# Supervised

*2019-5-15*

```
heart_disease = read_csv("./heart.csv") %>%
    mutate(target = ifelse(target==1, 0, 1)) %>%
    mutate(target=as.factor(target)) %>%
    mutate(target=as.factor(ifelse(target==0, "absence", "presence")))%>%
    mutate(target = relevel(target, "presence"))
```

```
## Parsed with column specification:
## cols(
##   age = col_double(),
##   sex = col_double(),
##   cp = col_double(),
##   trestbps = col_double(),
##   chol = col_double(),
##   fbs = col_double(),
##   restecg = col_double(),
##   thalach = col_double(),
##   exang = col_double(),
##   oldpeak = col_double(),
##   slope = col_double(),
##   ca = col_double(),
##   thal = col_double(),
##   target = col_double()
## )
```

```
#  %>%   arrange(-as.numeric(target))
#set.seed(1)
#trRows = createDataPartition(heart_disease$target, p = .75, list = FALSE)
#train = heart_disease[trRows,]
#test = heart_disease[-trRows,]

# heart_disease2 = read_csv("..\\data\\heart.csv") %>%
#     mutate(target = ifelse(target==1, 0, 1)) %>%
#     mutate(target=as.factor(heart_disease$target))

heart_disease = heart_disease %>%
    filter(thal != 0) %>%
    mutate(sex=as.factor(sex),
           cp=as.factor(cp),
           fbs=as.factor(fbs),
           restecg=as.factor(restecg),
           exang=as.factor(exang),
           slope=as.factor(slope),
           thal=factor(thal))

model.x <- model.matrix(target~.,heart_disease)[,-1]
model.y <- heart_disease$target

# test = test %>%
#     mutate(sex=as.factor(sex),
```

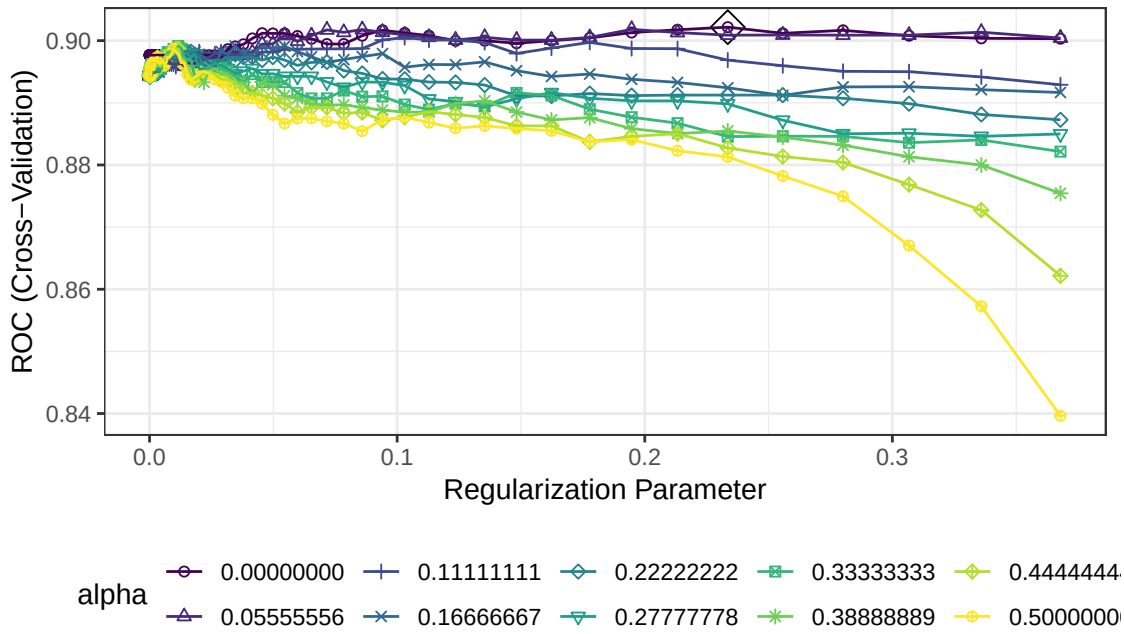
```
#             cp=as.factor(cp),
#             fbs=as.factor(fbs),
#             restecg=as.factor(restecg),
#             exang=as.factor(exang),
#             slope=as.factor(slope),
#             thal=as.factor(thal))
# test.x <- model.matrix(target~.,test)[,-1]
# test.y <- test$target
```

## Regularized logistic

```
ctrl = trainControl(method = "cv",
                    classProbs = TRUE,
                    summaryFunction = twoClassSummary)


glmnGrid <- expand.grid(.alpha = seq(0, 0.5, length = 10),
                        .lambda = exp(seq(-10,-1, length = 100)))
set.seed(1)
model.glm <- train(x = model.x,
                   y = model.y,
                   method = "glmnet",
                   tuneGrid = glmnGrid,
                   metric = "ROC",
                   trControl = ctrl)

ggplot(model.glm, highlight = T)  +
    viridis::scale_color_viridis(discrete = TRUE) +
    scale_shape_manual(values = seq(1,10))
```

```
## Scale for 'colour' is already present. Adding another scale for
## 'colour', which will replace the existing scale.
```

```
## Scale for 'shape' is already present. Adding another scale for 'shape',
## which will replace the existing scale.
```




```
model.glm$bestTune
```

```
##    alpha    lambda
## 95     0 0.2335065
```

```
glmnet = glmnet(x = model.x, y = model.y,
       family = "binomial",
       alpha = 0,
       lambda = 0.1946867)
broom::tidy(glmnet)
```

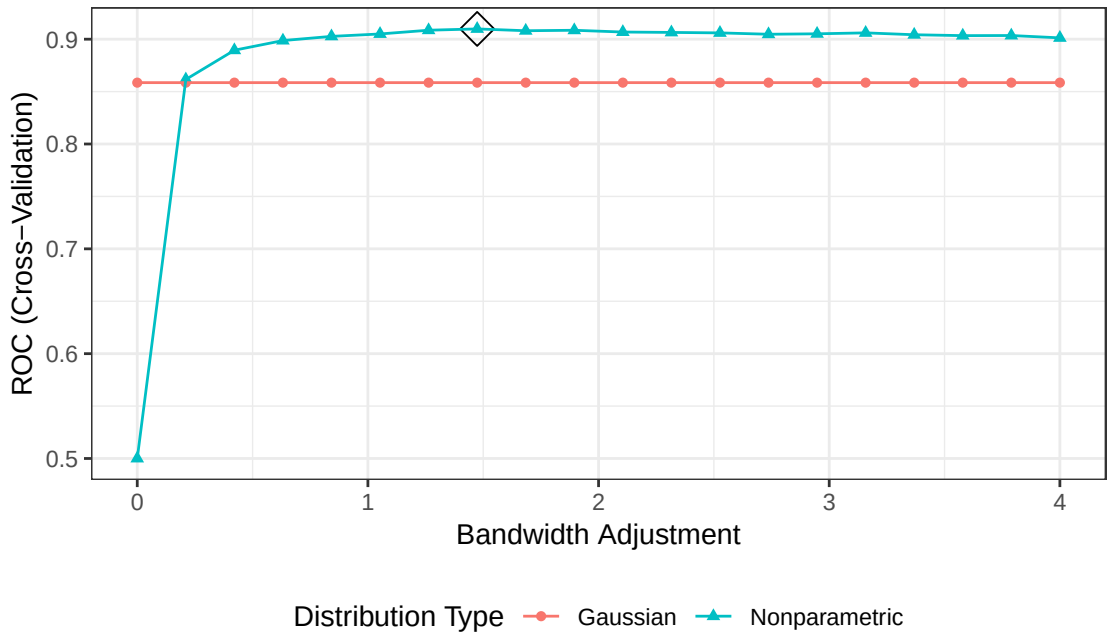
```
## # A tibble: 19 x 5
##    term         step estimate lambda dev.ratio
##    <chr>       <dbl>    <dbl>  <dbl>     <dbl>
##  1 (Intercept)     1  0.624    0.195     0.431
##  2 age             1 -0.00885  0.195     0.431
##  3 sex1            1 -0.462    0.195     0.431
##  4 cp1             1  0.385    0.195     0.431
##  5 cp2             1  0.586    0.195     0.431
##  6 cp3             1  0.524    0.195     0.431
##  7 trestbps        1 -0.00476  0.195     0.431
##  8 chol            1 -0.00120  0.195     0.431
##  9 fbs1            1  0.0693   0.195     0.431
## 10 restecg1        1  0.246    0.195     0.431
## 11 restecg2        1 -0.198    0.195     0.431
## 12 thalach         1  0.00952  0.195     0.431
## 13 exang1          1 -0.522    0.195     0.431
## 14 oldpeak         1 -0.199    0.195     0.431
## 15 slope1          1 -0.293    0.195     0.431
## 16 slope2          1  0.290    0.195     0.431
## 17 ca              1 -0.306    0.195     0.431
## 18 thal2           1  0.527    0.195     0.431
## 19 thal3           1 -0.526    0.195     0.431
```

