Week11

Mathi Manavalan

4/6/2020

Libraries

```
library(tidyverse)
library(Hmisc)
library(caret)
```

Data Import and Cleaning

First, I am importing the entire dataset.

```
data <- as_tibble(spss.get("../data/GSS2006.sav", use.value.labels=TRUE))
```

Now, we only want the variables relevant to the personality inventory and the respondent's self-reported health. For the personality inventory, I am interpreting this to mean the variables (according to the code book for this dataset) from BIG5A1 to BIG5E2 from *The 2006 Module: Personality Traits*. For the respondent's self-reported health, I am interpreting this to mean the HEALTH variable.

```
import <- data %>%
  select(BIG5A1, BIG5A2, BIG5B1, BIG5B2, BIG5C1, BIG5C2, BIG5D1, BIG5D2, BIG5E1, BIG5E2, HEALTH) %>%
  filter_all(any_vars(!is.na(.))) %>%  #removes rows where ALL values in row are NA
  drop_na(HEALTH) %>%  # removes rows where our response variable, HEALTH, is NA
  mutate_all(. %>%
           factor() %>%
           as.numeric()
         )

clean <- import[!(rowSums(is.na(import)) == 10),]
```

First, I selected all of the data pertaining to the personality inventory as well as the HEALTH variable. Then I removed all the rows that had NA responses for ALL of the variables. (Responses of 'don't know', 'inapplicable', or other unclearly answered items are appropriately marked as NA according to R.) Then, I removed all the rows where the response to the HEALTH variable was NA, since HEALTH is what we are trying to predict so we only want rows with a valid response for HEALTH. Last but not least, I converted all the variables into numeric factors. Finally, I created a clean tibble that doesn't contain rows which have NA responses for all the predictor variables (aka, the personality inventory variables, of which there are 10). In other words, I am keeping the rows which may have some predictor variables that have NA but if all the predictors are NA, the row is removed.

Analysis

Here, I am splitting the clean data into two tibbles. One (holdout) contains a random 250 set from clean, and the other (train) contains the remaining samples from. Then, I am creating an OLS model with 10-fold cross-validation (named olsr).

