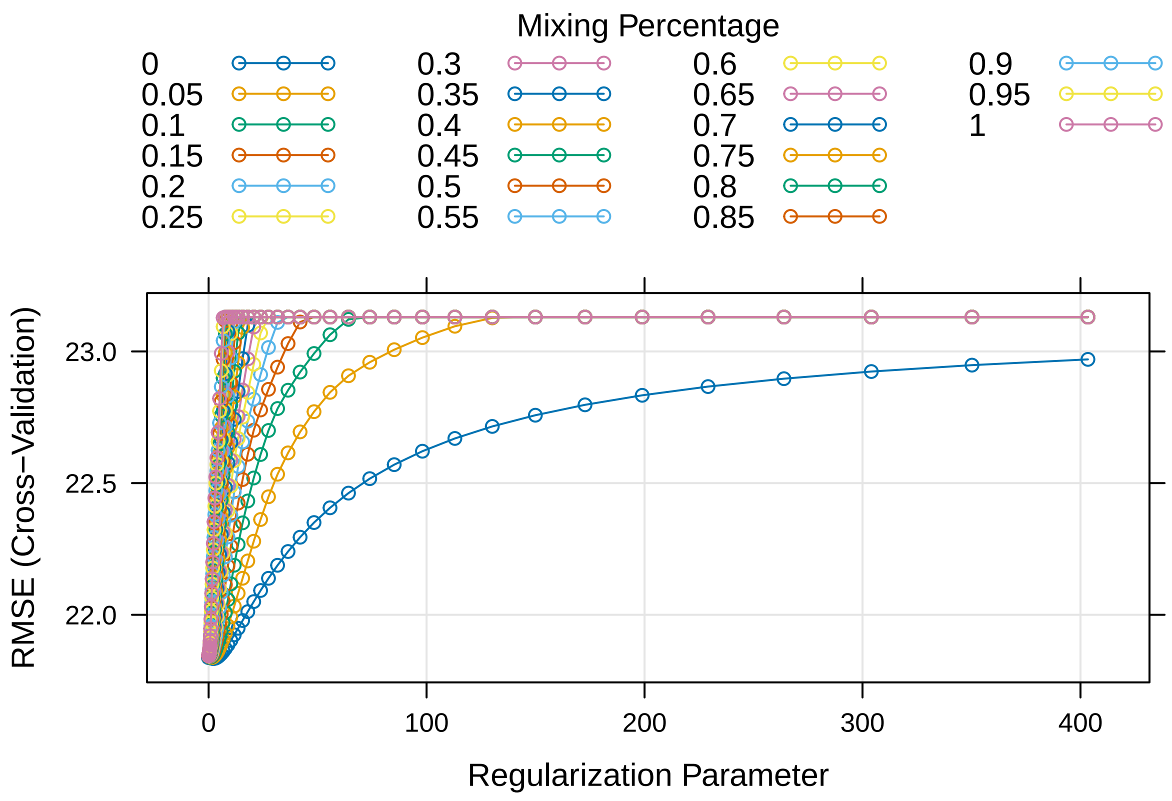
```r
lasso_grid <- expand.grid(
  alpha = 1,
  lambda = exp(seq(-6, 6, length.out = 100))
)

lasso_fit <- train(x, y,
  method = "glmnet",
  tuneGrid = lasso_grid,
  trControl = ctrl1
)
```

```r
plot(lasso_fit, xTrans = log)
```