## LDA

```
set.seed(1)
model.lda = train(x = model.x,
                  y = model.y,
                  method = "lda",
                  metric = "ROC",
                  trControl = ctrl)
```

## Naive bayes

```
set.seed(1)
nbGrid = expand.grid(usekernel = c(FALSE,TRUE),
                     fL = 1, adjust = seq(0, 4, length = 20))
model.bayes = train(x = model.x,
                    y = model.y,
                    method = "nb",
                    tuneGrid = nbGrid,
                    metric = "ROC",
                    trControl = ctrl)
save(model.bayes, file = "./bayes.rda")

ggplot(model.bayes, highlight = T)
```



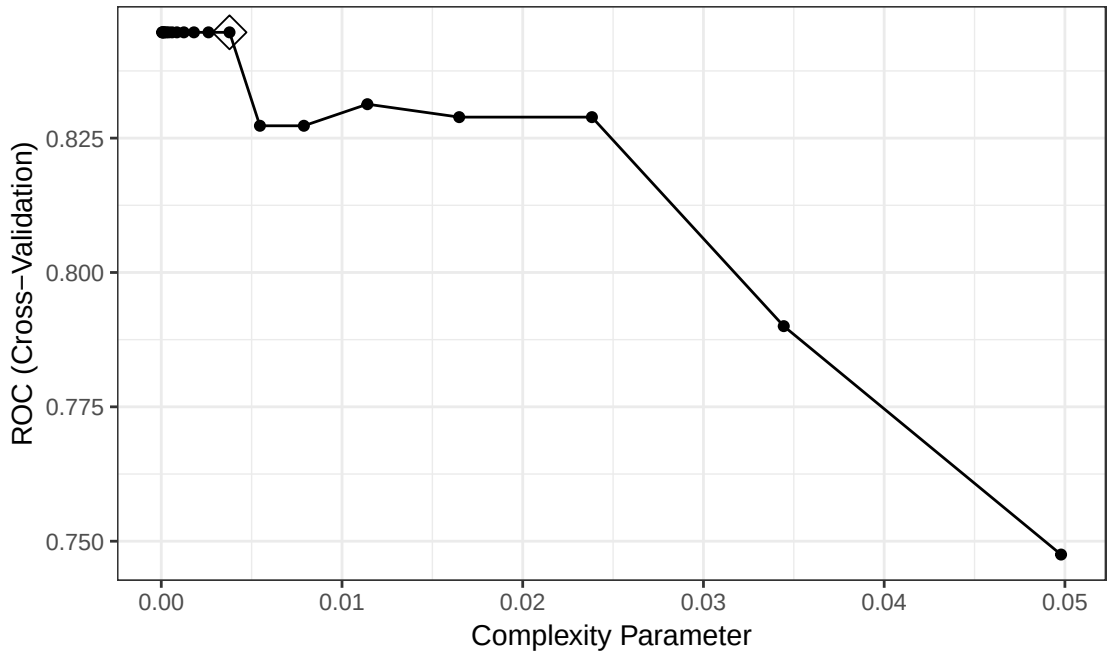

```
model.bayes$bestTune
```

```
##    fL usekernel   adjust
## 28  1      TRUE 1.473684
```

##Tree

```
set.seed(1)
tree.class <- train(model.x, model.y,
```

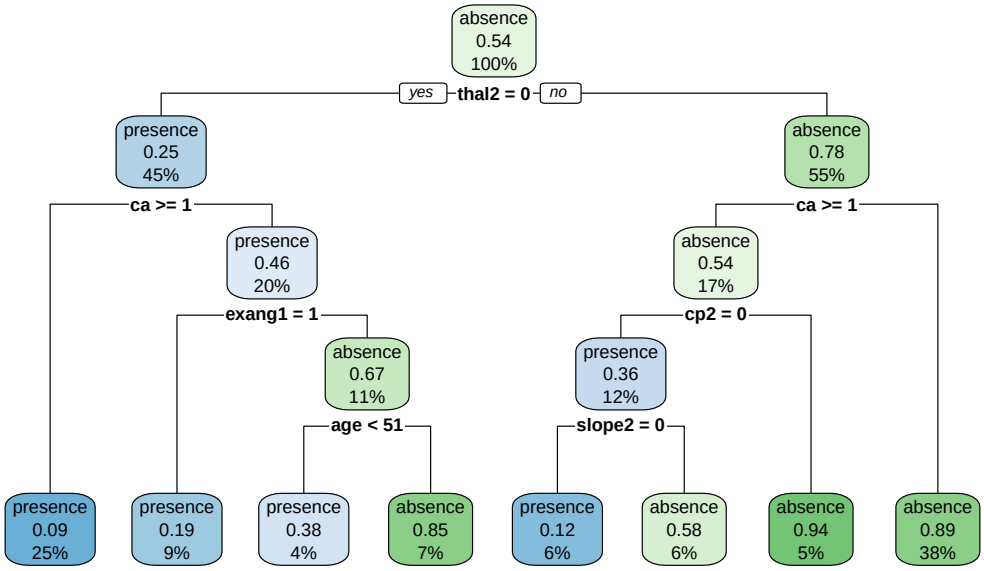
```
                    method = "rpart",
                    tuneGrid = data.frame(cp = exp(seq(-10,-3, len = 20))),
                    trControl = ctrl,
                    metric = "ROC")
ggplot(tree.class, highlight = TRUE)
```




```
tree.class$bestTune
```

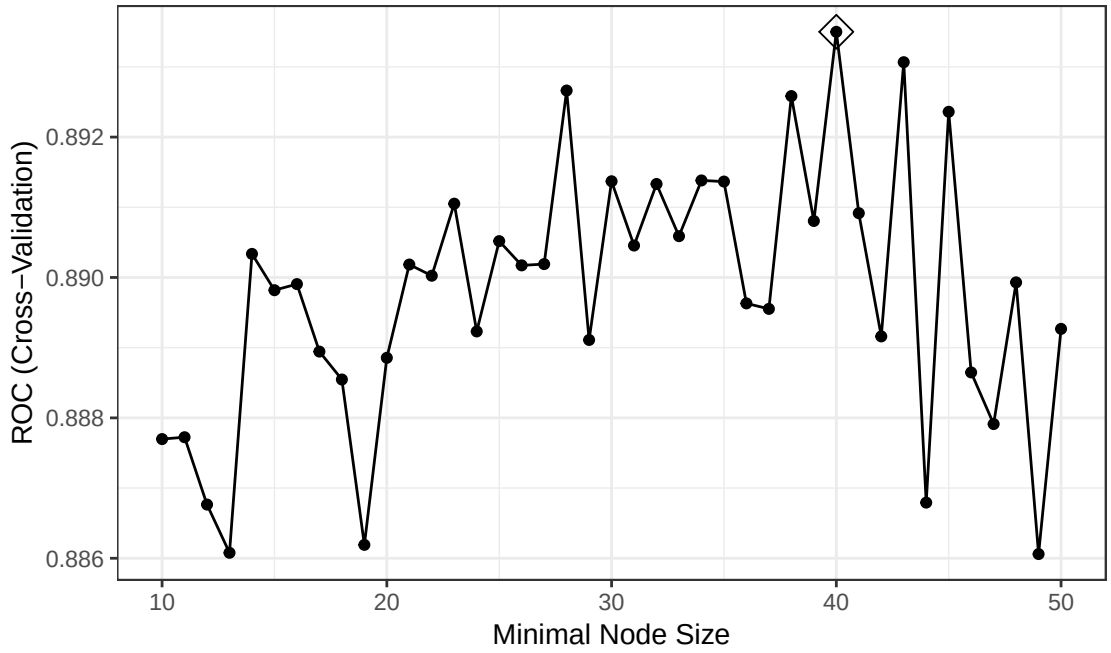
```
##             cp
## 13 0.003776539
```

```
rpart.plot(tree.class$finalModel)
```