Here are the output of olsr as well as prediction in holdout.

```
summary(olsr)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.79099 -0.66207 -0.05125  0.45282  2.36667
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.032793   0.023281  87.316   <2e-16 ***
## BIG5A1          -0.100346   0.196259  -0.511   0.6092
## BIG5A2          -0.074697   0.238516  -0.313   0.7542
## BIG5B1           0.361268   0.217247   1.663   0.0966 .
## BIG5B2          -0.406069   0.199525  -2.035   0.0421 *
## BIG5C1           0.226416   0.185020   1.224   0.2213
## BIG5C2           0.342332   0.190845   1.794   0.0731 .
## BIG5D1           0.251920   0.213129   1.182   0.2374
## BIG5D2           0.291398   0.213042   1.368   0.1716
## BIG5E1           0.011767   0.225794   0.052   0.9584
## BIG5E2          -0.120597   0.181746  -0.664   0.5071
## `BIG5A1:BIG5A2`  0.062731   0.069494   0.903   0.3669
## `BIG5A1:BIG5B1`  0.037515   0.082563   0.454   0.6496
## `BIG5A1:BIG5B2` -0.067409   0.101149  -0.666   0.5053
## `BIG5A1:BIG5C1` -0.018991   0.094974  -0.200   0.8415
## `BIG5A1:BIG5C2` -0.091709   0.126325  -0.726   0.4680
## `BIG5A1:BIG5D1` -0.030228   0.085773  -0.352   0.7246
## `BIG5A1:BIG5D2`  0.062700   0.111317   0.563   0.5734
## `BIG5A1:BIG5E1`  0.081657   0.080235   1.018   0.3090
## `BIG5A1:BIG5E2`  0.031937   0.089601   0.356   0.7216
## `BIG5A2:BIG5B1`  0.187472   0.094501   1.984   0.0475 *
## `BIG5A2:BIG5B2`  0.232992   0.111162   2.096   0.0363 *
## `BIG5A2:BIG5C1` -0.160811   0.105854  -1.519   0.1290
## `BIG5A2:BIG5C2` -0.132338   0.142297  -0.930   0.3525
## `BIG5A2:BIG5D1` -0.033693   0.102461  -0.329   0.7423
## `BIG5A2:BIG5D2` -0.033867   0.095217  -0.356   0.7221
## `BIG5A2:BIG5E1` -0.028444   0.089723  -0.317   0.7513
## `BIG5A2:BIG5E2` -0.021067   0.093456  -0.225   0.8217
## `BIG5B1:BIG5B2` -0.114282   0.086578  -1.320   0.1871
## `BIG5B1:BIG5C1` -0.078512   0.097494  -0.805   0.4208
## `BIG5B1:BIG5C2` -0.084605   0.130181  -0.650   0.5159
## `BIG5B1:BIG5D1` -0.221020   0.090183  -2.451   0.0144 *
## `BIG5B1:BIG5D2` -0.096848   0.107634  -0.900   0.3684
## `BIG5B1:BIG5E1`  0.039899   0.084186   0.474   0.6356
```

```
## `BIG5B1:BIG5E2` -0.117301   0.092577  -1.267   0.2054
## `BIG5B2:BIG5C1`  0.063058   0.104433   0.604   0.5461
## `BIG5B2:BIG5C2`  0.047747   0.139782   0.342   0.7327
## `BIG5B2:BIG5D1`  0.157079   0.095127   1.651   0.0989 .
## `BIG5B2:BIG5D2`  0.150476   0.124120   1.212   0.2256
## `BIG5B2:BIG5E1`  0.152610   0.103028   1.481   0.1388
## `BIG5B2:BIG5E2`  0.132097   0.104977   1.258   0.2085
## `BIG5C1:BIG5C2`  0.011543   0.108087   0.107   0.9150
## `BIG5C1:BIG5D1`  0.125126   0.111280   1.124   0.2611
## `BIG5C1:BIG5D2` -0.061881   0.109296  -0.566   0.5714
## `BIG5C1:BIG5E1` -0.095285   0.101743  -0.937   0.3492
## `BIG5C1:BIG5E2` -0.059102   0.097057  -0.609   0.5427
## `BIG5C2:BIG5D1` -0.158952   0.129591  -1.227   0.2202
## `BIG5C2:BIG5D2` -0.245234   0.153363  -1.599   0.1101
## `BIG5C2:BIG5E1` -0.065621   0.125603  -0.522   0.6015
## `BIG5C2:BIG5E2`  0.003532   0.131898   0.027   0.9786
## `BIG5D1:BIG5D2` -0.055930   0.074589  -0.750   0.4535
## `BIG5D1:BIG5E1` -0.132214   0.090596  -1.459   0.1447
## `BIG5D1:BIG5E2`  0.112807   0.092100   1.225   0.2209
## `BIG5D2:BIG5E1` -0.120994   0.099749  -1.213   0.2254
## `BIG5D2:BIG5E2` -0.083156   0.111535  -0.746   0.4561
## `BIG5E1:BIG5E2`  0.140816   0.078013   1.805   0.0713 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8274 on 1210 degrees of freedom
## Multiple R-squared:  0.07825,    Adjusted R-squared:  0.03636
## F-statistic: 1.868 on 55 and 1210 DF,  p-value: 0.0001662
```

```
olsr
```

```
## Linear Regression
##
## 1266 samples
##   10 predictor
##
## Pre-processing: centered (55), scaled (55), median imputation (55)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1140, 1140, 1139, 1139, 1138, 1139, ...
## Resampling results:
##
##   RMSE       Rsquared    MAE
##   0.8513884  0.02229439  0.6560583
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
predict(olsr, holdout, na.action = na.pass)
```

```
##        1        2        3        4        5        6        7        8
## 2.116260 1.734208 2.145191 2.247923 1.966916 2.167533 2.177837 1.969752
##        9       10       11       12       13       14       15       16
## 2.372109 2.346684 1.991081 2.273329 2.478216 2.299674 1.965975 1.674821
##       17       18       19       20       21       22       23       24
```

