model_building_I

Yun He

April 2, 2019

import data

```
train = read_csv("./data/train.csv")
test = read_csv("./data/test.csv")

x <- model.matrix(transformed_value~., train)[,-1]
y <- train$transformed_value

x2 <- model.matrix(transformed_value~.,test)[,-1]
y2 <- test$transformed_value
```

linear model

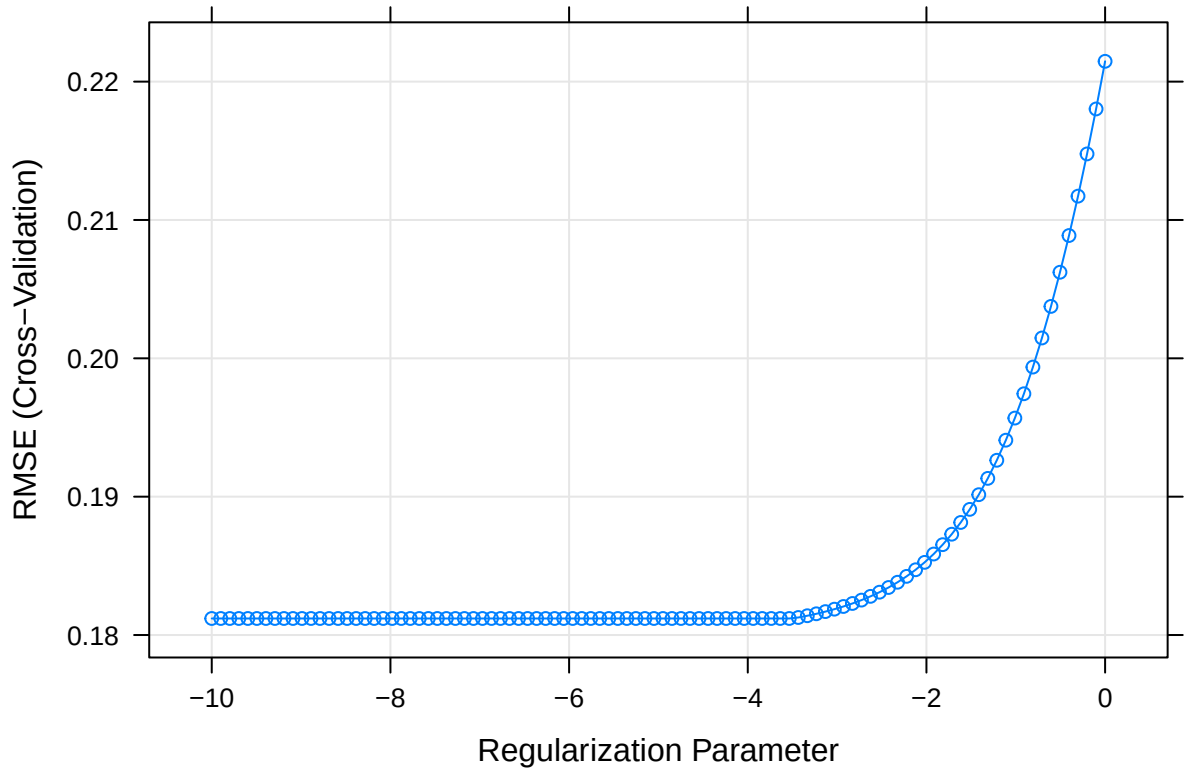
```
ctrl1 <- trainControl(method = "cv", number = 10)
set.seed(1)
lm.fit <- train(x, y,
                method = "lm",
                trControl = ctrl1)
predy2.lm <- predict(lm.fit$finalModel, newdata = data.frame(x2))
lm_test = mean((predy2.lm-y2)^2)

varImp(lm.fit)
```

```
## lm variable importance
##
##   only 20 most important variables shown (out of 42)
##
##                     Overall
## age                  100.00
## balance               38.63
## dribbling             35.60
## special               35.55
## penalties             35.43
## aggression            34.64
## vision                33.44
## positioning           33.27
## acceleration          32.78
## long_passing          32.22
## jumping               31.53
## composure             31.37
## heading_accuracy      30.31
## strength              29.78
## ball_control          29.78
## free_kick_accuracy    29.77
## crossing              29.68
## stamina               29.63
```

```
## sprint_speed          29.48
## shot_power            29.20
```