##Bagging

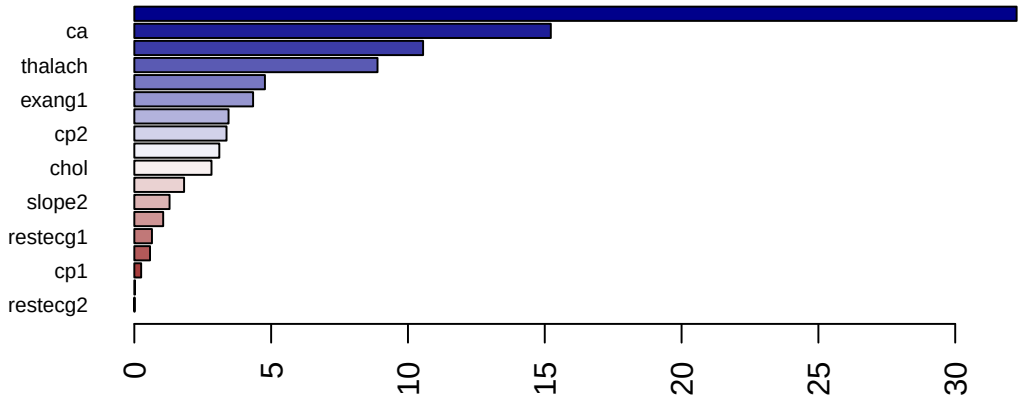
```
bagging.grid <- expand.grid(mtry = 18,
                            splitrule = "gini",
                            min.node.size = 10:50)

set.seed(1)
bagging.class <- train(model.x, model.y,
                method = "ranger",
                tuneGrid = bagging.grid,
                metric = "ROC",
                trControl = ctrl,
                importance = "impurity")
save(bagging.class, file = "./bagging.rda")

ggplot(bagging.class, highlight = TRUE)
```




```
bagging.class$bestTune
```

```
##    mtry splitrule min.node.size
## 31   18      gini            40
```

```
barplot(sort(ranger::importance(bagging.class$finalModel),
             decreasing = FALSE),
las = 2, horiz = TRUE, cex.names = 0.7,
col = colorRampPalette(colors = c("darkred","white","darkblue"))(18))
```

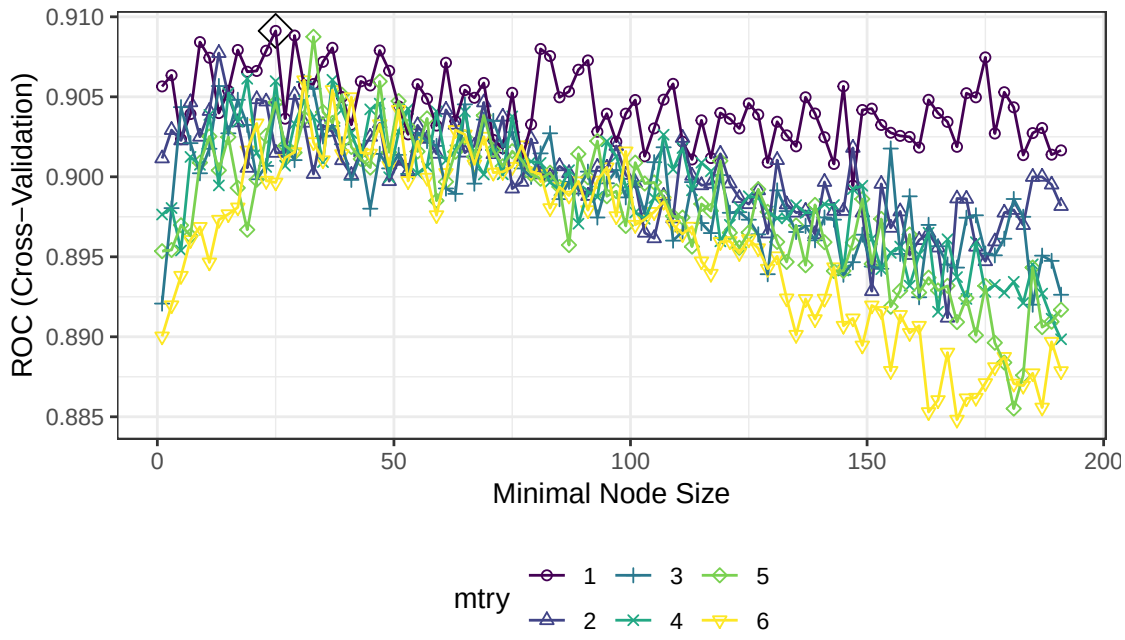



##Random Forest

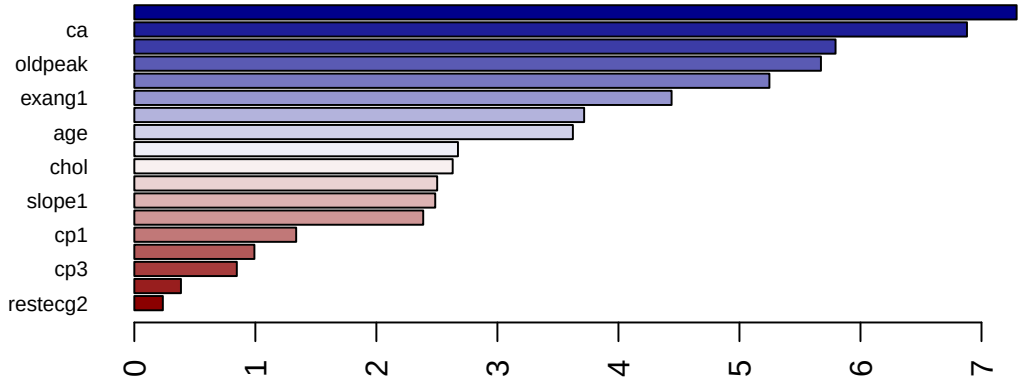
```
rf.grid <- expand.grid(mtry = 1:6,
                       splitrule = "gini",
                       min.node.size = seq(1,191, by = 2))

set.seed(1)
rf.class <- train(model.x, model.y,
                  method = "ranger",
                  tuneGrid = rf.grid,
                  metric = "ROC",
                  trControl = ctrl,
                  importance = "impurity")
save(rf.class, file = "./rf.rda")

rf.class$bestTune
```

```
##    mtry splitrule min.node.size
## 13    1      gini            25
```

```
ggplot(rf.class, highlight = TRUE) +
    viridis::scale_color_viridis(discrete = TRUE) +
    scale_shape_manual(values = seq(1,7))
```




```
barplot(sort(ranger::importance(rf.class$finalModel), decreasing = FALSE),
las = 2, horiz = TRUE, cex.names = 0.7,
col = colorRampPalette(colors = c("darkred","white","darkblue"))(18))
```




##Boosting

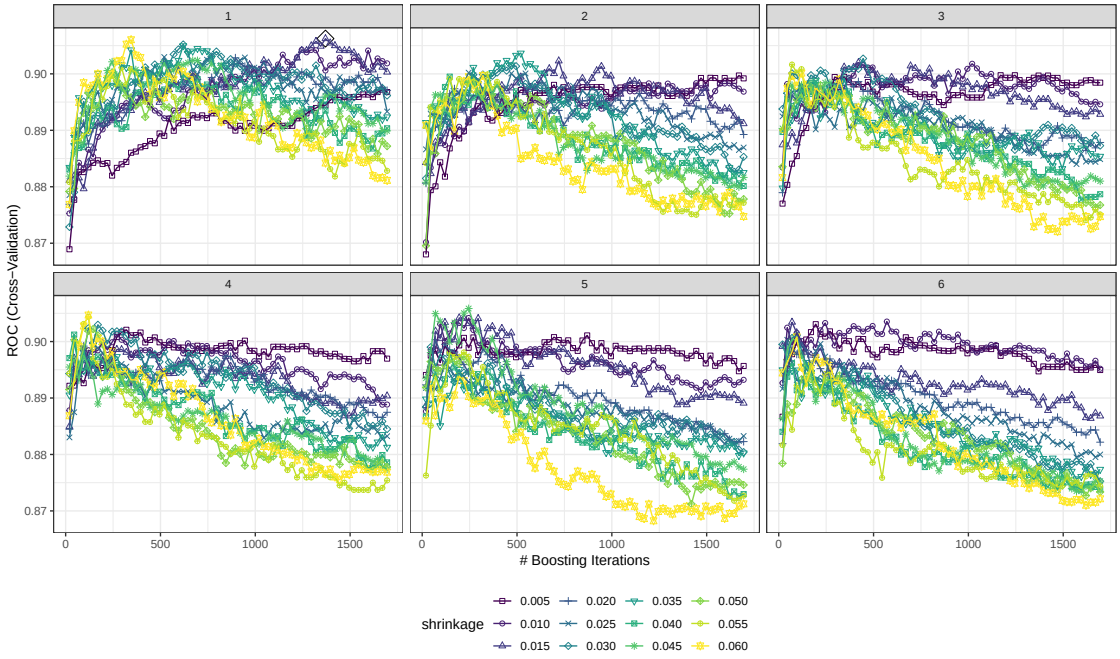
```
boost.grid <- expand.grid(n.trees = seq(20, 1700, by = 25),
                          interaction.depth = 1:6,
                          shrinkage =  seq(0.005, 0.06, by = 0.005),
                          n.minobsinnode = 1)

set.seed(1)
# Adaboost loss function
boost.class = train(model.x, model.y,
                    tuneGrid = boost.grid,
                    trControl = ctrl,
                    method = "gbm",
                    distribution = "adaboost",
                    metric = "ROC",
                    verbose = FALSE)
save(boost.class, file = "./boost.rda")

boost.class$bestTune
```

```
##     n.trees interaction.depth shrinkage n.minobsinnode
## 871    1370                 1     0.015              1
```