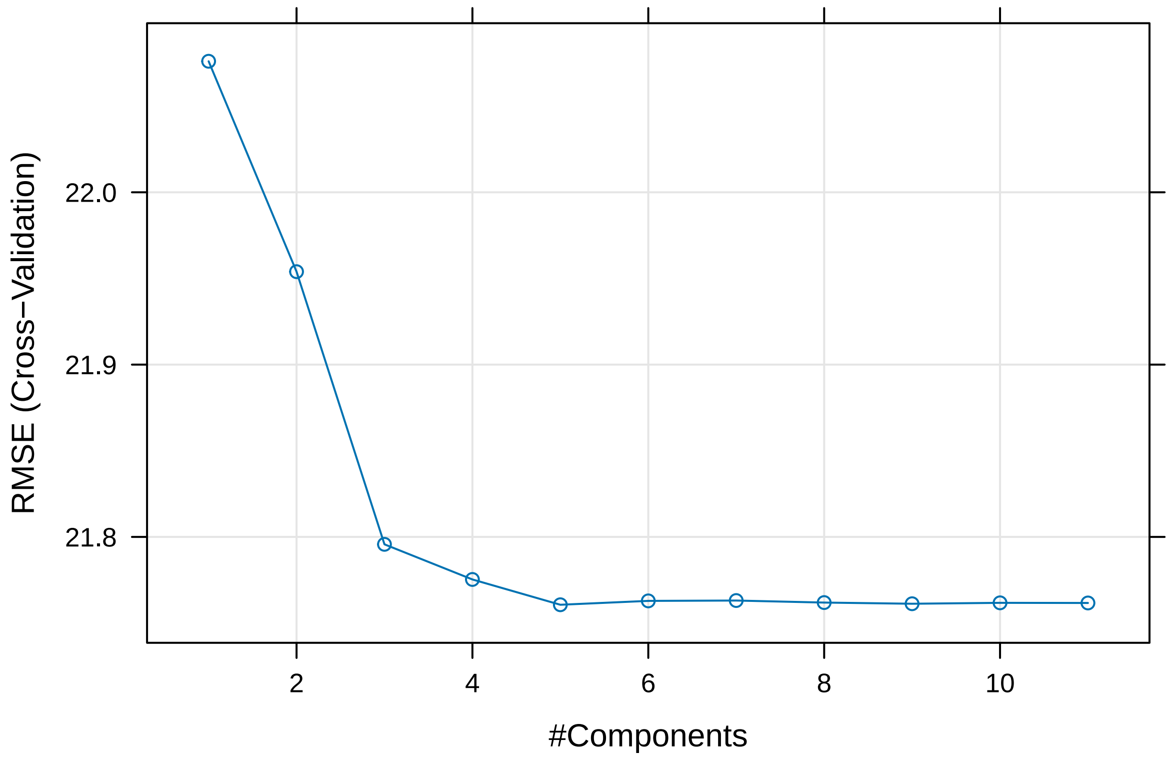
Elastic Net Model

```r
enet_grid <- expand.grid(
  alpha = seq(0, 1, length.out = 21),
  lambda = exp(seq(-8, 6, length.out = 100))
)

enet_fit <- train(x, y,
                  method = "glmnet",
                  tuneGrid = enet_grid,
                  trControl = ctrl1,
                  preProcess = c("center", "scale")
)
plot(enet_fit)
```



Partial Least Squares

```
pls_fit <- train(x, y,
                 method = "pls",
                 tuneLength = 20,
                 trControl = ctrl1,
                 preProcess = c("center", "scale")
                 )
plot(pls_fit)
```



Evaluate the performance of linear models

```r
lasso_pred <- predict(lasso_fit, newdata = x_test)
enet_pred <- predict(enet_fit, newdata = x_test)
pls_pred <- predict(pls_fit, newdata = x_test)

lasso_performance <- postResample(pred = lasso_pred, obs = test_data$recovery_time)
lasso_performance
```

```
##       RMSE   Rsquared        MAE
## 19.9896682  0.1042464 13.2470122
```

```r
enet_performance <- postResample(pred = enet_pred, obs = test_data$recovery_time)
enet_performance
```

```
##       RMSE   Rsquared        MAE
## 19.9495620  0.1063479 13.1938360
```

```r
pls_performance <- postResample(pred = pls_pred, obs = test_data$recovery_time)
pls_performance
```

```
##       RMSE   Rsquared        MAE
## 19.8669565  0.1177016 13.4284540
```

Model Training: Nonlinear Methods

The EDA plots show that the relationship between predictors and recovery time is likely non-linear, and there may be interactions between variables, especially considering the difference between study groups A and B.

