My Document

Me

Today

Import data

```
l<-load("recovery.Rdata")

dat <- eval(parse(text = l))%>%
  mutate(study = factor(study))

head(dat)
```

```
##   id age gender race smoking height weight  bmi hypertension diabetes SBP LDL
## 1  1  56      0    1       2  170.2   78.7 27.2            0        0 120  97
## 2  2  70      1    1       1  169.6   73.1 25.4            1        0 134 112
## 3  3  57      1    1       0  168.4   77.4 27.3            1        0 131  88
## 4  4  53      0    1       0  166.7   76.1 27.4            0        0 115  87
## 5  5  59      1    1       2  173.6   70.2 23.3            0        0 127 118
## 6  6  60      1    3       1  162.8   75.1 28.4            0        0 129 104
##   vaccine severity study recovery_time
## 1       0        0     A            31
## 2       0        0     A            44
## 3       1        0     A            29
## 4       0        1     A            47
## 5       1        0     A            40
## 6       0        0     A            34
```

```
set.seed(5296)
data.1 <- dat[sample(1:10000, 2000),]
```

```
set.seed(5095)
data.2 <- dat[sample(1:10000, 2000),]
```

```
reco.data<-rbind(data.1, data.2)%>%
  unique.array()%>%
  dplyr::select(-id)

head(reco.data)
```

```
##      age gender race smoking height weight  bmi hypertension diabetes SBP LDL
## 6338  61      0    1       0  171.5   73.9 25.1            1        0 133 118
```

```
## 6800  57      1    1       0  169.7   74.5 25.9            0        0 127 120
## 4395  56      1    1       0  165.9   92.1 33.5            1        0 133 118
## 4128  68      1    3       0  177.7   84.5 26.8            1        0 134 124
## 6589  61      1    4       0  165.2   83.5 30.6            1        0 135 141
## 7384  62      1    1       0  175.0   86.1 28.1            1        0 144  88
##      vaccine severity study recovery_time
## 6338       0        0     B            24
## 6800       1        0     B            28
## 4395       0        0     B           109
## 4128       1        0     B            24
## 6589       1        0     B            98
## 7384       1        0     B             8
```

```
reco.data.bin<-reco.data%>%
  mutate(recovery_time = factor(ifelse(recovery_time>30,"long","short")))
```

Split data

```
set.seed(2023)
rowtrain <- createDataPartition(y=reco.data.bin$recovery_time, p=0.8, list=FALSE)
training_set.bin <- reco.data.bin[rowtrain,]
test_set.bin <- reco.data.bin[-rowtrain,]
x_train.bin <- model.matrix(recovery_time~.,training_set.bin)[,-1]
y_train.bin <- factor(training_set.bin$recovery_time)
x_test.bin <- model.matrix(recovery_time~.,test_set.bin)[,-1]
y_test.bin <- factor(test_set.bin$recovery_time)
contrasts(y_train.bin)
```

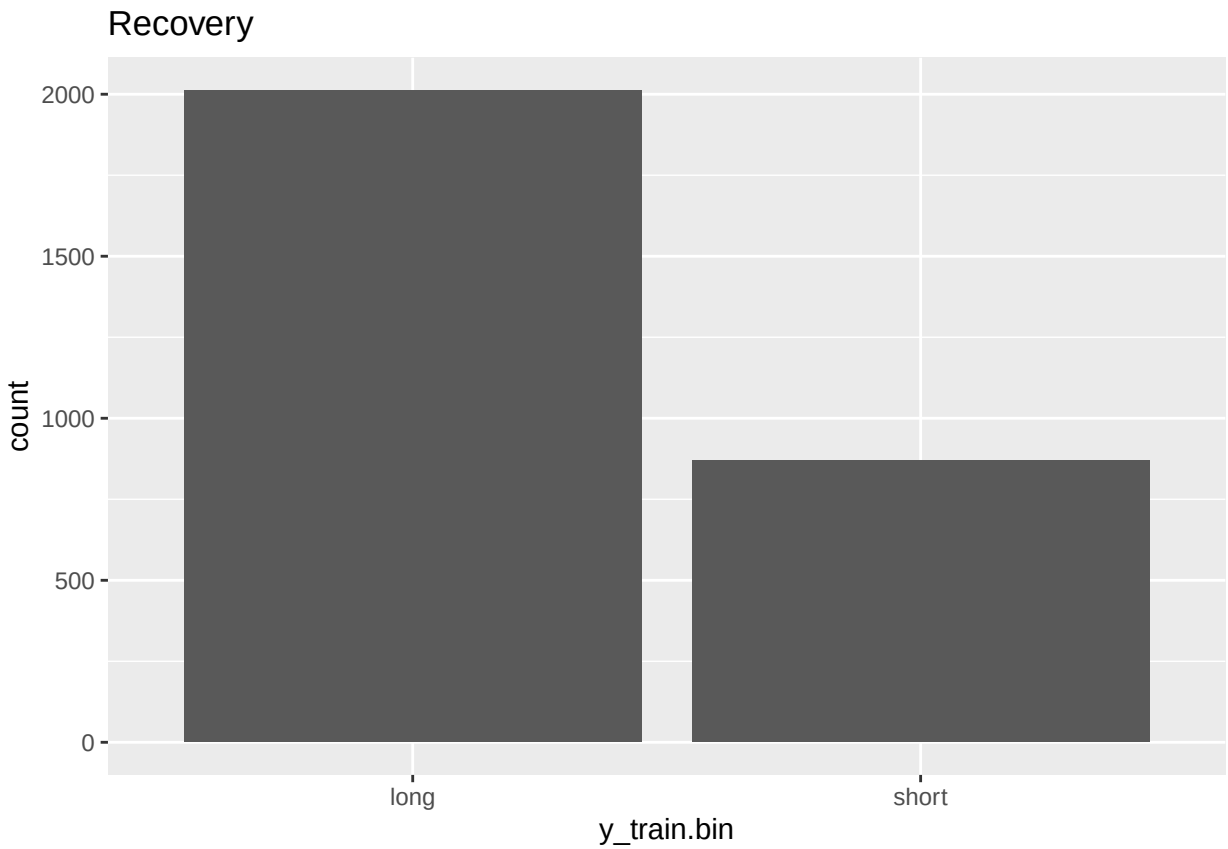
```
##       short
## long      0
## short     1
```