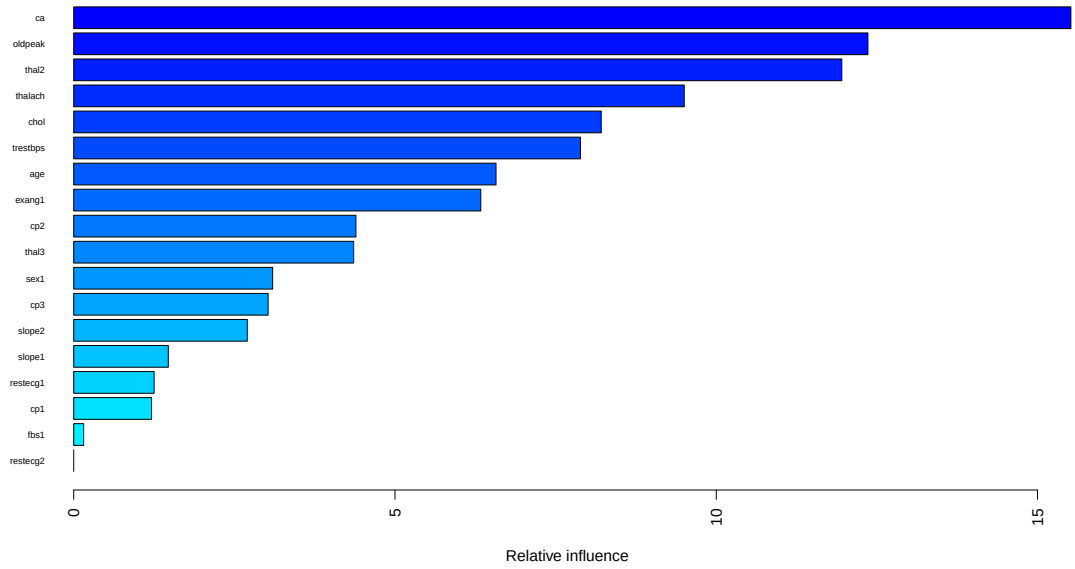
```
ggplot(boost.class, highlight = TRUE) +
    viridis::scale_color_viridis(discrete = TRUE) +
    scale_shape_manual(values = seq(0,11))
```

```
## Scale for 'colour' is already present. Adding another scale for
## 'colour', which will replace the existing scale.
```

```
## Scale for 'shape' is already present. Adding another scale for 'shape',
## which will replace the existing scale.
```




```
summary(boost.class$finalModel, las = 2, cBars = 19, cex.names = 0.6)
```

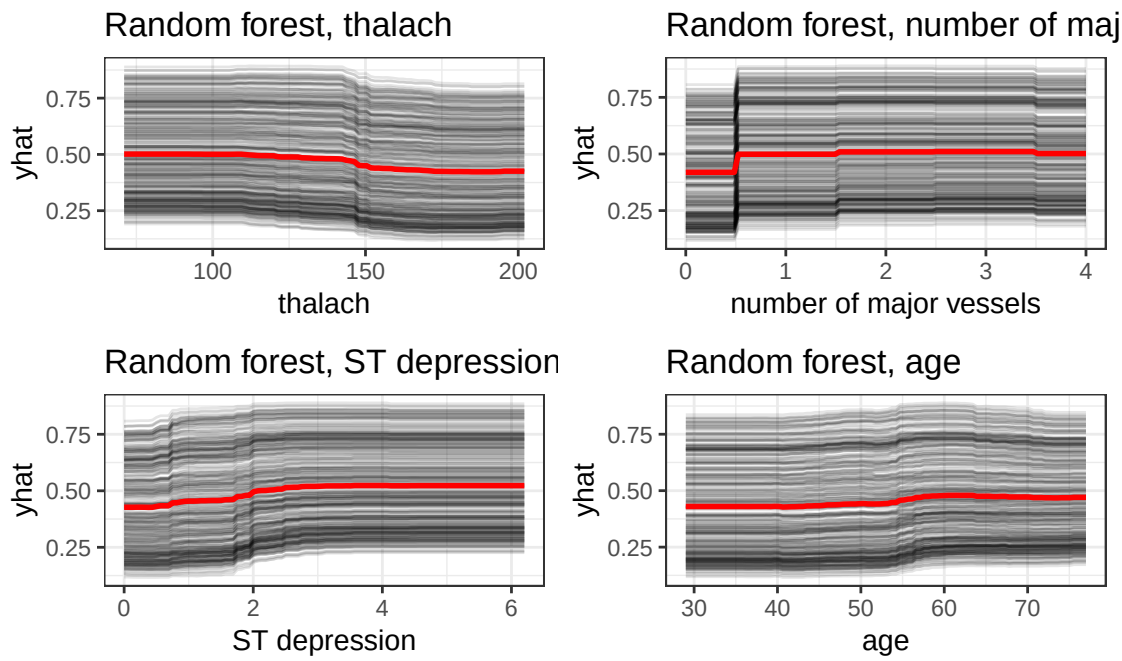



```
##                var    rel.inf
## ca              ca 15.5192989
## oldpeak   oldpeak 12.3604570
## thal2       thal2 11.9540543
## thalach   thalach  9.5013925
## chol         chol  8.2096136
## trestbps trestbps  7.8870009
## age           age  6.5723359
## exang1     exang1  6.3354175
```

```
## cp2           cp2  4.3918154
## thal3       thal3  4.3569247
## sex1         sex1  3.0962254
## cp3           cp3  3.0254006
## slope2     slope2  2.7015426
## slope1     slope1  1.4720726
## restecg1 restecg1  1.2509585
## cp1           cp1  1.2108281
## fbs1         fbs1  0.1546617
## restecg2 restecg2  0.0000000
```

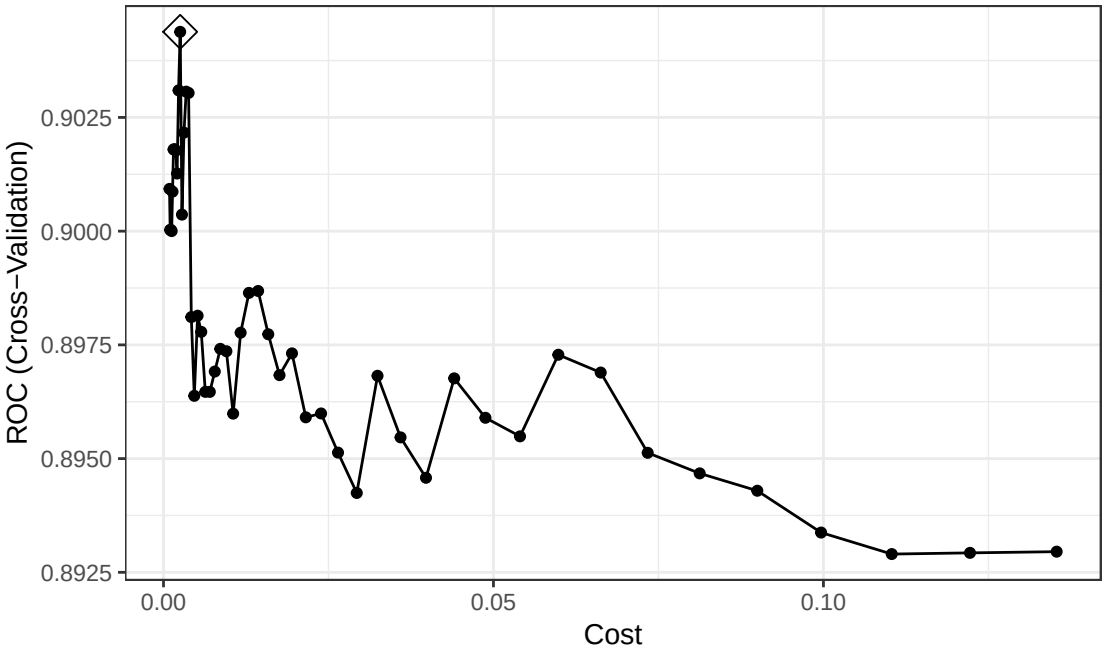
###centered ICE

```
ice_thalach.rf = rf.class %>%
    pdp::partial(pred.var = "thalach",
            grid.resolution = 100,
            ice = TRUE,
            prob = TRUE) %>%
    autoplot(train = heart_disease, alpha = .1) +
    ggtitle("Random forest, thalach")

ice_ca.rf = rf.class %>%
    pdp::partial(pred.var = "ca",
            grid.resolution = 100,
            ice = TRUE,
            prob = TRUE) %>%
    autoplot(train = heart_disease, alpha = .1,
             xlab = "number of major vessels") +
    ggtitle("Random forest, number of major vessels")

ice_oldpeak.rf =  rf.class %>%
    partial(pred.var = "oldpeak",
            grid.resolution = 100,
            ice = TRUE,
            prob = TRUE) %>%
    autoplot(train = heart_disease, alpha = .1,
             xlab = "ST depression") +
    ggtitle("Random forest, ST depression")

ice_age.rf = rf.class %>%
    pdp::partial(pred.var = "age",
            grid.resolution = 100,
            ice = TRUE,
            prob = TRUE) %>%
    autoplot(train = heart_disease, alpha = .1) +
    ggtitle("Random forest, age")

grid.arrange(ice_thalach.rf,  ice_ca.rf,
             ice_oldpeak.rf,  ice_age.rf, nrow = 2)
```