```
## 1.950557 2.311573 2.056714 2.328769 1.585388 2.074220 2.006294 2.420033
##       25       26       27       28       29       30       31       32
## 1.927838 1.880970 2.120307 2.016021 2.073859 2.014656 2.202375 2.206796
##       33       34       35       36       37       38       39       40
## 2.225275 2.234072 2.427504 1.913155 1.862722 1.677132 1.829604 2.111246
##       41       42       43       44       45       46       47       48
## 1.974736 2.046495 2.094210 2.300394 2.411145 2.129418 1.871219 1.741259
##       49       50       51       52       53       54       55       56
## 1.713776 1.978727 2.166233 2.410398 1.816264 1.976902 2.028333 2.154654
##       57       58       59       60       61       62       63       64
## 2.115359 1.989717 2.003073 2.047380 1.713909 2.311017 2.180292 1.963491
##       65       66       67       68       69       70       71       72
## 2.424554 2.038563 2.026548 2.186884 2.383111 1.944442 2.377622 1.936341
##       73       74       75       76       77       78       79       80
## 2.114196 2.006750 2.054366 2.073859 2.091270 1.845179 2.296709 1.918980
##       81       82       83       84       85       86       87       88
## 2.062109 2.013723 1.695967 2.087039 2.183719 2.324096 1.876337 1.873375
##       89       90       91       92       93       94       95       96
## 2.264734 2.817373 2.033662 1.884132 1.933761 2.218484 2.019225 1.882354
##       97       98       99      100      101      102      103      104
## 2.053858 2.541883 1.800349 2.085291 1.936085 1.591927 2.611652 2.137665
##      105      106      107      108      109      110      111      112
## 2.011991 1.721318 1.972823 1.978419 1.871430 2.188506 1.817758 2.062434
##      113      114      115      116      117      118      119      120
## 1.994248 1.917729 1.915040 1.951146 2.385293 1.768031 2.053580 2.089343
##      121      122      123      124      125      126      127      128
## 2.162998 2.103723 2.007164 2.393627 1.944446 1.992436 2.160764 2.065004
##      129      130      131      132      133      134      135      136
## 2.305146 1.897331 1.832840 2.273424 1.875179 1.991828 2.018559 1.963730
##      137      138      139      140      141      142      143      144
## 1.947633 1.949932 2.199817 2.187711 1.879850 2.180821 1.714023 1.460697
##      145      146      147      148      149      150      151      152
## 1.871524 1.931742 1.921801 2.021835 1.906879 1.564101 2.023370 1.834544
##      153      154      155      156      157      158      159      160
## 1.834667 2.041092 2.210447 1.664550 2.116693 1.908287 2.554526 1.879003
##      161      162      163      164      165      166      167      168
## 2.304645 1.777602 2.239179 2.252064 2.069021 1.974169 1.948438 1.986224
##      169      170      171      172      173      174      175      176
## 2.004106 2.170545 2.154829 2.179185 1.875577 2.047798 1.991441 2.014656
##      177      178      179      180      181      182      183      184
## 2.135068 2.144029 2.122959 2.129092 2.005050 1.968765 2.029635 1.504436
##      185      186      187      188      189      190      191      192
## 2.601862 1.978419 1.981109 1.957562 1.625836 1.913894 2.153368 1.899570
##      193      194      195      196      197      198      199      200
## 2.279627 2.168247 1.578887 1.675322 1.660827 1.936829 1.892459 1.551281
##      201      202      203      204      205      206      207      208
## 2.006766 2.108336 2.216940 2.103417 1.817788 2.140171 2.275224 2.001573
##      209      210      211      212      213      214      215      216
## 1.965147 2.203719 1.569665 1.935584 1.996508 1.904068 2.113079 2.037179
##      217      218      219      220      221      222      223      224
## 2.022692 1.717186 1.966529 1.564101 1.981603 2.446470 2.148756 1.708584
##      225      226      227      228      229      230      231      232
## 2.183811 2.497465 2.292875 2.471294 1.797526 2.164863 2.350788 1.698294
##      233      234      235      236      237      238      239      240
```

```
## 2.112052 1.680638 1.645562 2.086431 2.407712 2.144029 1.885850 2.075203
##      241      242      243      244      245      246      247      248
## 1.914028 2.062109 2.094023 1.641840 2.934233 2.251352 1.975096 1.603464
##      249      250
## 1.700985 1.991529
```