EDA

```
table(y_train.bin)
```

```
## y_train.bin
##  long short
##  2012   870
```

```
ggplot(training_set.bin, aes(y_train.bin)) +
  geom_bar()+
  ggtitle("Recovery")
```




```
p1 <- ggplot(training_set.bin,aes(x = recovery_time, y = age)) +
  geom_boxplot()
p2 <- ggplot(training_set.bin,aes(x = recovery_time, y = bmi)) +
  geom_boxplot()
p5 <- ggplot(training_set.bin,aes(x = recovery_time, y = height)) +
  geom_boxplot()
p6 <- ggplot(training_set.bin,aes(x = recovery_time, y = weight)) +
  geom_boxplot()
p3 <- ggplot(training_set.bin,aes(x = recovery_time, y = SBP)) +
  geom_boxplot()
p4 <- ggplot(training_set.bin,aes(x = recovery_time, y = LDL)) +
  geom_boxplot()

arrange = ggarrange(p1,p2,p5,p6, p3,p4, ncol = 3, nrow = 2)
ggsave("arrangedplot3.png", arrange)
```

```
## Saving 6.5 x 4.5 in image
```

Modeling

logistic Regression

```r
ctrl <- trainControl(method = "repeatedcv",
                     summaryFunction = twoClassSummary,
                      classProbs = TRUE)

set.seed(2023)

logit.fit <- train(x= x_train.bin,
                   y= y_train.bin,
                   method = 'glm',
                   metric = "ROC",
                   family = binomial(),
                   trControl = ctrl )
```

```r
# Performance
coef(logit.fit$finalModel)%>%knitr::kable()
```

	x
(Intercept)	77.9377809
age	-0.0014737
gender	0.2872523
race2	0.0049070
race3	0.1832813
race4	0.1183182
smoking1	-0.3491295
smoking2	-0.5142643
height	-0.4556903
weight	0.4958408
bmi	-1.5094498
hypertension	-0.2501449
diabetes	0.0040744
SBP	0.0000674
LDL	-0.0000503
vaccine	0.5584663
severity	-0.6682278
studyB	1.1348875
studyC	0.0696563

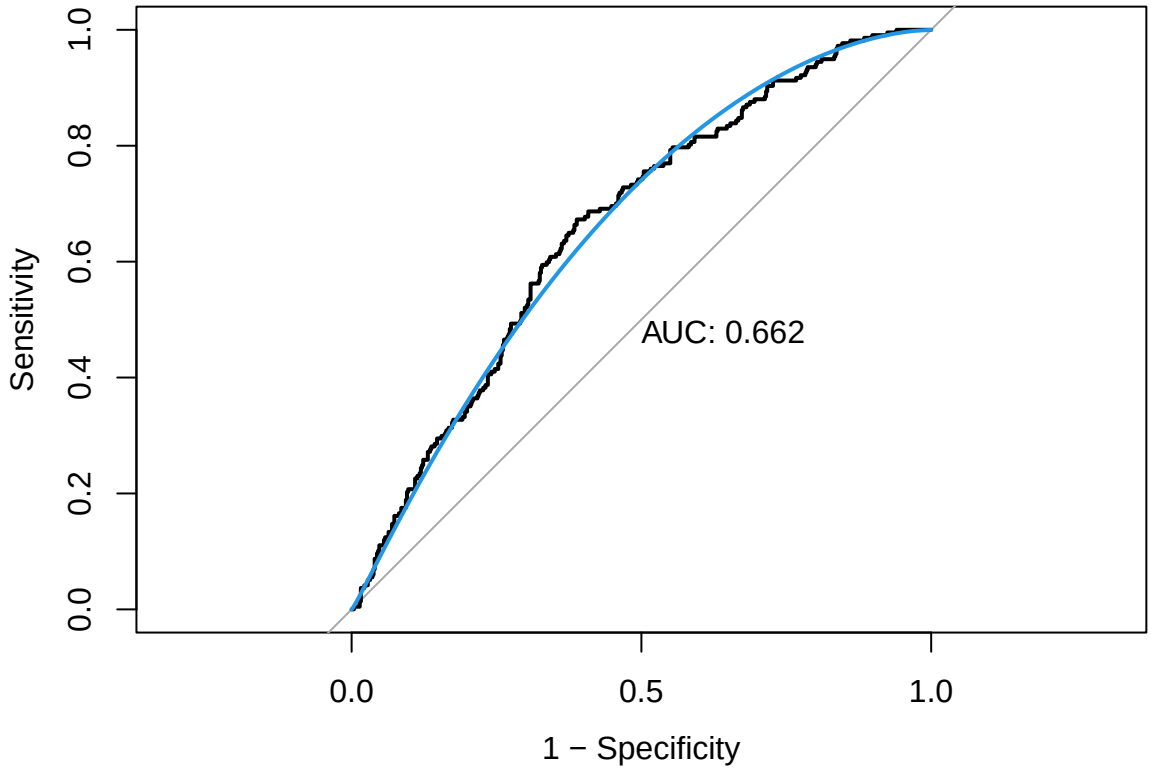
```r
# ROC Curve

pred.logit.1 <- predict(logit.fit, newdata  = x_test.bin, type = "prob")[,2]
roc.logit <- roc(y_test.bin, pred.logit.1)
```

```
## Setting levels: control = long, case = short

## Setting direction: controls < cases
```

```r
plot(roc.logit, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.logit), col = 4, add = TRUE)
```




```
# Confusion Matrix
logit.pred.2 <- predict(logit.fit, newdata  = x_test.bin)
confusionMatrix(data = logit.pred.2   ,
                y_test.bin)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction long short
##      long   447   169
##      short   55    48
##
##                Accuracy : 0.6885
##                  95% CI : (0.6532, 0.7222)
##     No Information Rate : 0.6982
##     P-Value [Acc > NIR] : 0.7299
##
##                   Kappa : 0.1312
##
##  Mcnemar's Test P-Value : 4.348e-14
##
##             Sensitivity : 0.8904
##             Specificity : 0.2212
##          Pos Pred Value : 0.7256
##          Neg Pred Value : 0.4660
```

```
##              Prevalence : 0.6982
##          Detection Rate : 0.6217
##    Detection Prevalence : 0.8567
##       Balanced Accuracy : 0.5558
##
##        'Positive' Class : long
##
```

The accuracy of logistic regression with the best tunning parameter is 0.6885

MARS

```
set.seed(2023)
mars.fit <- train(x = x_train.bin,
                  y = y_train.bin,
                  method = "earth",
                   tuneGrid = expand.grid(degree = 1:3,
                   nprune = 2:19),
                   metric = "ROC",
                   trControl = ctrl)
```

```
## 载入需要的程辑包: earth
```