## SVM ROC

```
## linear boundary
set.seed(1)
svml.fit <- train(target~.,
                  data = heart_disease,
                  method = "svmLinear2",
                  preProcess = c("center", "scale"),
                  tuneGrid = data.frame(cost = exp(seq(-7,-2,len=50))),
                  trControl = ctrl,
                  metric = "ROC")

ggplot(svml.fit, highlight = TRUE)
```



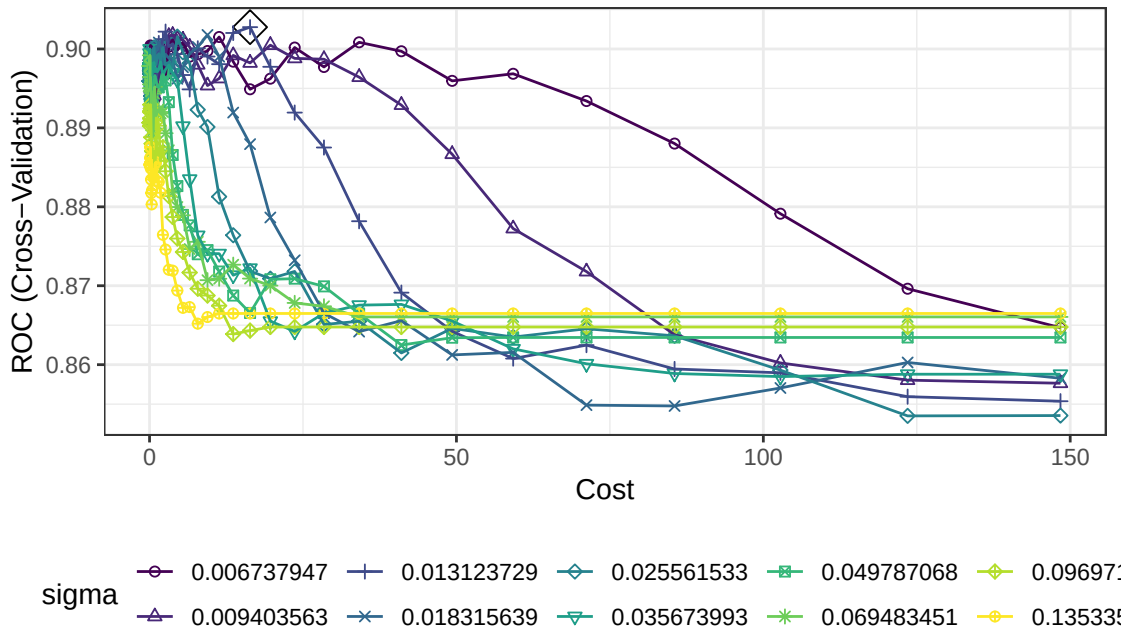

```
svml.fit$bestTune
```

```
##           cost
## 11 0.002529859
```

```
## radial kernel
svmr.grid <- expand.grid(C = exp(seq(-4,5,len=50)),
                         sigma = exp(seq(-5,-2,len=10)))
set.seed(1)
svmr.fit <- train(target~.,
                  data = heart_disease,
                  method = "svmRadial",
                  preProcess = c("center", "scale"),
                  tuneGrid = svmr.grid,
                  trControl = ctrl,
                  metric = "ROC")

ggplot(svmr.fit, highlight = TRUE) +
    viridis::scale_color_viridis(discrete = TRUE) +
    scale_shape_manual(values = seq(1,10))
```

```
## Scale for 'colour' is already present. Adding another scale for
## 'colour', which will replace the existing scale.
```

```
## Scale for 'shape' is already present. Adding another scale for 'shape',
## which will replace the existing scale.
```

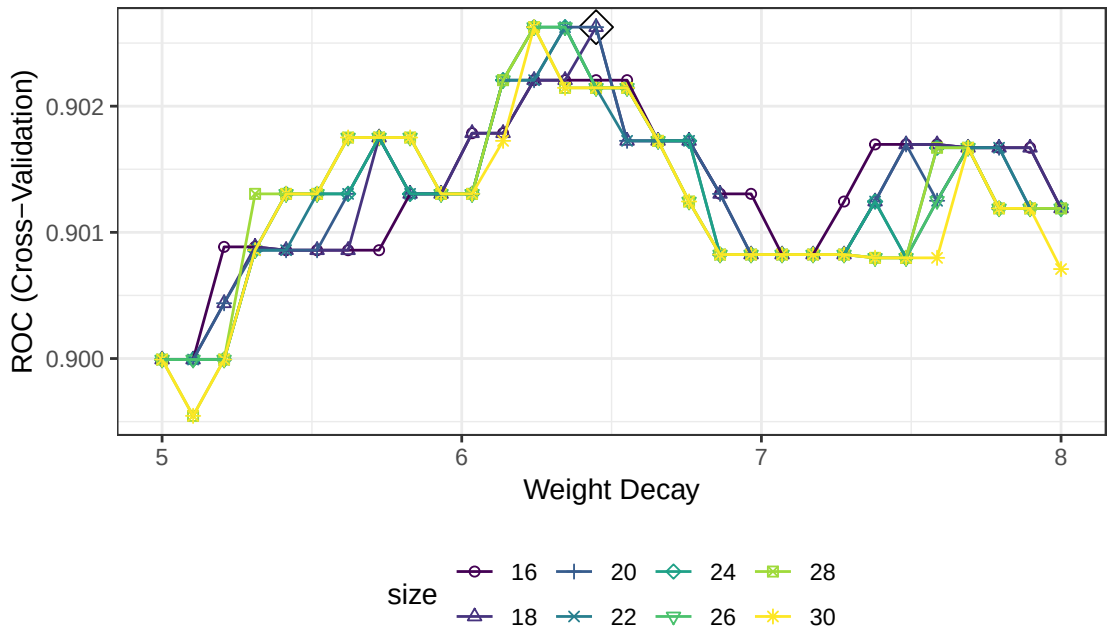



```
svmr.fit$bestTune
```

```
##          sigma        C
## 373 0.01312373 16.37766
```