We can see from the above output that the R squared value is almost zero.

I have then ran this model 3 more times using different types of regression, along with the prediction on the holdout set of each model.

10-fold elastic net regression

```
elastic <- train(
  HEALTH ~ .*.,
  train,
  method = "glmnet",
  preProcess = c("center", "scale", "zv", "medianImpute"),
  trControl = trainControl(
    method = "cv",
    number = 10,
    verboseIter = TRUE
  ),
  na.action = na.pass
)
```

```
## + Fold01: alpha=0.10, lambda=0.02531
## - Fold01: alpha=0.10, lambda=0.02531
## + Fold01: alpha=0.55, lambda=0.02531
## - Fold01: alpha=0.55, lambda=0.02531
## + Fold01: alpha=1.00, lambda=0.02531
## - Fold01: alpha=1.00, lambda=0.02531
## + Fold02: alpha=0.10, lambda=0.02531
## - Fold02: alpha=0.10, lambda=0.02531
## + Fold02: alpha=0.55, lambda=0.02531
## - Fold02: alpha=0.55, lambda=0.02531
## + Fold02: alpha=1.00, lambda=0.02531
## - Fold02: alpha=1.00, lambda=0.02531
## + Fold03: alpha=0.10, lambda=0.02531
## - Fold03: alpha=0.10, lambda=0.02531
## + Fold03: alpha=0.55, lambda=0.02531
## - Fold03: alpha=0.55, lambda=0.02531
## + Fold03: alpha=1.00, lambda=0.02531
## - Fold03: alpha=1.00, lambda=0.02531
## + Fold04: alpha=0.10, lambda=0.02531
## - Fold04: alpha=0.10, lambda=0.02531
## + Fold04: alpha=0.55, lambda=0.02531
## - Fold04: alpha=0.55, lambda=0.02531
## + Fold04: alpha=1.00, lambda=0.02531
## - Fold04: alpha=1.00, lambda=0.02531
## + Fold05: alpha=0.10, lambda=0.02531
## - Fold05: alpha=0.10, lambda=0.02531
## + Fold05: alpha=0.55, lambda=0.02531
## - Fold05: alpha=0.55, lambda=0.02531
```

```
## + Fold05: alpha=1.00, lambda=0.02531
## - Fold05: alpha=1.00, lambda=0.02531
## + Fold06: alpha=0.10, lambda=0.02531
## - Fold06: alpha=0.10, lambda=0.02531
## + Fold06: alpha=0.55, lambda=0.02531
## - Fold06: alpha=0.55, lambda=0.02531
## + Fold06: alpha=1.00, lambda=0.02531
## - Fold06: alpha=1.00, lambda=0.02531
## + Fold07: alpha=0.10, lambda=0.02531
## - Fold07: alpha=0.10, lambda=0.02531
## + Fold07: alpha=0.55, lambda=0.02531
## - Fold07: alpha=0.55, lambda=0.02531
## + Fold07: alpha=1.00, lambda=0.02531
## - Fold07: alpha=1.00, lambda=0.02531
## + Fold08: alpha=0.10, lambda=0.02531
## - Fold08: alpha=0.10, lambda=0.02531
## + Fold08: alpha=0.55, lambda=0.02531
## - Fold08: alpha=0.55, lambda=0.02531
## + Fold08: alpha=1.00, lambda=0.02531
## - Fold08: alpha=1.00, lambda=0.02531
## + Fold09: alpha=0.10, lambda=0.02531
## - Fold09: alpha=0.10, lambda=0.02531
## + Fold09: alpha=0.55, lambda=0.02531
## - Fold09: alpha=0.55, lambda=0.02531
## + Fold09: alpha=1.00, lambda=0.02531
## - Fold09: alpha=1.00, lambda=0.02531
## + Fold10: alpha=0.10, lambda=0.02531
## - Fold10: alpha=0.10, lambda=0.02531
## + Fold10: alpha=0.55, lambda=0.02531
## - Fold10: alpha=0.55, lambda=0.02531
## + Fold10: alpha=1.00, lambda=0.02531
## - Fold10: alpha=1.00, lambda=0.02531
## Aggregating results
## Selecting tuning parameters
## Fitting alpha = 1, lambda = 0.0253 on full training set
```

```
predict(elastic, holdout, na.action = na.pass)
```

```
##        1        2        3        4        5        6        7        8
## 2.052305 1.767648 2.229763 1.908733 1.939095 2.058307 1.908733 1.939022
##        9       10       11       12       13       14       15       16
## 2.147694 2.175647 1.943874 2.162715 2.110588 2.232065 1.942950 1.736628
##       17       18       19       20       21       22       23       24
## 2.033746 2.127197 1.995900 2.112620 1.826639 1.980060 2.115698 2.256973
##       25       26       27       28       29       30       31       32
## 1.919939 1.933087 2.082795 2.002293 2.031248 2.002293 2.162827 2.285506
##       33       34       35       36       37       38       39       40
## 2.073172 2.152884 2.070032 2.010480 2.021766 1.831048 2.052305 2.125288
##       41       42       43       44       45       46       47       48
## 1.981332 2.019634 2.013339 2.173457 2.218704 2.101944 1.927807 2.037481
##       49       50       51       52       53       54       55       56
## 1.919204 2.074280 2.126141 2.185634 1.885883 2.057206 1.992201 2.069539
##       57       58       59       60       61       62       63       64
## 2.102600 1.977724 2.019647 2.161742 1.843985 2.194415 2.240119 1.878831
```

```
##       65       66       67       68       69       70       71       72
## 2.103533 2.019647 2.082123 2.252158 2.081841 2.017015 2.169227 1.952904
##       73       74       75       76       77       78       79       80
## 2.073172 2.050722 1.986313 2.031248 2.021217 1.814991 2.084127 1.918180
##       81       82       83       84       85       86       87       88
## 2.031248 2.072779 2.141178 2.031248 2.103897 2.015476 1.932526 1.887338
##       89       90       91       92       93       94       95       96
## 2.032581 2.515172 1.874115 1.939798 1.842219 2.097368 2.093670 2.018308
##       97       98       99      100      101      102      103      104
## 1.960648 2.294264 1.804433 2.086161 1.896323 1.886817 2.169858 2.117926
##      105      106      107      108      109      110      111      112
## 2.048420 1.924513 1.977724 2.006429 2.066798 1.982566 1.939408 1.995809
##      113      114      115      116      117      118      119      120
## 2.139324 1.898007 1.907158 2.061978 2.033746 1.931632 2.007706 2.052210
##      121      122      123      124      125      126      127      128
## 1.975160 2.073172 1.998686 2.119593 1.889822 2.083189 2.099657 1.871406
##      129      130      131      132      133      134      135      136
## 2.210203 1.907898 2.102902 2.212516 2.012784 1.851246 2.098453 1.964009
##      137      138      139      140      141      142      143      144
## 1.968552 1.978719 2.142008 2.160826 1.950010 2.174567 2.027872 1.898989
##      145      146      147      148      149      150      151      152
## 1.950001 1.865625 1.781094 2.036321 2.017407 1.719323 1.952851 1.945688
##      153      154      155      156      157      158      159      160
## 1.774408 1.997734 2.155155 2.117343 1.881739 1.899741 2.338619 1.996809
##      161      162      163      164      165      166      167      168
## 2.265492 1.900645 2.076502 2.083964 2.027391 2.015197 1.944244 2.047808
##      169      170      171      172      173      174      175      176
## 2.078666 2.046603 2.218422 2.047810 1.949885 2.029209 2.006429 2.002293
##      177      178      179      180      181      182      183      184
## 2.146681 2.073172 1.963224 2.061108 2.064404 2.062930 2.074837 1.802863
##      185      186      187      188      189      190      191      192
## 2.016792 1.931465 2.031248 2.010287 1.948844 1.925688 2.093171 1.948913
##      193      194      195      196      197      198      199      200
## 2.231541 2.096663 1.779712 1.823387 1.963380 1.985324 1.867131 1.907413
##      201      202      203      204      205      206      207      208
## 2.097897 2.091501 2.218986 2.119970 1.934226 2.063160 2.027050 2.014757
##      209      210      211      212      213      214      215      216
## 2.189879 2.127408 2.079737 2.037719 2.031248 1.946636 2.003836 2.031248
##      217      218      219      220      221      222      223      224
## 2.074373 2.139749 2.165250 1.719323 1.970507 2.067451 2.062669 1.997048
##      225      226      227      228      229      230      231      232
## 1.995900 2.253448 2.093157 2.239797 1.867588 2.066056 2.248529 1.875676
##      233      234      235      236      237      238      239      240
## 2.151924 1.895270 1.787423 2.058776 2.226817 2.073172 1.931943 2.084127
##      241      242      243      244      245      246      247      248
## 1.970670 2.031248 2.147638 1.694896 2.380484 2.089292 1.978719 1.977100
##      249      250
## 1.900375 1.997048
```