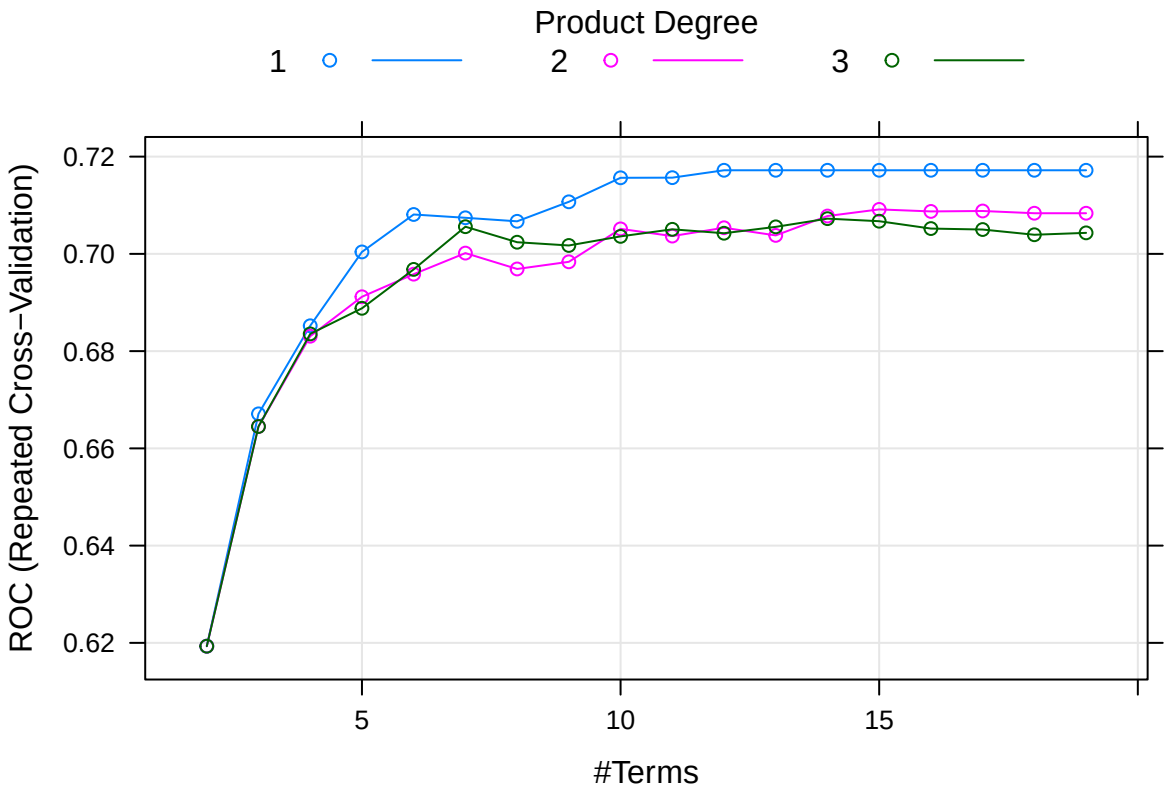
```
## 载入需要的程辑包: Formula
```

```
## 载入需要的程辑包: plotmo
```

```
## 载入需要的程辑包: plotrix
```

```
## 载入需要的程辑包: TeachingDemos
```

```
#performance
plot(mars.fit)
```




```
mars.fit$bestTune
```

```
##    nprune degree
## 11     12      1
```

```
coef(mars.fit$finalModel)%>%knitr::kable()
```

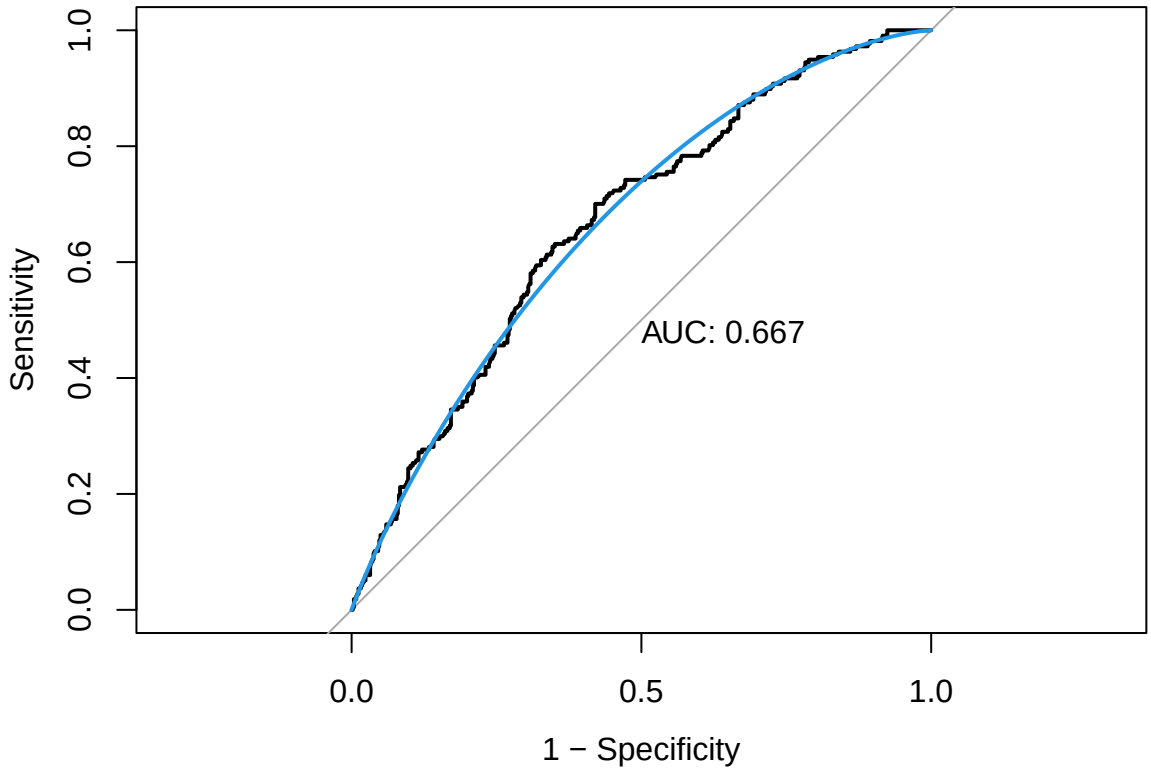
	x
(Intercept)	0.1116993
studyB	1.1310163
h(27.7-bmi)	-0.4801829
vaccine	0.5724142
severity	-0.7050607
gender	0.3084459
smoking2	-0.5517739
smoking1	-0.3567230
h(SBP-146)	-0.2366423
h(bmi-24)	-0.3634087
hypertension	-0.2360256

```
# ROC Curve
pred.mars.1 <- predict(mars.fit, newdata  = x_test.bin, type = "prob")[,2]
mars.roc <- roc(y_test.bin, pred.mars.1)
```

```
## Setting levels: control = long, case = short

## Setting direction: controls < cases
```

```
plot(mars.roc, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(mars.roc), col = 4, add = TRUE)
```