ridge model

```
set.seed(1)
ridge.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 0,
                                          lambda = exp(seq(-10, 0, length=100))),
                   # preProc = c("center", "scale"),
                   trControl = ctrl1)
plot(ridge.fit, xTrans = function(x) log(x))
```




```
predy2.ridge <- predict(ridge.fit$finalModel, newx = x2,
                        s = ridge.fit$bestTune$lambda, type = "response")
ridge_test = mean((predy2.ridge-y2)^2)

coef(ridge.fit$finalModel,ridge.fit$bestTune$lambda)
```

```
## 43 x 1 sparse Matrix of class "dgCMatrix"
##                                 1
## (Intercept)        -1.897364e+00
## age                -1.763865e-02
## nationalityas      -1.254453e-02
## nationalityeu       1.050577e-02
```

```
## nationalityna      -1.113662e-02
## nationalityoc      -6.630870e-03
## nationalitysa      -2.994470e-02
## potential           7.541403e-03
## special             2.298660e-04
## acceleration       -2.133166e-04
## aggression         -9.617806e-04
## agility             9.441624e-04
## balance            -1.488919e-03
## ball_control       -7.835165e-05
## composure           1.648875e-03
## crossing           -1.417865e-03
## curve              -1.054478e-05
## dribbling          -2.530617e-03
## finishing          -1.553289e-03
## free_kick_accuracy -6.905732e-04
## gk_diving           8.575535e-03
## gk_handling         8.989981e-03
## gk_kicking          2.229259e-03
## gk_positioning      5.217241e-03
## gk_reflexes         7.445288e-03
## heading_accuracy   -1.299186e-03
## interceptions       6.027679e-04
## jumping             3.868069e-04
## long_passing       -1.611657e-04
## long_shots          2.201049e-03
## marking            -1.323440e-03
## penalties          -1.158813e-03
## positioning        -2.775969e-03
## reactions           5.351495e-03
## short_passing       1.559862e-03
## shot_power          5.394068e-04
## sliding_tackle      5.062604e-04
## sprint_speed        4.290660e-04
## stamina             4.913771e-04
## standing_tackle     1.045669e-03
## strength            9.835697e-04
## vision              2.463476e-04
## volleys            -4.518024e-04
```

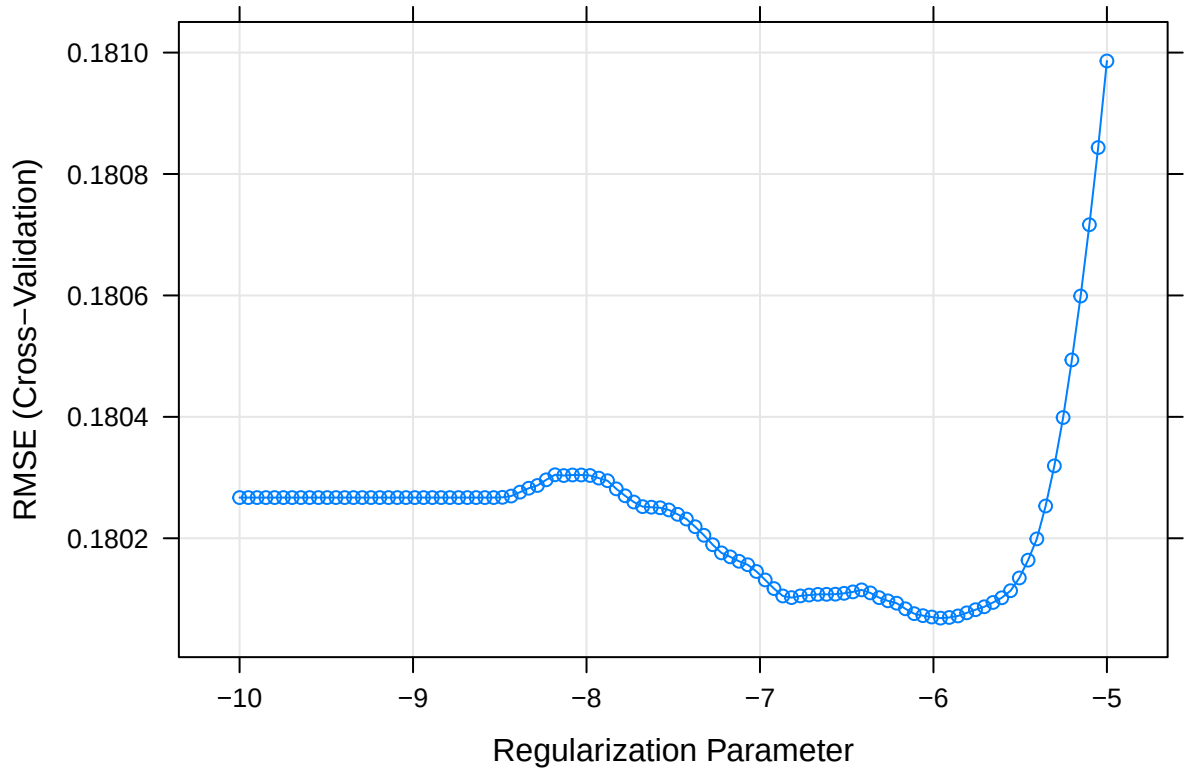
```
varImp(ridge.fit)
```

```
## glmnet variable importance
##
##   only 20 most important variables shown (out of 42)
##
##                Overall
## nationalitysa  100.000
## age             58.890
## nationalityas   41.872
## nationalityna   37.168
## nationalityeu   35.061
## gk_handling     29.997
## gk_diving       28.613
## potential       25.158
```

```
## gk_reflexes     24.837
## nationalityoc   22.116
## reactions       17.842
## gk_positioning  17.394
## positioning      9.238
## dribbling        8.419
## gk_kicking       7.412
## long_shots       7.318
## composure        5.473
## short_passing    5.176
## finishing        5.154
## balance          4.939
```