## Neural network

```
nnetGrid <- expand.grid(size = seq(from = 16, to = 30, by = 2),
                        decay = seq(from = 5, to = 8, length = 30))

set.seed(1)
cnnet.fit <- train(target~.,
                   heart_disease,
                   method = "nnet",
                   tuneGrid = nnetGrid,
                   preProcess = c("center","scale"),
                   trControl = ctrl,
                   metric = "ROC",
                   trace = FALSE)

ggplot(cnnet.fit, highlight = TRUE) +
    viridis::scale_color_viridis(discrete = TRUE) +
    scale_shape_manual(values = seq(1,13))
```




```
save(cnnet.fit, file = "./cnnet.rda")
summary(cnnet.fit)
```

```
## a 18-18-1 network with 361 weights
## options were - entropy fitting  decay=6.448276
##   b->h1  i1->h1  i2->h1  i3->h1  i4->h1  i5->h1  i6->h1  i7->h1  i8->h1
##    0.00   -0.04   -0.10    0.07    0.12    0.06   -0.04   -0.03    0.01
##  i9->h1 i10->h1 i11->h1 i12->h1 i13->h1 i14->h1 i15->h1 i16->h1 i17->h1
##    0.06   -0.01    0.11   -0.12   -0.11   -0.07    0.07   -0.15    0.13
## i18->h1
##   -0.13
##   b->h2  i1->h2  i2->h2  i3->h2  i4->h2  i5->h2  i6->h2  i7->h2  i8->h2
##    0.00   -0.04   -0.10    0.07    0.12    0.06   -0.04   -0.03    0.01
##  i9->h2 i10->h2 i11->h2 i12->h2 i13->h2 i14->h2 i15->h2 i16->h2 i17->h2
##    0.06   -0.01    0.11   -0.12   -0.11   -0.07    0.07   -0.15    0.13
## i18->h2
##   -0.13
##   b->h3  i1->h3  i2->h3  i3->h3  i4->h3  i5->h3  i6->h3  i7->h3  i8->h3
##    0.00    0.04    0.11   -0.07   -0.14   -0.07    0.04    0.03   -0.01
##  i9->h3 i10->h3 i11->h3 i12->h3 i13->h3 i14->h3 i15->h3 i16->h3 i17->h3
##   -0.06    0.01   -0.11    0.13    0.12    0.08   -0.08    0.16   -0.14
## i18->h3
##    0.14
##   b->h4  i1->h4  i2->h4  i3->h4  i4->h4  i5->h4  i6->h4  i7->h4  i8->h4
##    0.00   -0.04   -0.10    0.07    0.12    0.06   -0.04   -0.03    0.01
##  i9->h4 i10->h4 i11->h4 i12->h4 i13->h4 i14->h4 i15->h4 i16->h4 i17->h4
##    0.06   -0.01    0.11   -0.12   -0.11   -0.07    0.07   -0.15    0.13
## i18->h4
##   -0.13
##   b->h5  i1->h5  i2->h5  i3->h5  i4->h5  i5->h5  i6->h5  i7->h5  i8->h5
##    0.00   -0.04   -0.10    0.07    0.12    0.06   -0.04   -0.03    0.01
##  i9->h5 i10->h5 i11->h5 i12->h5 i13->h5 i14->h5 i15->h5 i16->h5 i17->h5
```

```
##    0.06   -0.01    0.11   -0.12   -0.11   -0.07    0.07   -0.15    0.13
## i18->h5
##   -0.13
##   b->h6  i1->h6  i2->h6  i3->h6  i4->h6  i5->h6  i6->h6  i7->h6  i8->h6
##    0.00   -0.04   -0.10    0.07    0.12    0.06   -0.04   -0.03    0.01
##  i9->h6 i10->h6 i11->h6 i12->h6 i13->h6 i14->h6 i15->h6 i16->h6 i17->h6
##    0.06   -0.01    0.11   -0.12   -0.11   -0.07    0.07   -0.15    0.13
## i18->h6
##   -0.13
##   b->h7  i1->h7  i2->h7  i3->h7  i4->h7  i5->h7  i6->h7  i7->h7  i8->h7
##    0.00    0.04    0.11   -0.07   -0.14   -0.07    0.04    0.03   -0.01
##  i9->h7 i10->h7 i11->h7 i12->h7 i13->h7 i14->h7 i15->h7 i16->h7 i17->h7
##   -0.06    0.01   -0.11    0.13    0.12    0.08   -0.08    0.16   -0.14
## i18->h7
##    0.14
##   b->h8  i1->h8  i2->h8  i3->h8  i4->h8  i5->h8  i6->h8  i7->h8  i8->h8
##    0.00   -0.04   -0.10    0.07    0.12    0.06   -0.04   -0.03    0.01
##  i9->h8 i10->h8 i11->h8 i12->h8 i13->h8 i14->h8 i15->h8 i16->h8 i17->h8
##    0.06   -0.01    0.11   -0.12   -0.11   -0.07    0.07   -0.15    0.13
## i18->h8
##   -0.13
##   b->h9  i1->h9  i2->h9  i3->h9  i4->h9  i5->h9  i6->h9  i7->h9  i8->h9
##    0.00    0.04    0.11   -0.07   -0.14   -0.07    0.04    0.03   -0.01
##  i9->h9 i10->h9 i11->h9 i12->h9 i13->h9 i14->h9 i15->h9 i16->h9 i17->h9
##   -0.06    0.01   -0.11    0.13    0.12    0.08   -0.08    0.16   -0.14
## i18->h9
##    0.14
##   b->h10  i1->h10  i2->h10  i3->h10  i4->h10  i5->h10  i6->h10  i7->h10
##     0.00     0.04     0.11    -0.07    -0.14    -0.07     0.04     0.03
##  i8->h10  i9->h10 i10->h10 i11->h10 i12->h10 i13->h10 i14->h10 i15->h10
##    -0.01    -0.06     0.01    -0.11     0.13     0.12     0.08    -0.08
## i16->h10 i17->h10 i18->h10
##     0.16    -0.14     0.14
##   b->h11  i1->h11  i2->h11  i3->h11  i4->h11  i5->h11  i6->h11  i7->h11
##     0.00    -0.04    -0.10     0.07     0.12     0.06    -0.04    -0.03
##  i8->h11  i9->h11 i10->h11 i11->h11 i12->h11 i13->h11 i14->h11 i15->h11
##     0.01     0.06    -0.01     0.11    -0.12    -0.11    -0.07     0.07
## i16->h11 i17->h11 i18->h11
##    -0.15     0.13    -0.13
##   b->h12  i1->h12  i2->h12  i3->h12  i4->h12  i5->h12  i6->h12  i7->h12
##     0.00     0.04     0.11    -0.07    -0.14    -0.07     0.04     0.03
##  i8->h12  i9->h12 i10->h12 i11->h12 i12->h12 i13->h12 i14->h12 i15->h12
##    -0.01    -0.06     0.01    -0.11     0.13     0.12     0.08    -0.08
## i16->h12 i17->h12 i18->h12
##     0.16    -0.14     0.14
##   b->h13  i1->h13  i2->h13  i3->h13  i4->h13  i5->h13  i6->h13  i7->h13
##     0.00    -0.04    -0.10     0.07     0.12     0.06    -0.04    -0.03
##  i8->h13  i9->h13 i10->h13 i11->h13 i12->h13 i13->h13 i14->h13 i15->h13
##     0.01     0.06    -0.01     0.11    -0.12    -0.11    -0.07     0.07
## i16->h13 i17->h13 i18->h13
##    -0.15     0.13    -0.13
##   b->h14  i1->h14  i2->h14  i3->h14  i4->h14  i5->h14  i6->h14  i7->h14
##     0.00    -0.04    -0.10     0.07     0.12     0.06    -0.04    -0.03
##  i8->h14  i9->h14 i10->h14 i11->h14 i12->h14 i13->h14 i14->h14 i15->h14
```

```
##     0.01     0.06    -0.01     0.11    -0.12    -0.11    -0.07     0.07
## i16->h14 i17->h14 i18->h14
##    -0.15     0.13    -0.13
##   b->h15  i1->h15  i2->h15  i3->h15  i4->h15  i5->h15  i6->h15  i7->h15
##     0.00     0.04     0.11    -0.07    -0.14    -0.07     0.04     0.03
##  i8->h15  i9->h15 i10->h15 i11->h15 i12->h15 i13->h15 i14->h15 i15->h15
##    -0.01    -0.06     0.01    -0.11     0.13     0.12     0.08    -0.08
## i16->h15 i17->h15 i18->h15
##     0.16    -0.14     0.14
##   b->h16  i1->h16  i2->h16  i3->h16  i4->h16  i5->h16  i6->h16  i7->h16
##     0.00     0.04     0.11    -0.07    -0.14    -0.07     0.04     0.03
##  i8->h16  i9->h16 i10->h16 i11->h16 i12->h16 i13->h16 i14->h16 i15->h16
##    -0.01    -0.06     0.01    -0.11     0.13     0.12     0.08    -0.08
## i16->h16 i17->h16 i18->h16
##     0.16    -0.14     0.14
##   b->h17  i1->h17  i2->h17  i3->h17  i4->h17  i5->h17  i6->h17  i7->h17
##     0.00    -0.04    -0.10     0.07     0.12     0.06    -0.04    -0.03
##  i8->h17  i9->h17 i10->h17 i11->h17 i12->h17 i13->h17 i14->h17 i15->h17
##     0.01     0.06    -0.01     0.11    -0.12    -0.11    -0.07     0.07
## i16->h17 i17->h17 i18->h17
##    -0.15     0.13    -0.13
##   b->h18  i1->h18  i2->h18  i3->h18  i4->h18  i5->h18  i6->h18  i7->h18
##     0.00     0.04     0.11    -0.07    -0.14    -0.07     0.04     0.03
##  i8->h18  i9->h18 i10->h18 i11->h18 i12->h18 i13->h18 i14->h18 i15->h18
##    -0.01    -0.06     0.01    -0.11     0.13     0.12     0.08    -0.08
## i16->h18 i17->h18 i18->h18
##     0.16    -0.14     0.14
##   b->o  h1->o  h2->o  h3->o  h4->o  h5->o  h6->o  h7->o  h8->o  h9->o
##  -0.01   0.40   0.40  -0.44   0.40   0.40   0.40  -0.44   0.40  -0.44
## h10->o h11->o h12->o h13->o h14->o h15->o h16->o h17->o h18->o
##  -0.44   0.40  -0.44   0.40   0.40  -0.44  -0.44   0.40  -0.44
```

```
cnnet.fit$bestTune
```