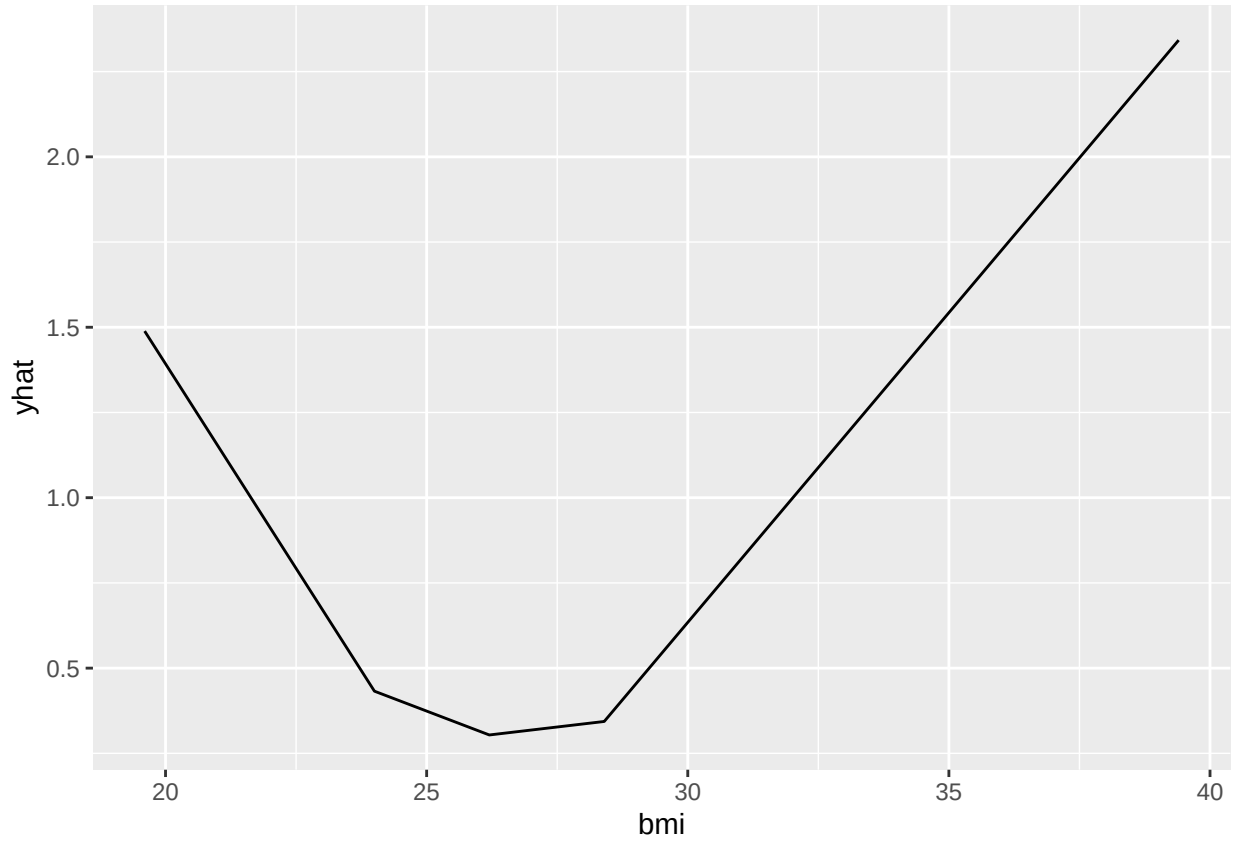



```
# Confusion Matrix
pred.mars.2 <- predict(mars.fit, newdata  = x_test.bin)
confusionMatrix(data = pred.mars.2,
                reference = y_test.bin)
```

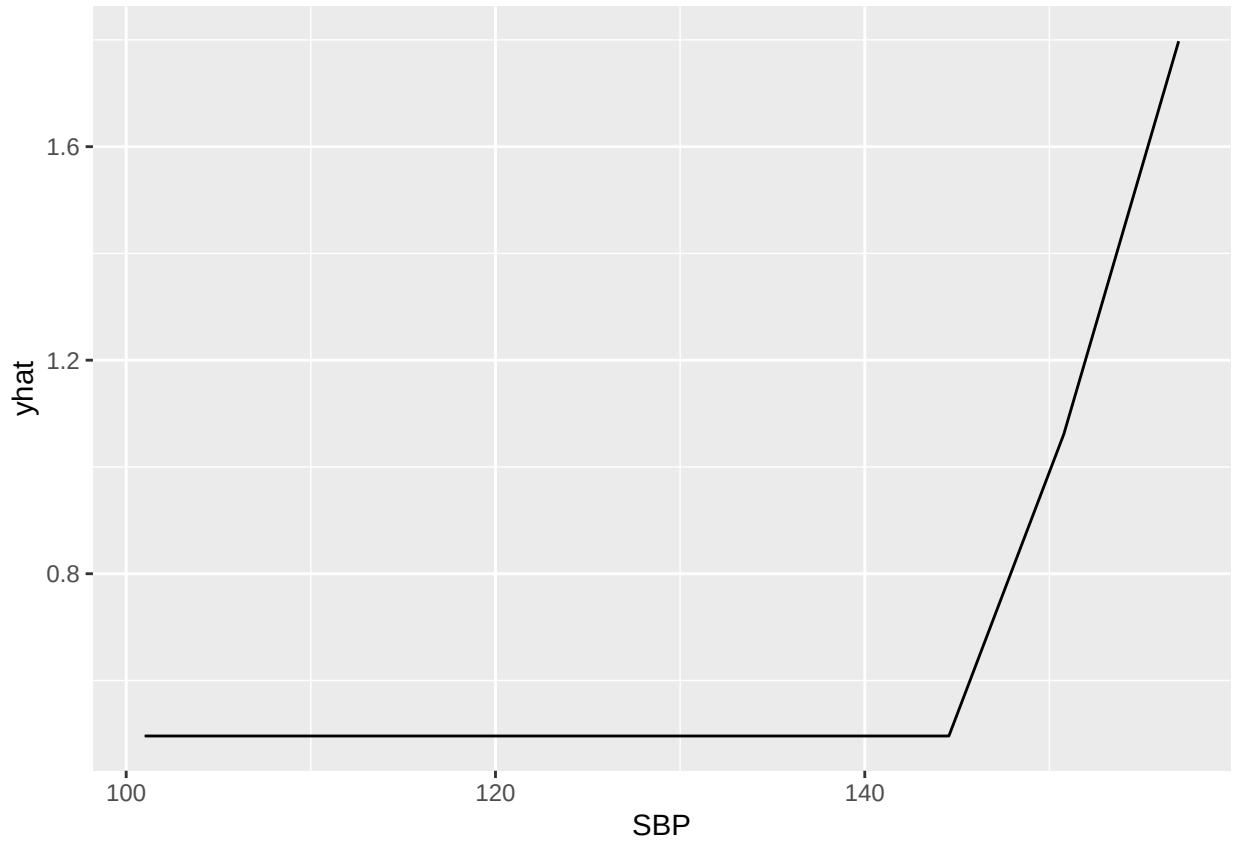
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction long short
##      long   444   160
##      short   58    57
##
##                Accuracy : 0.6968
##                  95% CI : (0.6618, 0.7302)
##     No Information Rate : 0.6982
##     P-Value [Acc > NIR] : 0.5506
##
```