lasso model

```
set.seed(1)
lasso.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 1,
                                          lambda = exp(seq(-10, -5, length=100))),
                   # preProc = c("center", "scale"),
                   trControl = ctrl1)
plot(lasso.fit, xTrans = function(x) log(x))
```




```
predy2.lasso <- predict(lasso.fit$finalModel, newx = x2,
                        s = lasso.fit$bestTune$lambda, type = "response")
```

```r
lasso_test = mean((predy2.lasso-y2)^2)

coef(lasso.fit$finalModel,lasso.fit$bestTune$lambda)
```

```
## 43 x 1 sparse Matrix of class "dgCMatrix"
##                                1
## (Intercept)        -1.771904e+00
## age                -2.308253e-02
## nationalityas       .
## nationalityeu       1.244760e-02
## nationalityna       .
## nationalityoc       .
## nationalitysa      -1.947723e-02
## potential           3.377166e-03
## special             5.191991e-06
## acceleration        .
## aggression          .
## agility             1.077229e-03
## balance            -8.232521e-04
## ball_control        .
## composure           1.884373e-03
## crossing           -4.293553e-04
## curve               .
## dribbling          -1.607169e-03
## finishing          -6.478201e-04
## free_kick_accuracy -3.193254e-06
## gk_diving           1.038727e-02
## gk_handling         1.133035e-02
## gk_kicking          1.810138e-03
## gk_positioning      6.313652e-03
## gk_reflexes         8.131384e-03
## heading_accuracy   -3.721334e-04
## interceptions       1.120823e-05
## jumping             4.176150e-04
## long_passing        .
## long_shots          8.394057e-04
## marking            -3.911293e-04
## penalties          -4.398981e-04
## positioning        -1.460392e-03
## reactions           6.240556e-03
## short_passing       1.686996e-03
## shot_power          5.092825e-04
## sliding_tackle      1.089893e-04
## sprint_speed        1.636265e-04
## stamina             3.261305e-04
## standing_tackle     1.313050e-04
## strength            1.005467e-03
## vision              1.027491e-04
## volleys             .
```