As we can see from the above output, the tuning parameters that worked best for the 10-fold elastic model was alpha = 1 and lambda = 0.0253. Since the optimal alpha value reported was 1, we know that the most optimal model was Lasso. And from the lambda value, we can see that there was a pretty moderate penalty.

support vector regression

```
support <- train(
  HEALTH ~ .*.,
  train,
  method = "svmLinear",
  preProcess = c("center", "scale", "zv", "medianImpute"),
  trControl = trainControl(
    method = "cv",
    number = 10,
    verboseIter = TRUE
  ),
  na.action = na.pass
)
```

```
## + Fold01: C=1
## - Fold01: C=1
## + Fold02: C=1
## - Fold02: C=1
## + Fold03: C=1
## - Fold03: C=1
## + Fold04: C=1
## - Fold04: C=1
## + Fold05: C=1
## - Fold05: C=1
## + Fold06: C=1
## - Fold06: C=1
## + Fold07: C=1
## - Fold07: C=1
## + Fold08: C=1
## - Fold08: C=1
## + Fold09: C=1
## - Fold09: C=1
## + Fold10: C=1
## - Fold10: C=1
## Aggregating results
## Fitting final model on full training set
```

```
predict(support, holdout, na.action = na.pass)
```

```
##        1        2        3        4        5        6        7        8
## 2.042573 1.680796 2.075836 1.968463 1.848860 2.064433 1.946923 1.912504
##        9       10       11       12       13       14       15       16
## 2.222704 2.090322 1.921459 2.105728 2.261021 2.106448 1.936066 1.778481
##       17       18       19       20       21       22       23       24
## 1.871213 2.176968 2.014230 2.053225 1.783707 1.971041 2.001215 2.170275
##       25       26       27       28       29       30       31       32
## 1.877038 1.799871 2.040004 2.078123 2.004202 2.022032 2.099426 2.068668
##       33       34       35       36       37       38       39       40
## 2.079988 2.009805 2.185636 1.961129 1.864209 1.584632 1.903471 2.021732
##       41       42       43       44       45       46       47       48
## 1.962411 2.014082 2.048501 2.054368 2.061712 2.016152 1.920468 1.987963
##       49       50       51       52       53       54       55       56
## 1.849724 1.977820 2.044548 2.274679 1.832925 1.948944 1.973379 2.033299
##       57       58       59       60       61       62       63       64
```

```
## 1.983513 1.958681 1.964583 2.046584 1.805400 2.094519 2.053852 1.894835
##       65       66       67       68       69       70       71       72
## 2.210337 2.007599 1.950252 2.065436 2.258888 1.939709 2.130780 1.948039
##       73       74       75       76       77       78       79       80
## 2.036989 2.042264 1.926185 2.004202 2.013872 1.835843 2.111543 1.904350
##       81       82       83       84       85       86       87       88
## 1.997451 1.978127 1.670413 1.996266 1.997465 2.034922 1.880613 1.890030
##       89       90       91       92       93       94       95       96
## 2.009591 2.349060 1.920762 1.930458 1.819398 2.025292 1.951167 1.895345
##       97       98       99      100      101      102      103      104
## 2.000019 2.018712 1.766802 1.964295 1.907542 1.837271 2.416250 2.060281
##      105      106      107      108      109      110      111      112
## 1.978765 1.777250 1.957364 1.973010 1.919068 1.985298 1.879726 1.990595
##      113      114      115      116      117      118      119      120
## 1.987634 1.941572 1.933875 1.982925 2.160636 1.818661 1.932621 2.002750
##      121      122      123      124      125      126      127      128
## 2.013165 2.035170 1.974056 2.163704 1.876615 1.863395 2.047994 1.968950
##      129      130      131      132      133      134      135      136
## 2.087061 1.893622 1.898059 2.143756 1.944650 1.981260 2.012192 1.931122
##      137      138      139      140      141      142      143      144
## 1.885322 1.942266 2.096468 2.068651 1.830368 2.096647 1.785139 1.701147
##      145      146      147      148      149      150      151      152
## 1.889451 1.984402 2.057011 1.981634 1.975149 1.656135 1.984866 1.869661
##      153      154      155      156      157      158      159      160
## 1.912608 1.938863 2.078756 1.690529 1.971341 1.902934 2.120514 1.896550
##      161      162      163      164      165      166      167      168
## 2.183046 1.834629 2.084443 2.126070 2.011112 1.967141 1.966707 2.003917
##      169      170      171      172      173      174      175      176
## 2.039628 2.067089 1.879220 2.150674 1.902125 1.920460 1.976274 2.022032
##      177      178      179      180      181      182      183      184
## 1.968082 2.046524 2.112764 1.945631 1.953093 2.024770 1.948218 1.858976
##      185      186      187      188      189      190      191      192
## 2.052698 1.925677 1.944496 1.956048 1.822671 1.935075 2.133248 1.927106
##      193      194      195      196      197      198      199      200
## 2.178625 2.105735 1.706469 1.786683 1.769953 1.901193 1.871598 1.782861
##      201      202      203      204      205      206      207      208
## 1.911727 2.000314 2.135310 2.002766 1.862023 2.023657 2.089502 1.968615
##      209      210      211      212      213      214      215      216
## 1.905905 2.099312 1.856312 1.914073 1.986571 1.883945 1.990480 1.998637
##      217      218      219      220      221      222      223      224
## 1.979528 1.780969 1.998159 1.656135 1.961194 2.247054 2.132295 1.890586
##      225      226      227      228      229      230      231      232
## 2.039599 2.284234 2.251250 2.250172 1.944585 2.120327 2.190437 1.857916
##      233      234      235      236      237      238      239      240
## 2.051047 1.721949 1.773638 2.010773 2.199406 2.046524 1.932100 2.015898
##      241      242      243      244      245      246      247      248
## 1.911132 1.997451 2.029197 1.608980 2.558319 2.145612 1.982728 1.776132
##      249      250
## 1.851294 1.972150
```