```
##                  Kappa : 0.1698
##
##  Mcnemar's Test P-Value : 7.887e-12
##
##             Sensitivity : 0.8845
##             Specificity : 0.2627
##          Pos Pred Value : 0.7351
##          Neg Pred Value : 0.4957
##              Prevalence : 0.6982
##          Detection Rate : 0.6175
##    Detection Prevalence : 0.8401
##       Balanced Accuracy : 0.5736
##
##        'Positive' Class : long
##
```

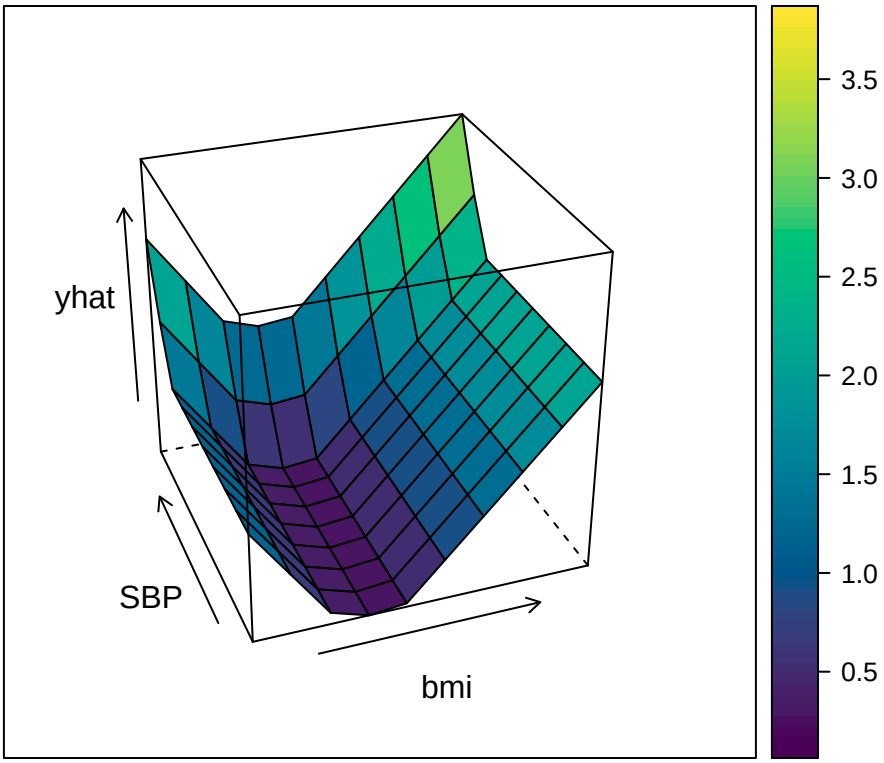
```
p1 <- pdp::partial(mars.fit, pred.var = c("bmi"), grid.resolution = 10) %>% autoplot()
p2 <- pdp::partial(mars.fit, pred.var = c("SBP"), grid.resolution = 10) %>% autoplot()
p4 <- pdp::partial(mars.fit, pred.var = c("bmi", "SBP"),
                   grid.resolution = 10) %>%
      pdp::plotPartial(levelplot = FALSE, zlab = "yhat", drape = TRUE,
                       screen = list(z = 20, x = -60))
p1
```



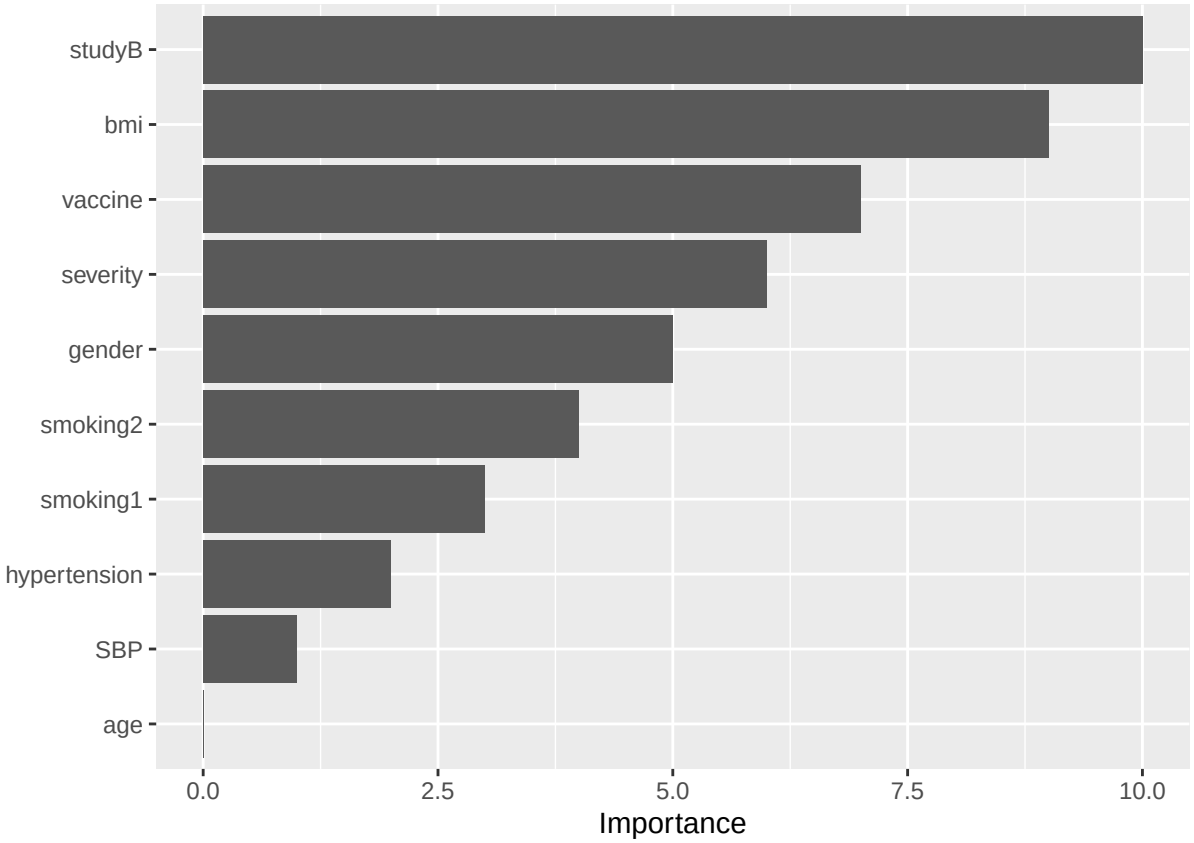
```
p2
```