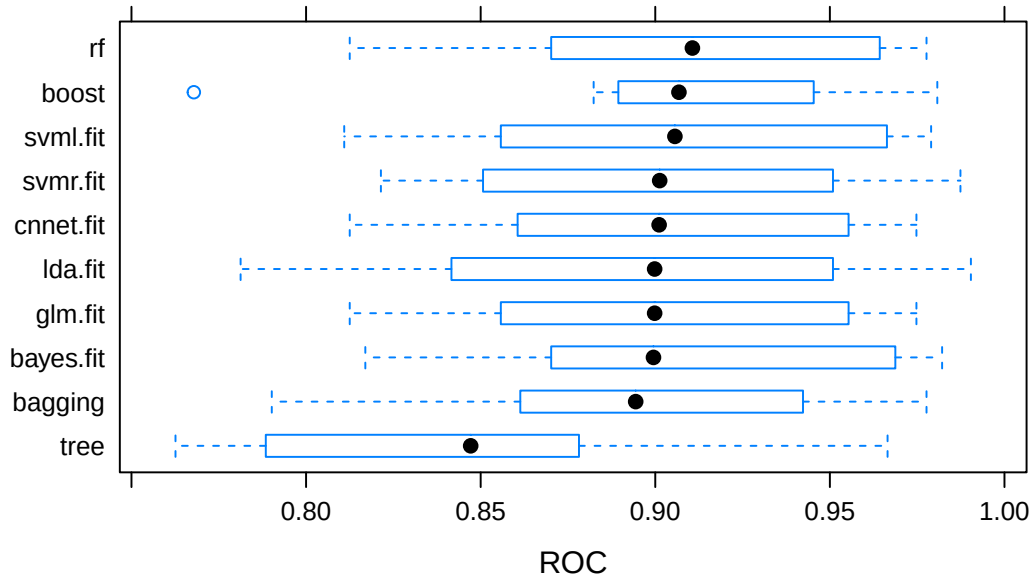
```
##    size    decay
## 45   18 6.448276
```

```
load(file = "./cnnet.rda")
load(file = "./boost.rda")
load(file = "./rf.rda")
load(file = "./bagging.rda")
load(file = "./bayes.rda")

resamp = resamples(list(
                        glm.fit = model.glm,
                        lda.fit = model.lda,
                        bayes.fit = model.bayes,
                        boost = boost.class,
                        rf = rf.class,
                        bagging = bagging.class,
                        tree = tree.class,
                        cnnet.fit = cnnet.fit,
                        svml.fit = svml.fit,
                        svmr.fit = svmr.fit
                        ))
```

```r
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: glm.fit, lda.fit, bayes.fit, boost, rf, bagging, tree, cnnet.fit, svml.fit, svmr.fit
## Number of resamples: 10
##
## ROC
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## glm.fit   0.8125000 0.8612839 0.8998162 0.9021715 0.9553571 0.9747899    0
## lda.fit   0.7812500 0.8463660 0.8998162 0.8981719 0.9497768 0.9903846    0
## bayes.fit 0.8169643 0.8725103 0.8994829 0.9097952 0.9681490 0.9821429    0
## boost     0.7678571 0.8918572 0.9067752 0.9062419 0.9434086 0.9807692    0
## rf        0.8125000 0.8747424 0.9106335 0.9091185 0.9637605 0.9776786    0
## bagging   0.7901786 0.8613445 0.8943924 0.8934995 0.9357224 0.9776786    0
## tree      0.7626050 0.7968750 0.8471386 0.8446792 0.8771008 0.9665179    0
## cnnet.fit 0.8125000 0.8641827 0.9010989 0.9026261 0.9553571 0.9747899    0
## svml.fit  0.8109244 0.8605769 0.9056238 0.9043815 0.9658310 0.9789916    0
## svmr.fit  0.8214286 0.8567590 0.9012605 0.9027614 0.9497768 0.9873950    0
##
## Sens
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## glm.fit   0.5714286 0.6923077 0.7857143 0.7950549 0.9065934 1.0000000    0
## lda.fit   0.5714286 0.6978022 0.7857143 0.7725275 0.8310440 1.0000000    0
## bayes.fit 0.6428571 0.7857143 0.8159341 0.8258242 0.9038462 1.0000000    0
## boost     0.6153846 0.6978022 0.7857143 0.7879121 0.8571429 1.0000000    0
## rf        0.6428571 0.6923077 0.7142857 0.7659341 0.8310440 1.0000000    0
## bagging   0.6428571 0.6978022 0.7500000 0.7659341 0.8310440 0.9285714    0
## tree      0.6153846 0.7280220 0.7774725 0.7725275 0.8392857 0.9285714    0
## cnnet.fit 0.5714286 0.6923077 0.7857143 0.7950549 0.9065934 1.0000000    0
## svml.fit  0.5714286 0.6552198 0.7500000 0.7653846 0.8543956 1.0000000    0
## svmr.fit  0.5384615 0.6401099 0.7857143 0.7653846 0.8571429 1.0000000    0
##
## Spec
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## glm.fit   0.7500000 0.8152574 0.8786765 0.8838235 0.9411765 1.0000000    0
## lda.fit   0.7500000 0.8152574 0.9099265 0.8838235 0.9411765 1.0000000    0
## bayes.fit 0.7500000 0.7500000 0.8235294 0.8470588 0.9264706 1.0000000    0
## boost     0.6875000 0.8281250 0.9375000 0.8775735 0.9402574 0.9411765    0
## rf        0.7058824 0.7812500 0.8786765 0.8658088 0.9402574 1.0000000    0
## bagging   0.6875000 0.7766544 0.8235294 0.8297794 0.8621324 1.0000000    0
## tree      0.5625000 0.8152574 0.8492647 0.8349265 0.8823529 0.9375000    0
## cnnet.fit 0.7500000 0.8152574 0.8786765 0.8838235 0.9411765 1.0000000    0
## svml.fit  0.7500000 0.7766544 0.8786765 0.8658088 0.9264706 1.0000000    0
## svmr.fit  0.7500000 0.8125000 0.8492647 0.8536765 0.9237132 0.9411765    0
```

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




#Comparing accuracy

##Regularized logistic

```
ctrl2 <- trainControl(method = "cv")

glmnGrid <- expand.grid(.alpha = 0,
                        .lambda = 0.2335065)

set.seed(1)
model.glm.2 <- train(x = model.x,
                     y = model.y,
                     tuneGrid = glmnGrid,
                     method = "glmnet",
                     metric = "Accuracy",
                     trControl = ctrl2)
```

##LDA

```
set.seed(1)
model.lda.2 = train(x = model.x,
                    y = model.y,
                    method = "lda",
                    metric = "Accuracy",
                    trControl = ctrl2)
```

##Naive bayes

```
set.seed(1)
nbGrid = expand.grid(usekernel = TRUE,
                     fL = 1, adjust = 1.473684)
model.bayes.2 = train(x = model.x,
                      y = model.y,
                      method = "nb",
                      tuneGrid = nbGrid,
```

```
                    metric = "Accuracy",
                    trControl = ctrl2)
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 15

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 22

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 1

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 7

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 11

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 21

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 22

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 23

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 25

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 26

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 27

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 28

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 5

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 6

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 9

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 11

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 13

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 17

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 19

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 21
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 23

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 25

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 27

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 1

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 4

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 11

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 17

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 29

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 31

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 1

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 10

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 15

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 21

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 28

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 29

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 2

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 3

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 4

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 12

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 15

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 23
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 27

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 1

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 4

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 8

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 10

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 18

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 24

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 28

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 10

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 11

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 24

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 26

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 29

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 2

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 7

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 8

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 11

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 13

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 14

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 21

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 22
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 24

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 26

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 27

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 29

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 17

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 21

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 22

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 26

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 28
```

##Tree

```
set.seed(1)
tree.class.2 <- train(model.x, model.y,
                      method = "rpart",
                      tuneGrid = data.frame(cp = 0.003776539),
                      trControl = ctrl2,
                      metric = "Accuracy")
```

##Bagging

```
bagging.grid <- expand.grid(mtry = 18,
                            splitrule = "gini",
                            min.node.size = 40)

set.seed(1)
bagging.class.2 <- train(model.x, model.y,
                  method = "ranger",
                  tuneGrid = bagging.grid,
                  metric = "Accuracy",
                  trControl = ctrl2,
                  importance = "impurity")
```

##Random Forest

```
rf.grid <- expand.grid(mtry = 1,
                       splitrule = "gini",
                       min.node.size = 25)

set.seed(1)
rf.class.2 <- train(model.x, model.y,
                    method = "ranger",
                    tuneGrid = rf.grid,
```

```
                  metric = "Accuracy",
                  trControl = ctrl2,
                  importance = "impurity")
```