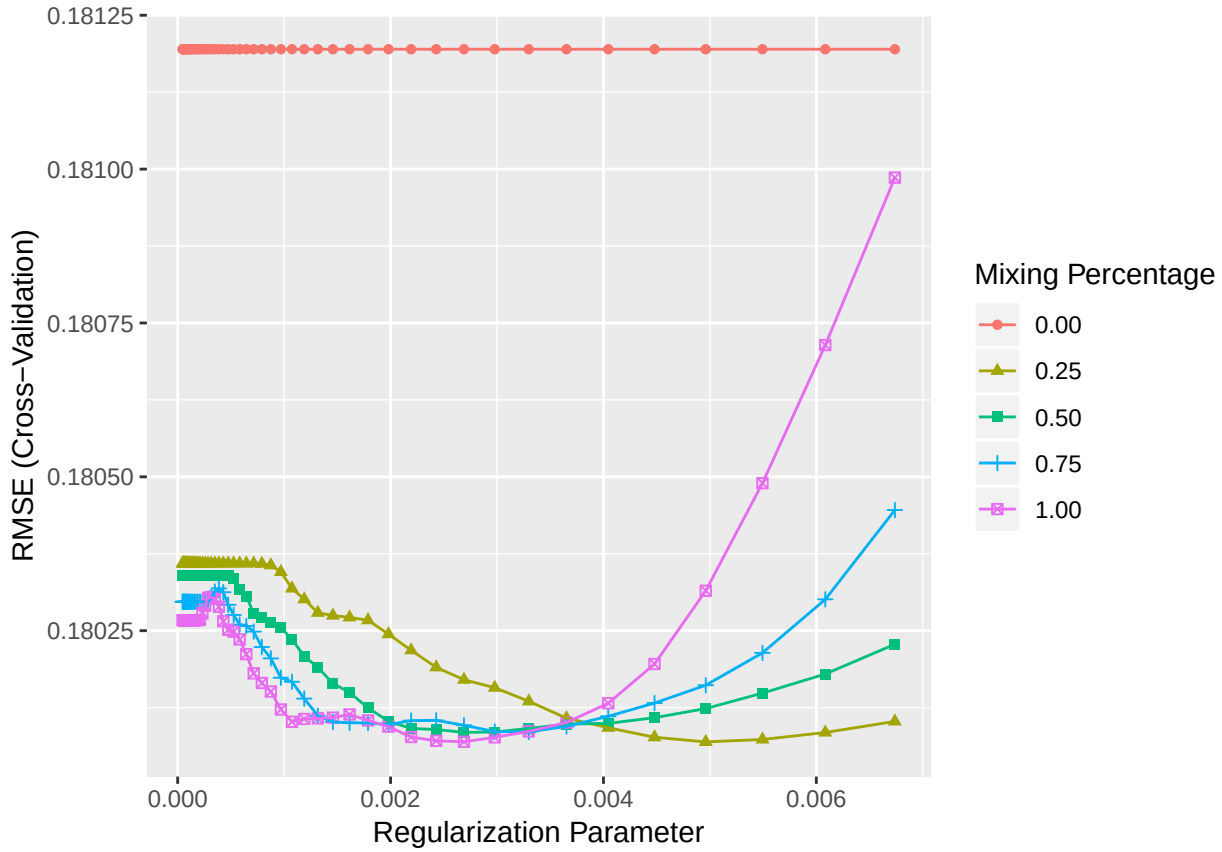
```r
varImp(lasso.fit)
```

```
## glmnet variable importance
##
```

```
##   only 20 most important variables shown (out of 42)
##
##                Overall
## age            100.000
## nationalitysa   84.381
## nationalityeu   53.926
## gk_handling     49.086
## gk_diving       45.001
## gk_reflexes     35.227
## gk_positioning  27.353
## reactions       27.036
## potential       14.631
## composure        8.164
## gk_kicking       7.842
## short_passing    7.309
## dribbling        6.963
## positioning      6.327
## agility          4.667
## strength         4.356
## long_shots       3.637
## balance          3.567
## finishing        2.807
## shot_power       2.206
```

