Bi-cycle Analysis R code

Bhasheyam and Hemanth

18 October 2017

# Read the Data

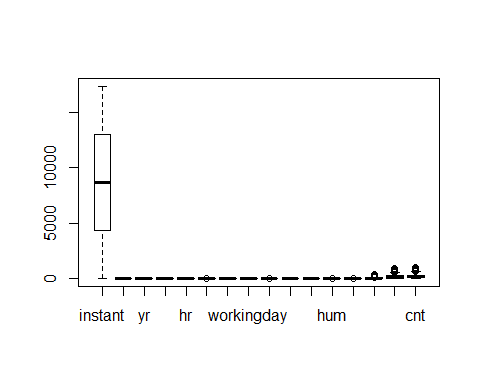
datacy=read.csv("B:/MS/Fall-2017/ML/Project/Bycycle/Data/hour.csv")  
dim(datacy)

## [1] 17379 17

fix(datacy)

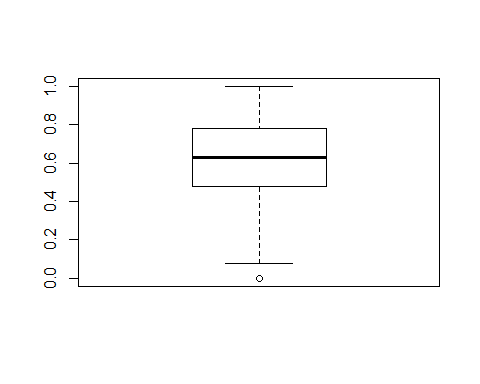
# Outliers are removed from the data

boxplot(Filter(is.numeric,datacy))

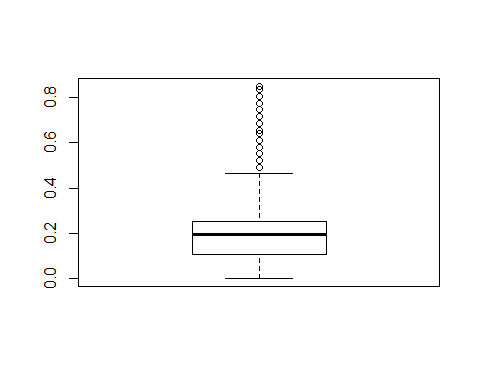


# from the above we are able to see hum, windspeed, has some outliers

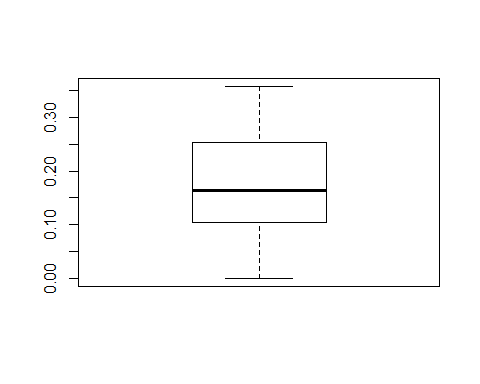
boxplot(datacy$hum)



boxplot(datacy$windspeed)



changed = datacy$windspeed[datacy$windspeed<0.38]  
boxplot(changed)



# After Removing the Outliers:

dim(datacy)

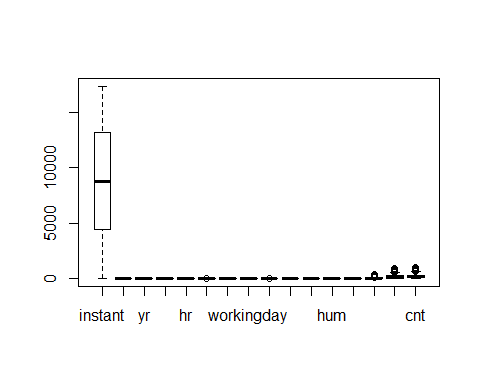
## [1] 17379 17

datacycle = datacy[datacy$windspeed < 0.37 & datacy$hum > 0.20,]  
dim(datacycle)