```
p4
```

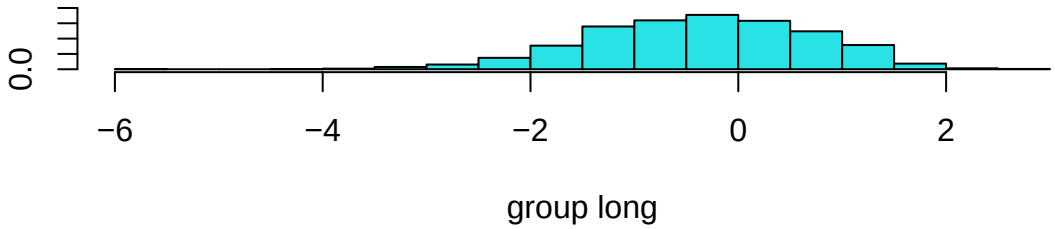



```
vip(mars.fit$finalModel)
```

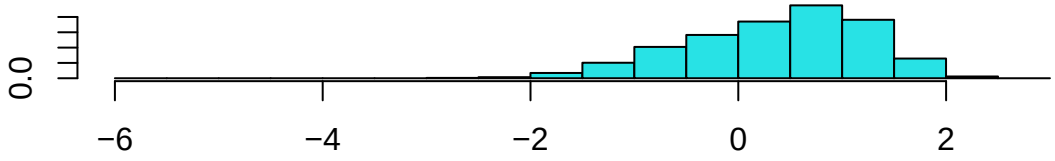


LDA

```
#LDA
lda.fit <- train(x = x_train.bin,
                 y = y_train.bin,
                 method = "lda",
                 metric = "ROC",
                 trControl = ctrl)
plot(lda(y_train.bin~x_train.bin))
```



group long



group short

```
#Performance
coef(lda.fit$finalModel)
```

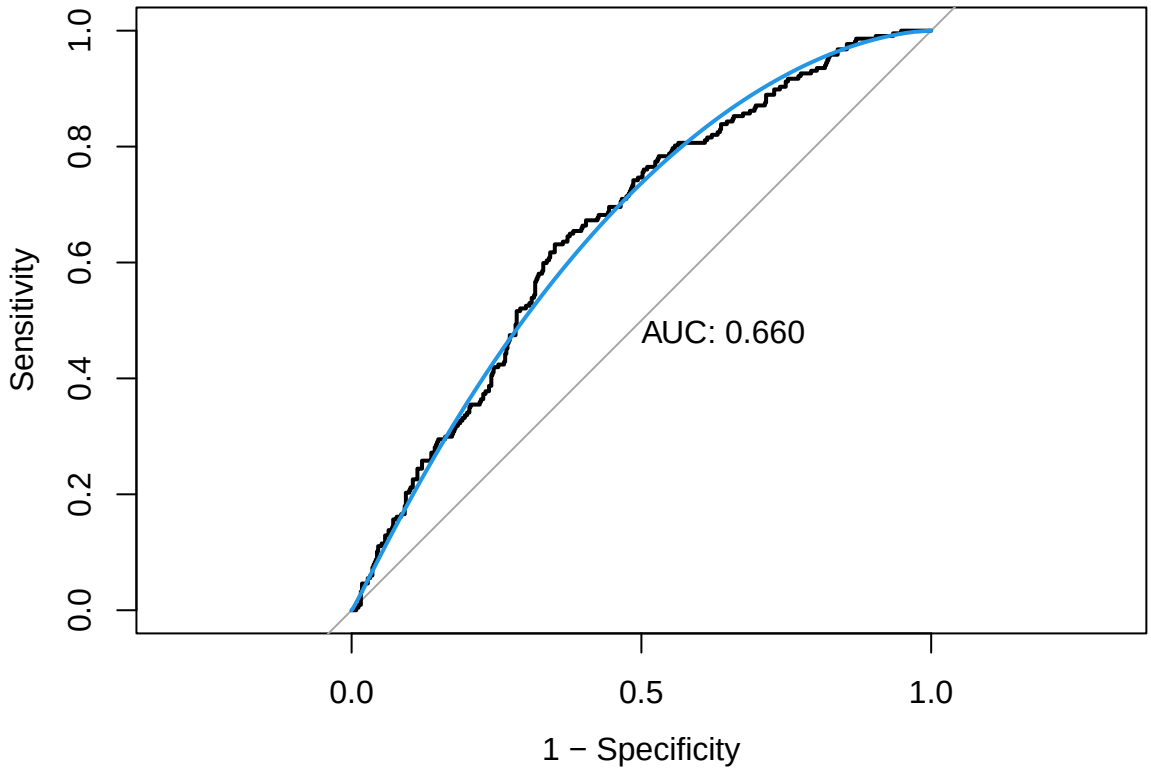
```
##                       LD1
## age           -0.0006725142
## gender         0.3916442927
## race2          0.0152252625
## race3          0.2532393459
## race4          0.1636328407
## smoking1      -0.4481450169
## smoking2      -0.6835692462
## height        -0.4834816425
## weight         0.5229986644
## bmi           -1.6020430301
## hypertension -0.3290637233
## diabetes       0.0210963168
## SBP           -0.0007270520
## LDL           -0.0001418491
## vaccine        0.7648376558
## severity      -0.8301518980
## studyB         1.5041801769
## studyC         0.0747323482
```

```
# ROC Curve
pred.lda.1 <- predict(lda.fit, newdata  = x_test.bin, type = "prob")[,2]
lda.roc <- roc(y_test.bin, pred.lda.1)
```