Given the results from the EDA plots and the nature of the data, both generalized additive models (GAM) and multivariate adaptive regression splines (MARS) could be suitable choices for modeling. They both are capable of modeling complex, non-linear relationships in the data.

Multivariate Adaptive Regression Spline (MARS)

Build the MARS model

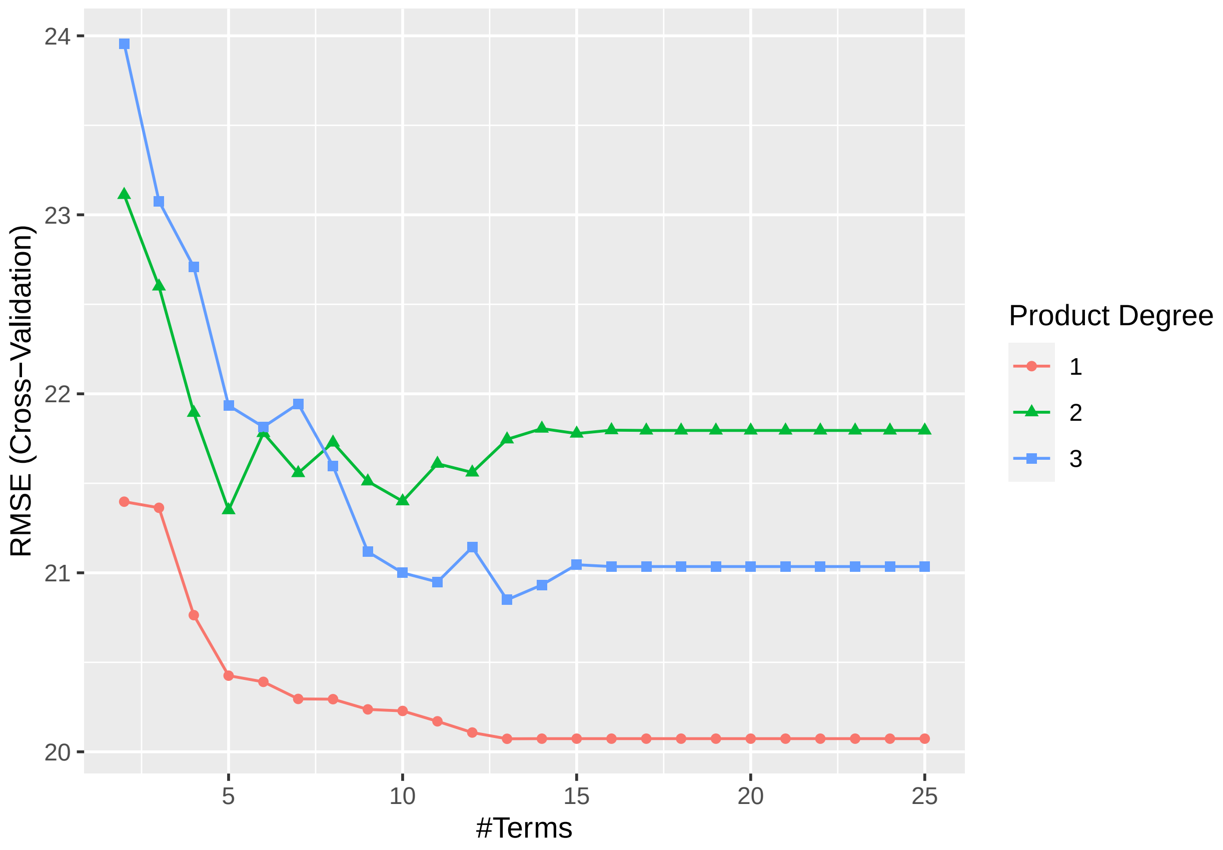
```
# train the MARS model
mars_grid <- expand.grid(degree = 1:3, nprune = 2:25)

set.seed(123) # set the same seed
mars_fit <- train(x, y,
                     method = "earth",
                     tuneGrid = mars_grid,
                     trControl = ctrl1)
```