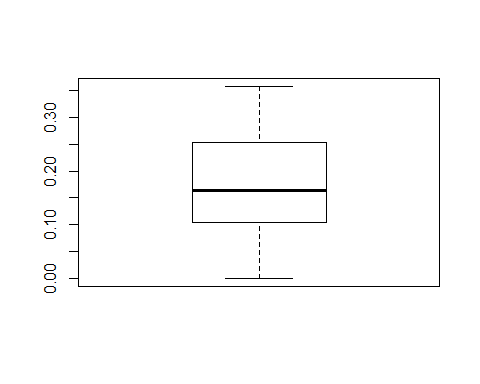
## [1] 15972 17

1392 Instance are removed as they have outliers

boxplot(Filter(is.numeric,datacycle))



boxplot(datacycle$windspeed)



# to find the better model and learning of the data

Lets Introduced new column as Contclass

# analysis 1 - for the Count

contclass = True -> High  
contclass = False -> Low

datacycle$countclass = apply(datacycle, 1, function(x) x[17] > mean(datacycle$cnt ))  
set.seed(123)  
jumble = runif(nrow(datacycle))  
datacycle = datacycle[ordered(jumble),]  
sampleindex = sample(2, nrow(datacycle),replace = TRUE, prob = c(0.80, 0.20))  
Train = datacycle[sampleindex == 1,]  
Test = datacycle[sampleindex == 2,]  
dim(Train)

## [1] 12802 18

dim(Test)

## [1] 3170 18

tdata = Train[3:14]  
tdata = cbind(tdata,Train[18])

library(OneR)

## Warning: package 'OneR' was built under R version 3.4.2

m = optbin(tdata)  
mod = OneR(m)  
summary(mod)

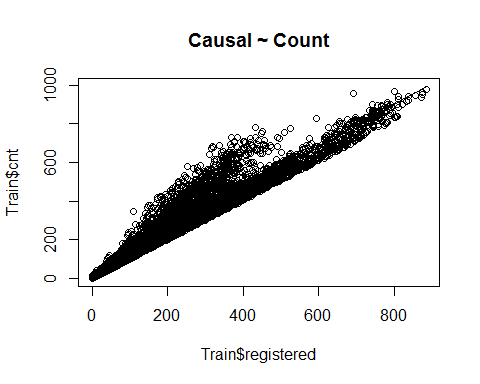
##   
## Call:  
## OneR.data.frame(x = m)  
##   
## Rules:  
## If atemp = (-0.001,0.559] then countclass = FALSE  
## If atemp = (0.559,1] then countclass = TRUE  
##   
## Accuracy:  
## 8589 of 12802 instances classified correctly (67.09%)  
##   
## Contingency table:  
## atemp  
## countclass (-0.001,0.559] (0.559,1] Sum  
## FALSE \* 5762 1919 7681  
## TRUE 2294 \* 2827 5121  
## Sum 8056 4746 12802  
## ---  
## Maximum in each column: '\*'  
##   
## Pearson's Chi-squared test:  
## X-squared = 1201.5, df = 1, p-value < 2.2e-16

predictmod = predict(mod,Test)  
eval\_model(predictmod, Test)

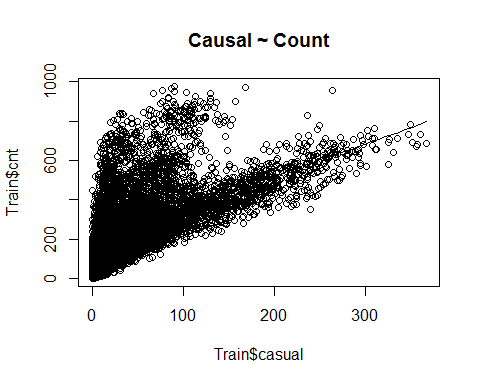
##   
## Confusion matrix (absolute):  
## Actual  
## Prediction FALSE TRUE Sum  
## FALSE 1456 576 2032  
## TRUE 449 689 1138  
## Sum 1905 1265 3170  
##   
## Confusion matrix (relative):  
## Actual  
## Prediction FALSE TRUE Sum  
## FALSE 0.46 0.18 0.64  
## TRUE 0.14 0.22 0.36  
## Sum 0.60 0.40 1.00  
##   
## Accuracy:  
## 0.6767 (2145/3170)  
##   
## Error rate:  
## 0.3233 (1025/3170)  
##   
## Error rate reduction (vs. base rate):  
## 0.1897 (p-value < 2.2e-16)