##Boosting

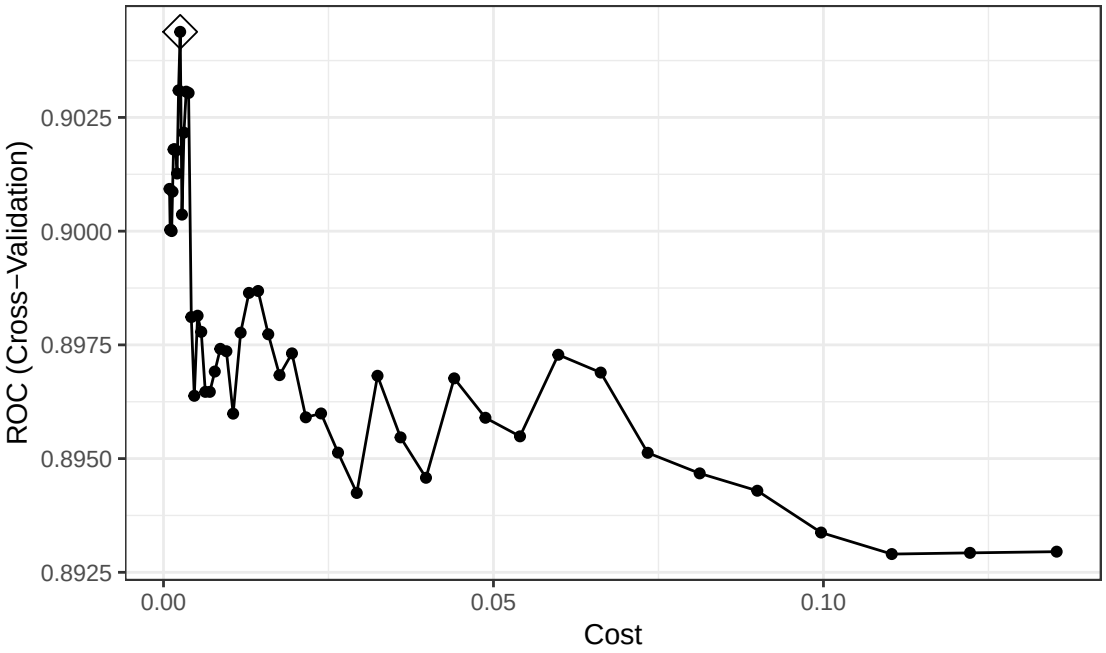
```
boost.grid <- expand.grid(n.trees = 1370,
                          interaction.depth = 1,
                          shrinkage = 0.015,
                          n.minobsinnode = 1)

set.seed(1)
# Adaboost loss function
boost.class.2 = train(model.x, model.y,
                      tuneGrid = boost.grid,
                      trControl = ctrl2,
                      method = "gbm",
                      distribution = "adaboost",
                      metric = "Accuracy",
                      verbose = FALSE)
```

## Neural network

```
nnetGrid <- expand.grid(size = 18,
                        decay = 6.448276)

set.seed(1)
cnnet.fit.2 <- train(target~.,
                     heart_disease,
                     method = "nnet",
                     tuneGrid = nnetGrid,
                     preProcess = c("center","scale"),
                     trControl = ctrl2,
                     metric = "Accuracy",
                     trace = FALSE)
```

## SVM

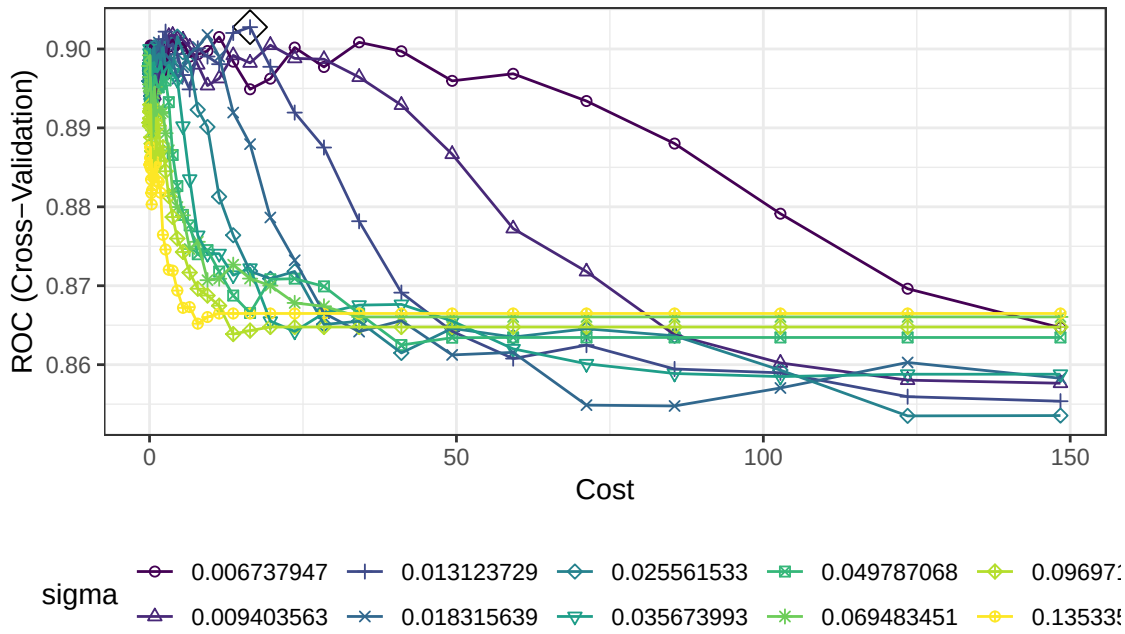
```
## linear boundary
set.seed(1)
svml.fit.2 <- train(target~.,
                    data = heart_disease,
                    method = "svmLinear2",
                    preProcess = c("center", "scale"),
                    tuneGrid = data.frame(cost = exp(seq(-7,-2,len=50))),
                    trControl = ctrl2)

ggplot(svml.fit, highlight = TRUE)
```




```
svml.fit$bestTune
```

```
##           cost
## 11 0.002529859
```

```
## radial kernel
svmr.grid <- expand.grid(C = exp(seq(-4,5,len=50)),
                         sigma = exp(seq(-5,-2,len=10)))
set.seed(1)
svmr.fit.2 <- train(target~.,
                    data = heart_disease,
                    method = "svmRadial",
                    preProcess = c("center", "scale"),
                    tuneGrid = svmr.grid,
                    trControl = ctrl2)

ggplot(svmr.fit, highlight = TRUE) +
    viridis::scale_color_viridis(discrete = TRUE) +
    scale_shape_manual(values = seq(1,10))
```




```
svmr.fit$bestTune
```

```
##          sigma        C
## 373 0.01312373 16.37766
```

```
resamp = resamples(list(
                        glm.fit = model.glm.2,
                        lda.fit = model.lda.2,
                        bayes.fit = model.bayes.2,
                        boost = boost.class.2,
                        rf = rf.class.2,
                        bagging = bagging.class.2,
                        tree = tree.class.2,
                        cnnet.fit = cnnet.fit.2,
                        svml.fit = svml.fit.2,
                        svmr.fit = svmr.fit.2
                        ))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: glm.fit, lda.fit, bayes.fit, boost, rf, bagging, tree, cnnet.fit, svml.fit, svmr.fit
## Number of resamples: 10
##
## Accuracy
##                Min.    1st Qu.     Median       Mean   3rd Qu.       Max. NA's
## glm.fit   0.7000000 0.8031609 0.8360215 0.8441416 0.9000000 0.9677419    0
## lda.fit   0.7096774 0.7732759 0.8333333 0.8340267 0.8846774 0.9677419    0
## bayes.fit 0.7000000 0.7732759 0.8526882 0.8374750 0.8916667 0.9677419    0
## boost     0.7096774 0.7482759 0.8500000 0.8341416 0.8927419 0.9655172    0
## rf        0.6774194 0.7606322 0.8032258 0.8175677 0.9066092 0.9354839    0
```

```
## bagging   0.6666667 0.7806452 0.8331479 0.8141268 0.8562291 0.9000000    0
## tree      0.6666667 0.7789210 0.8166667 0.8066704 0.8666667 0.8709677    0
## cnnet.fit 0.7000000 0.8031609 0.8360215 0.8441416 0.9000000 0.9677419    0
## svml.fit  0.6774194 0.8000000 0.8360215 0.8378124 0.9155172 0.9677419    0
## svmr.fit  0.7419355 0.8068966 0.8500000 0.8473674 0.8927419 0.9354839    0
##
## Kappa
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## glm.fit   0.3946188 0.5944980 0.6694856 0.6826138 0.8004166 0.9352818    0
## lda.fit   0.4025696 0.5423267 0.6603832 0.6617627 0.7653612 0.9352818    0
## bayes.fit 0.3946188 0.5469194 0.7004056 0.6724875 0.7814956 0.9352818    0
## boost     0.4101480 0.4940966 0.6916528 0.6626159 0.7826973 0.9307876    0
## rf        0.3404255 0.5082084 0.6012348 0.6283910 0.8082234 0.8697479    0
## bagging   0.3303571 0.5478123 0.6580195 0.6220325 0.7099677 0.7963801    0
## tree      0.3421053 0.5469316 0.6266968 0.6090815 0.7315396 0.7427386    0
## cnnet.fit 0.3946188 0.5944980 0.6694856 0.6826138 0.8004166 0.9352818    0
## svml.fit  0.3404255 0.5893826 0.6709540 0.6710978 0.8279169 0.9352818    0
## svmr.fit  0.4655172 0.6095944 0.6904463 0.6889089 0.7861955 0.8697479    0
```

```
bwplot(resamp)
```