MARS Model Summary

```
# Model summary
summary(mars_fit)
```

```
## Call: earth(x=data.frame[2402,12], y=c(29,34,41,50,3...), keepxy=TRUE,
##             degree=1, nprune=13)
##
##              coefficients
## (Intercept)    -17.553244
## gender1         -3.765001
## vaccine1        -6.264449
## severity1        7.704560
## studyB           4.340915
## h(1-smoking)    -2.851075
## h(bmi-23.9)      8.829273
## h(bmi-29.5)      5.982277
## h(31-bmi)        8.631275
## h(bmi-31)       -6.680835
## h(bmi-35.2)    152.393498
## h(bmi-35.7)   -140.684645
## h(137-SBP)      -0.225498
##
## Selected 13 of 16 terms, and 7 of 12 predictors (nprune=13)
## Termination condition: Reached nk 25
## Importance: bmi, vaccine1, severity1, studyB, gender1, SBP, smoking, ...
## Number of terms at each degree of interaction: 1 12 (additive model)
## GCV 373.804    RSS 879284.3    GRSq 0.3314631    RSq 0.3447615
```

```
ggplot(mars_fit)
```



```
mars_fit$bestTune
```

```
##    nprune degree
## 12     13      1
```

```
coef(mars_fit$finalModel)
```

```
##  (Intercept)    h(bmi-31)    h(31-bmi)  h(bmi-23.9)     vaccine1    severity1
##  -17.5532443   -6.6808345    8.6312754    8.8292727   -6.2644485    7.7045598
##       studyB      gender1  h(bmi-29.5)  h(137-SBP)  h(bmi-35.7)  h(bmi-35.2)
##    4.3409147   -3.7650013    5.9822775   -0.2254977 -140.6846454  152.3934981
## h(1-smoking)
##   -2.8510751
```

MARS Model Description:

The MARS model is a flexible regression method capable of uncovering complex nonlinear relationships between the dependent variable (recovery_time) and a set of independent variables. It does this by fitting piecewise linear regressions, which can adapt to various data shapes. This is particularly useful for modeling the recovery time from COVID-19 since the relationship between predictors and recovery time could be highly nonlinear and interaction-heavy.

Assumptions:

- The relationships between predictors and the response can be captured using piecewise linear functions.
- Interactions between variables can be important and are modeled by products of basis functions.
- There is no assumption of a parametric form of the relationship between predictors and the response.

Final Model Selection:

- The optimal hyperparameters were degree (degree of interaction) = 3 and nprune (number of terms)

= 16.
- The selected model terms involve interactions between patient characteristics, their biometrics, the specific study group they belong to, and some non-linear transformations of these variables.

Evaluate performance on the test set

```r
# Evaluate its performance on the test set:
predictions <- predict(mars_fit, newdata = test_data)
postResample(pred = predictions, obs = test_data$recovery_time)
```

```
##       RMSE   Rsquared        MAE
## 19.1868865  0.2104313 12.9248347
```

The results from evaluating the MARS model on the test set provide three key metrics:

1. **Root Mean Squared Error (RMSE):** RMSE measures the average magnitude of the prediction error. It represents the square root of the average squared differences between the predicted and actual values. An RMSE of 19.629 suggests that, on average, the model's predictions of the recovery time are about 19.629 days off from the actual recovery times.

2. **R-squared (R^2):** R^2 is a statistical measure that represents the proportion of the variance for the dependent variable that's explained by the independent variables in the model. In your case, the R^2 value is 0.2177, which means approximately 21.77% of the variance in the recovery time is explained by the model. This is a relatively low value, indicating that there is a lot of variability in the recovery time that is not captured by the model.

3. **Mean Absolute Error (MAE):** MAE measures the average absolute difference between the predicted values and the actual values, providing a linear score that reflects the average error magnitude without considering its direction. An MAE of 12.409 suggests that the model's predictions are, on average, 12.409 days different from the actual recovery time.

Interpretation

- The **RMSE** of 19.629 days is relatively high, depending on the context of the recovery times' range. If the typical recovery time is on the order of a few days, this is a substantial error. However, if recovery times are generally several weeks, the error may be more acceptable.