```
## Setting levels: control = long, case = short

## Setting direction: controls < cases
```

```
plot(lda.roc, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(lda.roc), col = 4, add = TRUE)
```



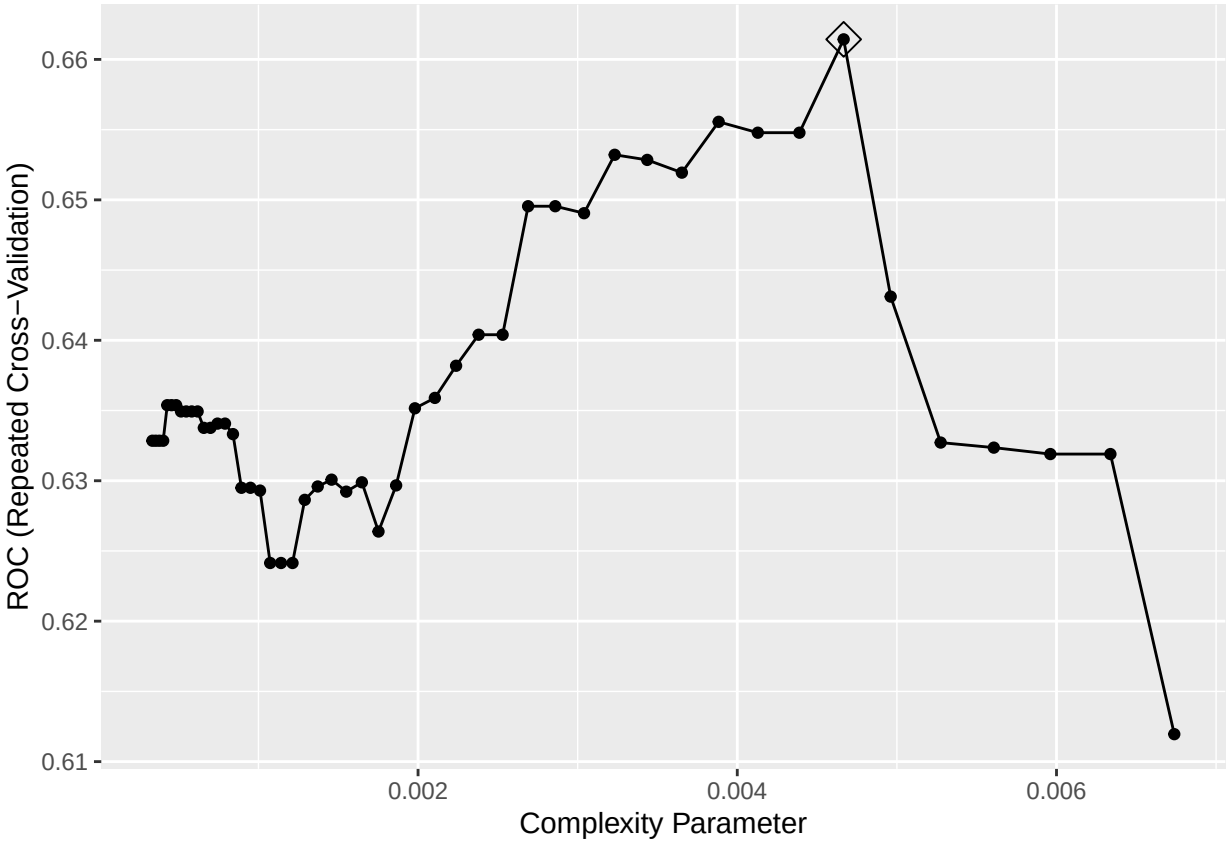

```
# Confusion Matrix
lda.pred.2 <- predict(lda.fit, newdata  = x_test.bin)
confusionMatrix(data = lda.pred.2,
                reference = y_test.bin)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction long short
##      long   452   172
##      short   50    45
##
##                Accuracy : 0.6912
##                  95% CI : (0.656, 0.7249)
##     No Information Rate : 0.6982
```

```
##     P-Value [Acc > NIR] : 0.674
##
##                   Kappa : 0.1282
##
##  Mcnemar's Test P-Value : 4.624e-16
##
##             Sensitivity : 0.9004
##             Specificity : 0.2074
##          Pos Pred Value : 0.7244
##          Neg Pred Value : 0.4737
##              Prevalence : 0.6982
##          Detection Rate : 0.6287
##    Detection Prevalence : 0.8679
##       Balanced Accuracy : 0.5539
##
##        'Positive' Class : long
##
```

Classification Tree

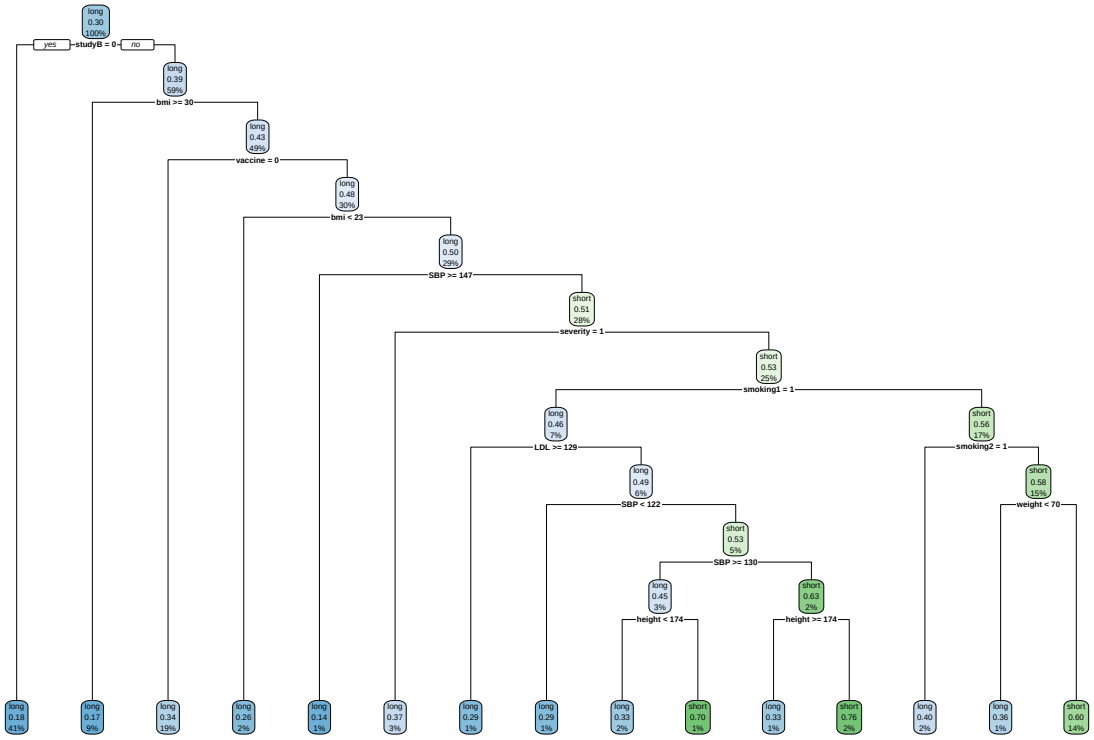
```
set.seed(2023)
rpart.fit <- train(x=x_train.bin, y = y_train.bin,
                       method = "rpart",
                       tuneGrid = data.frame(cp = exp(seq(-8,-5, len = 50))),
                       trControl = ctrl,
                       metric = "ROC")
ggplot(rpart.fit, highlight = TRUE)
```