Elastic net

```
set.seed(1)
enet.fit <- train(x, y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = seq(0, 1, length = 5),
                                         lambda = exp(seq(-10, -5, length=50))),
                 # preProc = c("center", "scale"),
                  trControl = ctrl1)
ggplot(enet.fit)
```




```
coef(enet.fit$finalModel,enet.fit$bestTune$lambda)
```

```
## 43 x 1 sparse Matrix of class "dgCMatrix"
##                                    1
## (Intercept)         -1.782965e+00
## age                 -2.290859e-02
## nationalityas       -3.049156e-04
## nationalityeu        1.465311e-02
## nationalityna        .
## nationalityoc        .
## nationalitysa       -2.039864e-02
## potential            3.586678e-03
## special              1.737466e-04
## acceleration         .
## aggression          -4.080915e-04
## agility              9.498667e-04
## balance             -1.217425e-03
## ball_control         .
## composure            1.896637e-03
## crossing            -1.024672e-03
## curve                .
## dribbling           -2.320874e-03
## finishing           -1.420519e-03
## free_kick_accuracy  -4.262853e-04
## gk_diving            9.988147e-03
## gk_handling          1.088536e-02
```

```
## gk_kicking          1.811080e-03
## gk_positioning      6.165404e-03
## gk_reflexes         8.006495e-03
## heading_accuracy   -9.118102e-04
## interceptions       3.323845e-04
## jumping             3.137730e-04
## long_passing        .
## long_shots          1.573973e-03
## marking            -1.041477e-03
## penalties          -8.992775e-04
## positioning        -2.241771e-03
## reactions           6.046682e-03
## short_passing       1.471072e-03
## shot_power          4.585584e-04
## sliding_tackle      3.922955e-04
## sprint_speed        4.818127e-05
## stamina             3.233077e-04
## standing_tackle     6.136202e-04
## strength            9.074761e-04
## vision              3.378389e-05
## volleys            -1.176993e-04
```

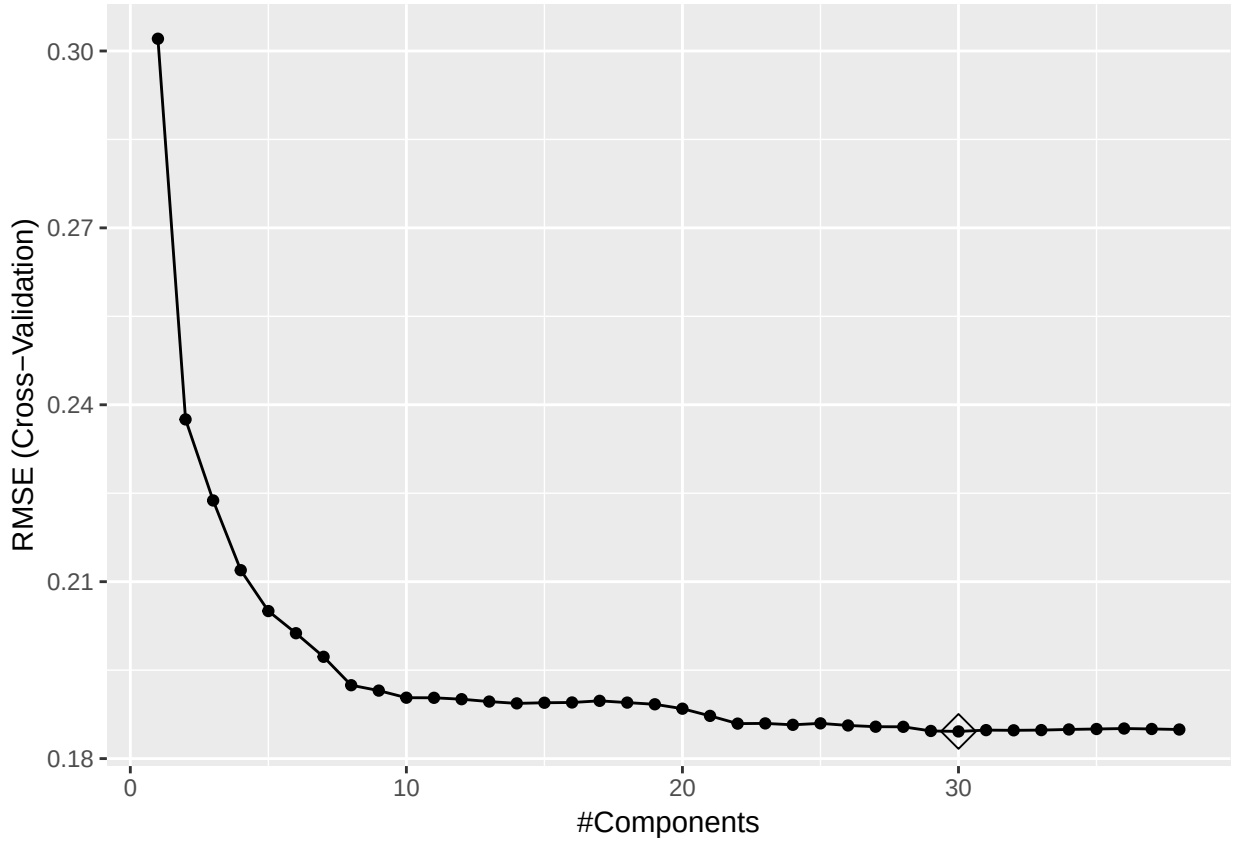
```
varImp(enet.fit)
```

```
## glmnet variable importance
##
##   only 20 most important variables shown (out of 42)
##
##                Overall
## age            100.000
## nationalitysa   89.044
## nationalityeu   63.963
## gk_handling     47.517
## gk_diving       43.600
## gk_reflexes     34.950
## gk_positioning  26.913
## reactions       26.395
## potential       15.656
## dribbling       10.131
## positioning      9.786
## composure        8.279
## gk_kicking       7.906
## long_shots       6.871
## short_passing    6.421
## finishing        6.201
## balance          5.314
## marking          4.546
## crossing         4.473
## agility          4.146
```