- The **R-squared** value of 0.2177 is not very high, suggesting that there might be other factors not included in the model that affect the recovery time. It also indicates that the relationship between the predictors and the recovery time has a significant amount of unexplained variability.

- The **MAE** gives us an indication that, despite the direction of the errors, the model's predictions are off by about two weeks on average. MAE is less sensitive to outliers than RMSE, so this value suggests that the model has a consistent average error across the test dataset.

set.seed(1)

GAM model

Build the GAM model

```r
set.seed(123) # set the same seed
gam_fit <- train(x = x, y = y,
                 method = "gam",
                 trControl = ctrl1)
```

Display the summary of the final model

```
gam_model_final <- gam_fit$finalModel
summary(gam_model_final)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + hypertension + diabetes + vaccine + severity +
##     study + smoking + race + s(age) + s(SBP) + s(LDL) + s(bmi)
##
## Parametric coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    43.0689     1.1322  38.039  < 2e-16 ***
## gender1        -3.6317     0.7936  -4.576 4.97e-06 ***
## hypertension1   3.2473     0.7998   4.060 5.06e-05 ***
## diabetes1      -1.2603     1.1111  -1.134 0.256797
## vaccine1       -6.3587     0.8101  -7.849 6.26e-15 ***
## severity1       8.1497     1.2720   6.407 1.78e-10 ***
## studyB          4.6462     0.8468   5.487 4.52e-08 ***
## smoking         1.9839     0.5779   3.433 0.000607 ***
## race           -0.1192     0.3699  -0.322 0.747330
## ---
## 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) 3.908e-07      9   0.000   0.510
## s(SBP) 1.737e-06      9   0.000   0.395
## s(LDL) 3.093e-01      9   0.049   0.231
## s(bmi) 8.115e+00      9 109.190  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.327   Deviance explained = 33.2%
## GCV = 378.61  Scale est. = 375.87    n = 2402
```

Evaluate the GAM model's performance

```
test_predictions <- predict(gam_model_final, x_test)
postResample(pred = test_predictions, obs = test_data$recovery_time)
```

```
##       RMSE   Rsquared        MAE
## 18.5198607  0.2440446 12.7977052
```

Random Forest

Build the rf model

```
# Parameters for Random Forest training
tunegrid <- expand.grid(mtry = 1:5)
```

```r
# build the rf model
rf_fit <- train(
   x = x, y = y,
   method = "rf",
   trControl = ctrl1,
   tuneGrid = tunegrid
 )
rf_model_final <- rf_fit$finalModel
```

Evaluate the rf model's performance

```r
# Calculate and print the RMSE for training and test datasets
rf_predictions <- predict(rf_model_final, x_test)
postResample(pred = rf_predictions, obs = test_data$recovery_time)
```

```
##       RMSE   Rsquared        MAE
## 17.9802667  0.2748427 12.0916634
```