```
rpart.fit$bestTune
```

```
##             cp
## 44 0.004666495
```

```
rpart.plot(rpart.fit$finalModel)
```



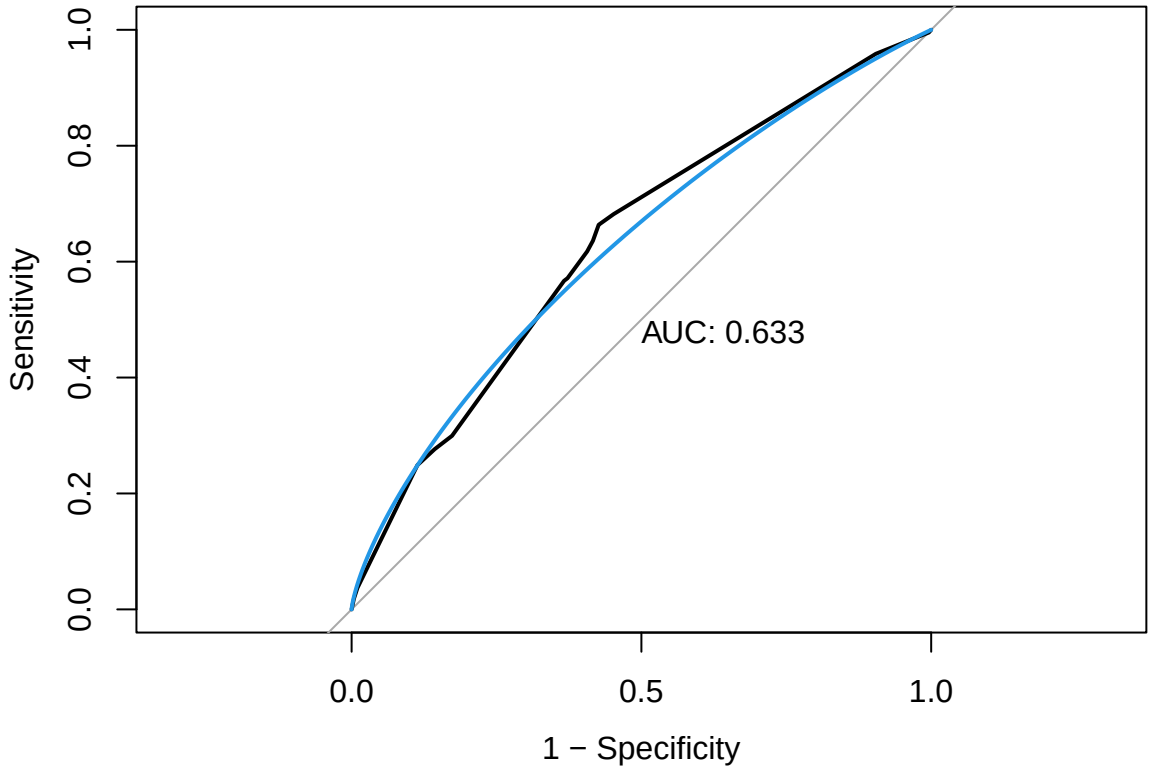

```
# Performance

# ROC Curve
pred.rpart.1 <- predict(rpart.fit, newdata  = x_test.bin, type = "prob")[,2]
rpart.roc <- roc(y_test.bin, pred.rpart.1)
```

```
## Setting levels: control = long, case = short
```

```
## Setting direction: controls < cases
```

```
plot(rpart.roc, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(rpart.roc), col = 4, add = TRUE)
```




```
# Confusion Matrix
rpart.pred.2 <- predict(rpart.fit, newdata = x_test.bin)
confusionMatrix(data = rpart.pred.2,
                reference = y_test.bin)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction long short
##      long   445   163
##      short   57    54
##
##                Accuracy : 0.694
##                  95% CI : (0.6589, 0.7275)
##     No Information Rate : 0.6982
##     P-Value [Acc > NIR] : 0.6138
##
##                   Kappa : 0.1571
##
##  Mcnemar's Test P-Value : 1.451e-12
##
##             Sensitivity : 0.8865
##             Specificity : 0.2488
##          Pos Pred Value : 0.7319
##          Neg Pred Value : 0.4865
```

```
##              Prevalence : 0.6982
##          Detection Rate : 0.6189
##    Detection Prevalence : 0.8456
##       Balanced Accuracy : 0.5677
##
##        'Positive' Class : long
##
```

Model Comparison

```
set.seed(2023)
res <- caret::resamples(list(Logit = logit.fit, Mars = mars.fit, lda = lda.fit, rpart=rpart.fit),metric
                 method = "cv",index = createFolds(trainData$binary_outcome, k = 10))

summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: Logit, Mars, lda, rpart
## Number of resamples: 10
##
## ROC
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Logit 0.6423694 0.6762452 0.6944587 0.6949499 0.7155744 0.7381928    0
## Mars  0.6708774 0.6934223 0.7138732 0.7171922 0.7356822 0.7727893    0
## lda   0.6523703 0.6795763 0.6929409 0.6944544 0.7013972 0.7444387    0
## rpart 0.6146852 0.6356794 0.6648653 0.6614290 0.6753588 0.7230283    0
##
## Sens
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Logit 0.8855721 0.9017413 0.9230703 0.9209891 0.9427861 0.9552239    0
## Mars  0.8366337 0.8967662 0.9079602 0.9021033 0.9203980 0.9356436    0
## lda   0.9059406 0.9104478 0.9131570 0.9239717 0.9303483 0.9701493    0
## rpart 0.8514851 0.8793532 0.8955224 0.8966159 0.9092040 0.9504950    0
##
## Spec
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Logit 0.14942529 0.1522989 0.1781609 0.2011494 0.2241379 0.3563218    0
## Mars  0.20689655 0.2183908 0.2758621 0.2770115 0.3103448 0.4137931    0
## lda   0.09195402 0.1752874 0.1954023 0.1896552 0.2183908 0.2643678    0
## rpart 0.14942529 0.2126437 0.2873563 0.2597701 0.2988506 0.3563218    0
```