extreme gradient boosted regression

```r
extreme <- train(
  HEALTH ~ .*.,
  train,
  method = "xgbLinear",
  preProcess = c("center", "scale", "zv", "medianImpute"),
  trControl = trainControl(
    method = "cv",
    number = 10,
    verboseIter = TRUE
  ),
  na.action = na.pass
)
```

```
## + Fold01: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold01: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold01: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold01: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold01: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold01: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold01: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold01: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold01: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold01: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold01: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold01: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold01: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold01: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold01: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold01: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold01: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold01: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold01: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold01: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold01: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold01: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold01: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold01: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold01: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold01: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold01: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold01: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold01: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold01: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold01: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold01: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold01: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold01: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold01: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold01: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold01: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold01: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold01: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold01: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold01: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold01: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold01: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold01: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold01: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold01: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold01: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold01: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold01: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold01: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold01: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold01: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold01: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold01: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold02: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold02: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold02: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold02: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold02: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold02: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold02: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold02: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold02: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold02: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold02: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold02: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold02: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold02: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold02: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold02: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold02: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold02: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold02: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold02: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold02: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold02: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold02: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold02: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold02: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold02: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold02: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold02: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold02: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold02: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold02: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold02: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold02: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold02: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold02: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold02: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold02: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold02: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold02: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold02: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold02: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold02: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold02: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold02: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold02: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold02: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold02: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold02: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold02: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold02: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold02: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold02: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold02: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold02: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold03: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold03: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold03: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold03: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold03: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold03: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold03: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold03: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold03: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold03: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold03: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold03: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold03: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold03: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold03: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold03: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold03: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold03: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold03: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold03: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold03: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold03: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold03: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold03: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold03: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold03: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold03: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold03: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold03: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold03: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold03: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold03: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold03: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold03: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold03: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold03: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold03: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold03: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold03: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold03: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold03: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold03: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold03: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold03: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold03: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold03: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold03: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold03: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold03: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold03: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold03: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold03: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold03: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold03: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold04: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold04: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold04: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold04: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold04: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold04: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold04: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold04: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold04: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold04: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold04: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold04: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold04: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold04: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold04: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold04: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold04: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold04: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold04: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold04: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold04: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold04: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold04: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold04: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold04: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold04: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold04: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold04: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold04: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold04: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold04: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold04: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold04: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold04: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold04: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold04: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold04: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold04: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold04: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold04: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold04: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold04: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold04: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold04: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold04: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold04: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold04: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold04: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold04: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold04: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold04: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold04: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold04: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold04: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold05: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold05: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold05: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold05: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold05: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold05: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold05: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold05: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold05: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold05: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold05: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold05: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold05: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold05: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold05: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold05: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold05: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold05: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold05: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold05: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold05: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold05: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold05: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold05: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold05: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold05: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold05: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold05: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold05: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold05: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold05: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold05: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold05: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold05: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold05: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold05: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold05: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold05: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold05: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold05: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold05: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold05: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold05: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold05: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold05: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold05: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold05: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold05: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold05: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold05: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold05: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold05: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold05: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold05: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold06: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold06: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold06: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold06: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold06: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold06: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold06: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold06: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold06: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold06: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold06: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold06: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold06: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold06: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold06: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold06: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold06: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold06: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold06: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold06: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold06: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold06: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold06: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold06: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold06: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold06: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold06: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold06: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold06: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold06: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold06: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold06: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold06: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold06: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold06: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold06: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold06: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold06: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold06: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold06: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold06: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold06: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold06: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold06: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold06: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold06: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold06: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold06: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold06: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold06: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold06: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold06: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold06: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold06: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold07: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold07: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold07: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold07: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold07: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold07: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold07: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold07: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold07: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold07: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold07: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold07: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold07: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold07: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold07: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold07: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold07: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold07: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold07: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold07: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold07: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold07: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold07: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold07: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold07: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold07: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold07: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold07: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold07: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold07: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold07: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold07: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold07: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold07: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold07: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold07: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold07: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold07: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold07: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold07: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold07: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold07: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold07: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold07: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold07: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold07: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold07: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold07: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold07: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold07: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold07: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold07: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold07: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold07: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold08: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold08: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold08: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold08: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold08: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold08: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold08: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold08: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold08: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold08: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold08: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold08: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold08: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold08: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold08: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold08: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold08: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold08: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold08: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold08: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold08: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold08: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold08: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold08: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold08: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold08: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold08: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold08: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold08: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold08: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold08: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold08: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold08: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold08: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold08: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold08: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold08: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold08: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold08: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold08: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold08: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold08: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold08: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold08: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold08: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold08: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold08: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold08: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold08: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold08: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold08: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold08: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold08: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold08: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold09: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold09: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold09: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold09: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold09: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold09: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold09: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold09: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold09: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold09: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold09: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold09: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold09: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold09: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold09: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold09: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold09: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold09: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold09: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold09: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold09: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold09: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold09: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold09: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold09: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold09: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold09: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold09: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold09: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold09: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold09: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold09: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold09: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold09: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold09: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold09: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold09: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold09: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold09: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold09: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold09: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold09: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold09: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold09: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold09: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold09: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold09: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold09: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold09: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold09: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold09: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold09: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold09: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold09: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## + Fold10: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold10: lambda=0e+00, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold10: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold10: lambda=1e-01, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold10: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## - Fold10: lambda=1e-04, alpha=0e+00, nrounds= 50, eta=0.3
## + Fold10: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold10: lambda=0e+00, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold10: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold10: lambda=1e-01, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold10: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## - Fold10: lambda=1e-04, alpha=1e-01, nrounds= 50, eta=0.3
## + Fold10: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold10: lambda=0e+00, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold10: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold10: lambda=1e-01, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold10: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## - Fold10: lambda=1e-04, alpha=1e-04, nrounds= 50, eta=0.3
## + Fold10: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## - Fold10: lambda=0e+00, alpha=0e+00, nrounds=100, eta=0.3
## + Fold10: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## - Fold10: lambda=1e-01, alpha=0e+00, nrounds=100, eta=0.3
## + Fold10: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## - Fold10: lambda=1e-04, alpha=0e+00, nrounds=100, eta=0.3
## + Fold10: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## - Fold10: lambda=0e+00, alpha=1e-01, nrounds=100, eta=0.3
## + Fold10: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## - Fold10: lambda=1e-01, alpha=1e-01, nrounds=100, eta=0.3
## + Fold10: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## - Fold10: lambda=1e-04, alpha=1e-01, nrounds=100, eta=0.3
## + Fold10: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## - Fold10: lambda=0e+00, alpha=1e-04, nrounds=100, eta=0.3
## + Fold10: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## - Fold10: lambda=1e-01, alpha=1e-04, nrounds=100, eta=0.3
## + Fold10: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## - Fold10: lambda=1e-04, alpha=1e-04, nrounds=100, eta=0.3
## + Fold10: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## - Fold10: lambda=0e+00, alpha=0e+00, nrounds=150, eta=0.3
## + Fold10: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
## - Fold10: lambda=1e-01, alpha=0e+00, nrounds=150, eta=0.3
```

```
## + Fold10: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## - Fold10: lambda=1e-04, alpha=0e+00, nrounds=150, eta=0.3
## + Fold10: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## - Fold10: lambda=0e+00, alpha=1e-01, nrounds=150, eta=0.3
## + Fold10: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## - Fold10: lambda=1e-01, alpha=1e-01, nrounds=150, eta=0.3
## + Fold10: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## - Fold10: lambda=1e-04, alpha=1e-01, nrounds=150, eta=0.3
## + Fold10: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## - Fold10: lambda=0e+00, alpha=1e-04, nrounds=150, eta=0.3
## + Fold10: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## - Fold10: lambda=1e-01, alpha=1e-04, nrounds=150, eta=0.3
## + Fold10: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## - Fold10: lambda=1e-04, alpha=1e-04, nrounds=150, eta=0.3
## Aggregating results
## Selecting tuning parameters
## Fitting nrounds = 50, lambda = 0.1, alpha = 1e-04, eta = 0.3 on full training set
```

```
predict(extreme, holdout, na.action = na.pass)
```

```
##   [1] 1.6944970 1.6117837 2.2052801 3.4928479 2.4057040 2.2509933 1.8367829
##   [8] 2.5512767 2.7302883 1.6080337 1.9810872 3.1080263 2.6655946 2.1404831
##  [15] 2.4718840 1.0934632 1.1631861 1.4700072 2.4707124 2.0820191 1.3239284
##  [22] 3.4295268 1.6749430 2.3504686 2.3843522 1.8520116 1.9717522 2.0813174
##  [29] 2.2336226 1.5235919 1.9715258 2.2071159 2.4875729 2.2645161 2.5892987
##  [36] 1.8450191 1.5717697 1.5180815 1.2355498 2.4492717 2.3488498 1.5884356
##  [43] 2.0346534 2.2675726 2.6940961 2.0023160 2.6715746 2.5786974 2.2089009
##  [50] 1.7462751 2.7121301 1.8065940 2.3427916 2.1368170 2.2072377 1.8635964
##  [57] 1.8230587 2.1039839 2.0545487 0.9384935 2.2082646 1.9432471 2.1749907
##  [64] 1.5624566 2.5147581 2.1197212 1.7221854 2.5985508 2.2912641 1.6204382
##  [71] 2.8804860 1.7890409 1.9520391 2.3797448 1.7427185 2.2336226 2.1845427
##  [78] 2.0385084 2.3133659 2.2455237 1.9112649 1.7744734 2.3107772 1.9043381
##  [85] 2.4443130 2.2907710 2.1246326 1.7284420 2.4906416 2.3964000 1.9919807
##  [92] 2.3548210 1.9308439 2.0419769 2.2891526 1.8873284 2.0639226 2.9643223
##  [99] 1.6778858 1.5543246 0.8909061 2.0083022 2.1389539 2.3255634 2.0241880
## [106] 1.1806684 2.5763471 2.2280841 2.0311511 2.9923077 2.2556820 1.6620353
## [113] 1.7212151 1.7813152 1.8987226 1.4518728 3.3581448 1.7822206 3.2764635
## [120] 1.8582592 2.7714114 1.9437122 2.3497307 2.1636467 2.3817205 2.2126334
## [127] 2.0465770 2.2643266 2.4741683 2.4852433 2.1836166 2.3836744 1.9525564
## [134] 1.4874690 2.2579346 1.9057978 1.9476016 1.4738175 2.6841538 1.9738107
## [141] 2.8604922 1.8250415 1.9051646 2.1658030 1.3143259 1.7912600 1.2948503
## [148] 1.8803596 1.4986722 1.0680523 2.0108275 1.4429421 1.3693070 2.0733767
## [155] 1.6485898 2.6234396 2.6002481 1.5392044 2.1154575 1.9702101 2.4354539
## [162] 3.7957606 1.9769295 1.3234624 2.3494325 2.0458307 1.5841187 1.7181900
## [169] 1.6890155 2.0365078 2.9755836 3.1724038 1.8599679 1.6180013 2.0730026
## [176] 1.5235919 2.3225303 2.0499773 1.5323993 1.5443909 1.7594185 2.1415913
## [183] 2.9545164 1.1679585 1.9943482 2.1116252 1.6561464 2.0558076 1.7695078
## [190] 1.5838197 1.8301274 2.1252351 2.8206496 2.2879620 1.3853714 2.0570476
## [197] 1.5505830 2.3497703 1.5622966 1.5529174 1.0122566 1.9810326 2.4670932
## [204] 1.8929286 1.5537291 2.7212007 1.7833291 2.0674901 2.5265610 1.8304470
## [211] 1.0107602 1.3679278 2.5116100 3.2227278 1.6350497 1.9629588 1.8844577
## [218] 1.1802627 1.5196164 1.0680523 2.2837069 2.4645686 3.0929368 2.1461434
## [225] 2.6082964 2.8020456 2.1352096 2.7720327 1.3562844 1.5931131 2.2842863
## [232] 1.6421077 1.9505115 2.6907041 2.0193369 1.7809763 2.2490141 2.0499773
```

```
## [239] 2.0179741 2.2275434 1.8978244 1.9112649 2.0993843 0.9616481 2.2475691
## [246] 1.5600419 1.8701262 1.4969221 1.5655675 1.9515821
```