Model Comparison

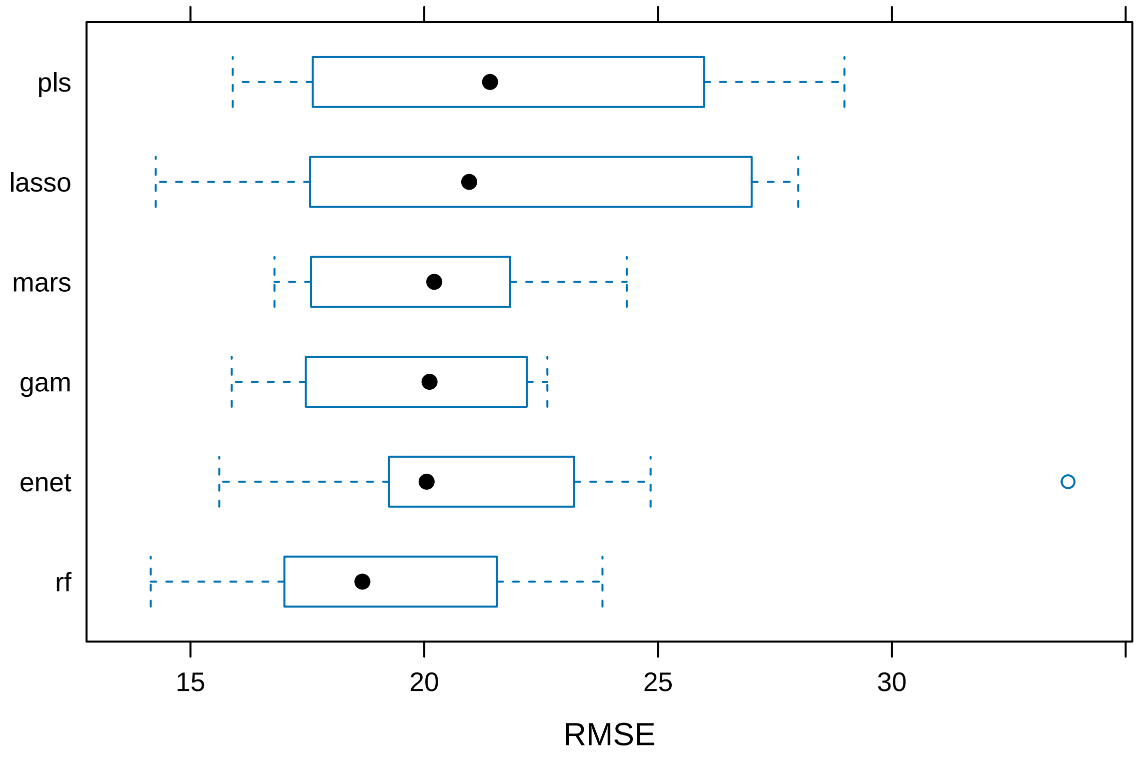
```r
resamp =
  resamples(list(lasso = lasso_fit,
                 gam = gam_fit,
                 enet = enet_fit,
                 pls = pls_fit,
                 mars = mars_fit,
                 rf = rf_fit))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: lasso, gam, enet, pls, mars, rf
## Number of resamples: 10
##
## MAE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lasso 11.04477 12.89252 13.51375 13.27118 13.95726 14.98760    0
## gam   11.46034 12.10115 12.62081 12.65627 13.19277 13.81414    0
## enet  11.70483 12.61601 13.18237 13.22244 13.43854 16.27790    0
## pls   11.86937 12.75490 13.11556 13.42258 14.32464 14.99726    0
## mars  11.83085 12.11632 12.55367 12.72646 13.34737 13.76872    0
## rf    10.45674 11.00215 12.25019 12.08675 13.08791 13.57029    0
##
## RMSE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lasso 14.25801 17.64976 20.96119 21.76276 26.92624 27.99875    0
## gam   15.88208 17.86226 20.11312 19.82547 22.04593 22.63304    0
## enet  15.61689 19.39761 20.05116 21.83425 23.10111 33.76814    0
## pls   15.90377 17.75838 21.40782 21.76064 25.74361 28.98678    0
## mars  16.79710 17.86670 20.21390 20.07275 21.67690 24.33004    0
## rf    14.15121 17.05019 18.67701 18.85218 21.29413 23.81112    0
##
```

```
## Rsquared
##              Min.    1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lasso 0.04912708 0.09066924 0.1084539 0.1200302 0.1521564 0.2024257    0
## gam   0.14183084 0.17613970 0.2522873 0.2878543 0.3472104 0.5360815    0
## enet  0.01276916 0.08882672 0.1188672 0.1206508 0.1698191 0.1924017    0
## pls   0.04639288 0.08799442 0.1181685 0.1243392 0.1424814 0.2553219    0
## mars  0.14865601 0.16991506 0.2075188 0.2816192 0.3742323 0.5637279    0
## rf    0.09105527 0.18567934 0.2962244 0.3218141 0.4717690 0.6241630    0
```

Using bw-plot to compare their RMSE

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


pls
lasso
mars
gam
enet
rf
RMSE