pcr model

```
set.seed(1)
pcr.fit <- train(x, y,
```

```
                  method = "pcr",
                  tuneLength = 38,
                  trControl = ctrl1,
                  scale = TRUE)
ggplot(pcr.fit, highlight = TRUE)
```



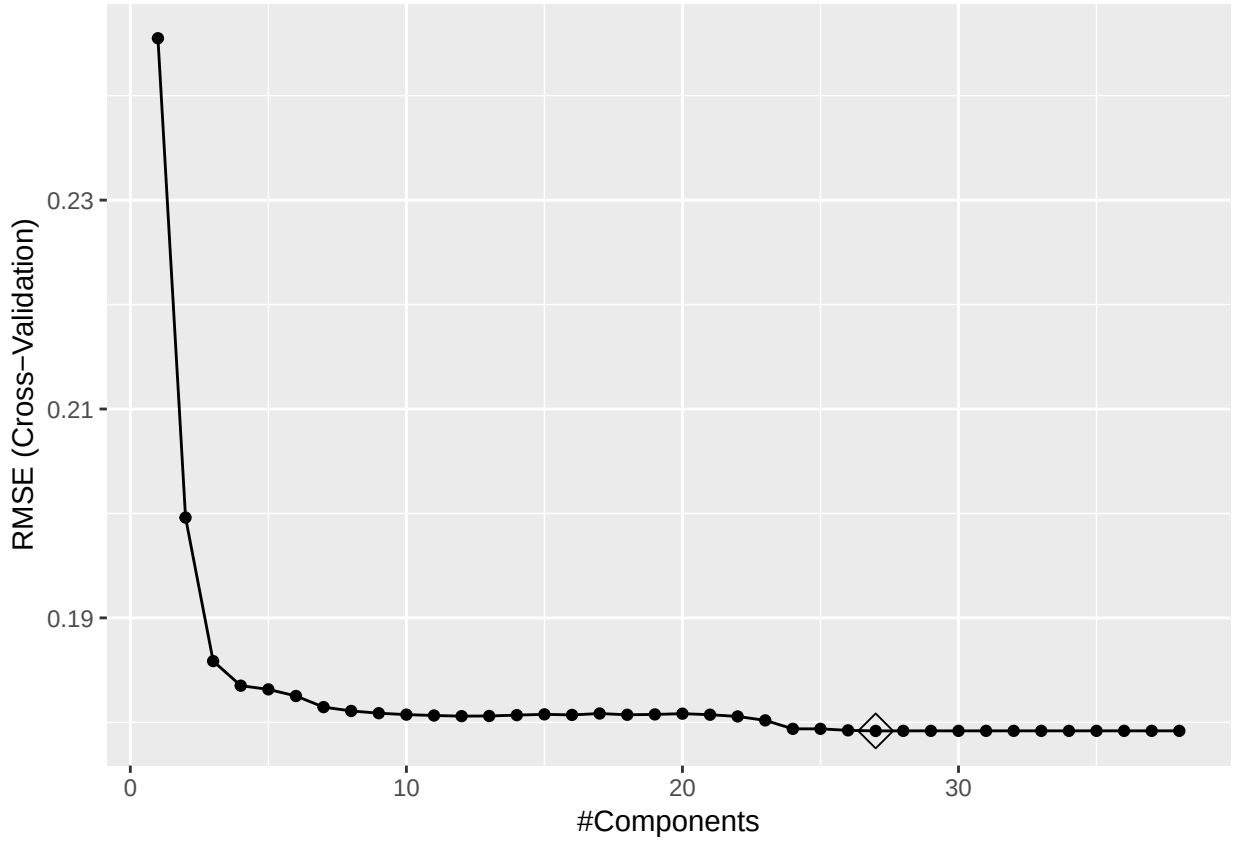

```
predy2.pcr <- predict(pcr.fit$finalModel, newdata = x2,
                      ncomp = pcr.fit$bestTune$ncomp)
pcr_test = mean((predy2.pcr-y2)^2)

varImp(pcr.fit)
```

```
## loess r-squared variable importance
##
##   only 20 most important variables shown (out of 42)
##
##                Overall
## gk_diving      100.000
## gk_reflexes     98.386
## potential       95.632
## gk_handling     91.859
## gk_positioning  82.945
## reactions       76.331
## special         71.032
## gk_kicking      60.218
## composure       22.340
```

```
## jumping          22.172
## sprint_speed     21.476
## vision           19.541
## acceleration     19.357
## agility          17.364
## strength         14.944
## stamina          10.533
## interceptions    10.007
## ball_control      9.887
## short_passing     8.595
## long_passing      6.765
```