Visualization

```
summary(resamples(list(olsr, elastic, support, extreme)))
```

```
##
## Call:
## summary.resamples(object = resamples(list(olsr, elastic, support, extreme)))
##
## Models: Model1, Model2, Model3, Model4
## Number of resamples: 10
##
## MAE
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Model1 0.6080851 0.6496338 0.6581339 0.6560583 0.6701995 0.6952174    0
## Model2 0.5605979 0.6221854 0.6381640 0.6250704 0.6417653 0.6613737    0
## Model3 0.6011652 0.6142458 0.6261845 0.6349516 0.6556160 0.6923870    0
## Model4 0.6366264 0.6985805 0.7806560 0.7479222 0.7931389 0.8415343    0
##
## RMSE
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Model1 0.7984674 0.8298836 0.8653742 0.8513884 0.8733618 0.8766919    0
## Model2 0.7822162 0.8294483 0.8404518 0.8318916 0.8478310 0.8591986    0
## Model3 0.8041186 0.8354252 0.8429469 0.8469226 0.8624834 0.8767677    0
## Model4 0.8318977 0.9134514 0.9572803 0.9446273 0.9924128 1.0344237    0
##
## Rsquared
##                Min.      1st Qu.      Median       Mean    3rd Qu.       Max.
## Model1 1.392038e-03 0.0045181666 0.013514115 0.02229439 0.02834600 0.07137373
## Model2 3.164320e-04 0.0073113087 0.024925590 0.03196934 0.05496261 0.07715496
## Model3 3.382984e-04 0.0015852676 0.006100190 0.01280118 0.01952833 0.04816227
## Model4 9.409172e-05 0.0003687839 0.005089432 0.01577306 0.02856751 0.06246679
##        NA's
## Model1    0
## Model2    0
## Model3    0
## Model4    0
```

```
#dotplot(resamples(list(olsr, elastic, support, extreme)), metric="ROC")
```

Using the above output, we can easily compare the 4 different models. Looking at the RMSE and the R squared values, we can see that the models are relatively close in performance. Looking just at the RMSE values, it looks like overall, model 2 seems to be performing the best, with smaller RMSEs. In running the models, I noticed that the extreme gradient regression model look abnormally long but for relatively similar performance to the other models.