pls model

```
set.seed(1)
pls.fit <- train(x, y,
                 method = "pls",
                 tuneLength = 38,
                 trControl = ctrl1,
                 scale = TRUE)
ggplot(pls.fit, highlight = TRUE)
```




```
predy2.pls <- predict(pls.fit$finalModel, newdata = x2,
                      ncomp = pls.fit$bestTune$ncomp)
pls_test = mean((predy2.pls-y2)^2)
```

```
varImp(pls.fit)
```

```
## Warning: package 'pls' was built under R version 3.5.2
```

```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:caret':
##
##     R2
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
## pls variable importance
##
##   only 20 most important variables shown (out of 42)
##
##                Overall
## potential       100.00
## gk_diving        91.33
## gk_reflexes      90.03
## gk_handling      87.01
## gk_positioning   78.78
## reactions        75.00
## gk_kicking       65.51
## special          64.55
## sprint_speed     36.61
## acceleration     35.92
## composure        35.70
## jumping          34.46
## vision           33.86
## agility          31.63
## age              27.43
## interceptions    26.77
## strength         26.30
## stamina          24.90
## ball_control     23.83
## long_shots       20.62
```

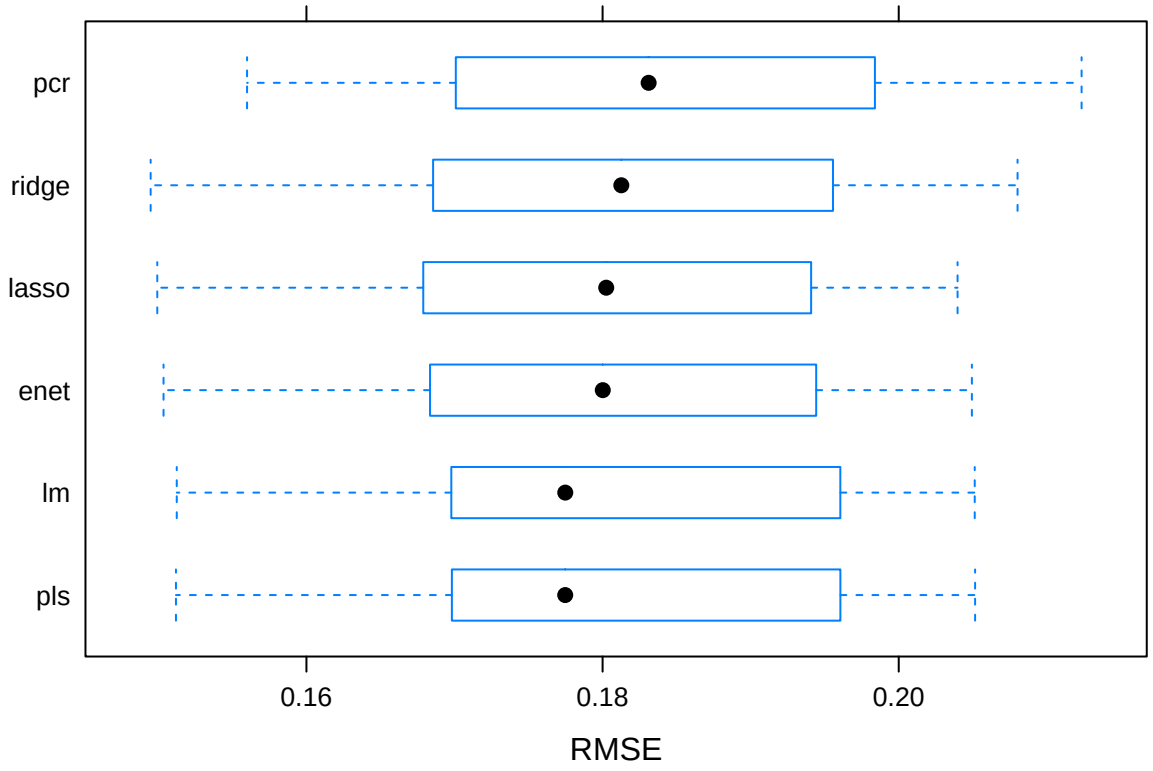
summarize

```
resamp <- resamples(list(lasso = lasso.fit,
                         ridge = ridge.fit,
                         enet = enet.fit,
                         pcr = pcr.fit,
                         pls = pls.fit,
                         lm = lm.fit))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: lasso, ridge, enet, pcr, pls, lm
```

```
## Number of resamples: 10
##
## MAE
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lasso 0.10205764 0.1030962 0.1086979 0.1100080 0.1154672 0.1230177    0
## ridge 0.10238478 0.1036699 0.1108511 0.1111234 0.1154873 0.1247666    0
## enet  0.10285582 0.1034415 0.1095330 0.1107157 0.1156639 0.1240157    0
## pcr   0.10509740 0.1098126 0.1140492 0.1155562 0.1198668 0.1293410    0
## pls   0.09989983 0.1056448 0.1090365 0.1111397 0.1169945 0.1230287    0
## lm    0.09997055 0.1056307 0.1090161 0.1111360 0.1169643 0.1230271    0
##
## RMSE
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lasso 0.1499253 0.1687402 0.1802447 0.1800685 0.1924932 0.2039717    0
## ridge 0.1494777 0.1691021 0.1812648 0.1811948 0.1938712 0.2080249    0
## enet  0.1503483 0.1687419 0.1800135 0.1800693 0.1927041 0.2049398    0
## pcr   0.1559914 0.1707663 0.1831084 0.1846069 0.1970660 0.2123488    0
## pls   0.1511874 0.1702986 0.1774748 0.1791764 0.1926560 0.2051569    0
## lm    0.1512410 0.1702754 0.1774798 0.1791838 0.1926812 0.2051311    0
##
## Rsquared
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lasso 0.6898485 0.7510821 0.8004871 0.7810596 0.8085208 0.8346502    0
## ridge 0.6861517 0.7443270 0.8001998 0.7785595 0.8070815 0.8289576    0
## enet  0.6910436 0.7491928 0.8014740 0.7810765 0.8106707 0.8325230    0
## pcr   0.6753470 0.7372744 0.7892220 0.7698066 0.8029685 0.8148068    0
## pls   0.7077368 0.7500180 0.8053972 0.7833781 0.8097563 0.8297895    0
## lm    0.7073062 0.7500721 0.8054011 0.7833462 0.8098160 0.8298176    0
```

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


pcr
ridge
lasso
enet
lm
pls
0.16
0.18
0.20
RMSE