### LINEAR REGRESSION

scatter.smooth(x=Train$registered, y=Train$cnt, main="Causal ~ Count")



scatter.smooth(x=Train$casual, y=Train$cnt, main="Causal ~ Count")



##Partioning the dataset to registered users & causal users  
x\_data = subset(datacycle, select = -c(cnt, countclass))  
y\_data = subset(datacycle, select = cnt)  
x\_data = subset(x\_data, select = -registered)  
x\_data = subset(x\_data, select = -casual)

## Partioning   
#casual  
ytrain\_casual = Train['casual']  
ytest\_casual = Test['casual']  
  
xtrain\_casual = subset(Train, select = -c(casual, registered, cnt))  
xtest\_casual = subset(Test, select = -c( casual, countclass, cnt))

x\_train = subset(Train, select = -c(registered, cnt, instant, dteday, countclass))  
x\_test = subset(Test, select = -c(registered, cnt, instant, dteday, countclass))

lmMod <- lm(x\_train$casual~. , data=x\_train)  
summary(lmMod)

##   
## Call:  
## lm(formula = x\_train$casual ~ ., data = x\_train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -95.503 -20.447 -3.669 13.467 274.076   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 21.218726 2.128127 9.971 < 2e-16 \*\*\*  
## season 1.253834 0.543757 2.306 0.02113 \*   
## yr 9.552786 0.636656 15.005 < 2e-16 \*\*\*  
## mnth -0.001053 0.170719 -0.006 0.99508   
## hr 1.252732 0.048068 26.061 < 2e-16 \*\*\*  
## holiday -10.080291 1.968485 -5.121 3.09e-07 \*\*\*  
## weekday 0.793014 0.159019 4.987 6.22e-07 \*\*\*  
## workingday -34.683961 0.704845 -49.208 < 2e-16 \*\*\*  
## weathersit 2.661940 0.565574 4.707 2.55e-06 \*\*\*  
## temp 55.194901 11.520253 4.791 1.68e-06 \*\*\*  
## atemp 54.504404 13.020840 4.186 2.86e-05 \*\*\*  
## hum -71.488740 2.059297 -34.715 < 2e-16 \*\*\*  
## windspeed 9.064393 3.450285 2.627 0.00862 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 35.81 on 12789 degrees of freedom  
## Multiple R-squared: 0.4548, Adjusted R-squared: 0.4543   
## F-statistic: 888.9 on 12 and 12789 DF, p-value: < 2.2e-16

casual\_predict = predict(lmMod, x\_test)  
actuals\_preds <- data.frame(cbind(actuals=x\_test$casual, predicteds=casual\_predict))  
cor(actuals\_preds)

## actuals predicteds  
## actuals 1.0000000 0.6836118  
## predicteds 0.6836118 1.0000000

head(actuals\_preds)

## actuals predicteds  
## 13830 108 102.85721  
## 15474 10 33.16796  
## 16664 2 16.13314  
## 12007 258 97.24347  
## 10159 12 41.93040  
## 15749 29 22.77337

## 68.3% test accuracy for causal users

xtrain\_reg = subset(Train, select = -c(casual, cnt, instant, dteday, countclass))  
xtest\_reg = subset(Test, select = -c(casual, cnt, instant, dteday, countclass))

lmMod\_reg <- lm(xtrain\_reg$registered~. , data=xtrain\_reg)  
reg\_predict = predict(lmMod\_reg, xtest\_reg)  
actuals\_preds\_reg <- data.frame(cbind(actuals=xtest\_reg$registered, predicteds=reg\_predict))  
cor(actuals\_preds\_reg)

## actuals predicteds  
## actuals 1.0000000 0.5572793  
## predicteds 0.5572793 1.0000000

## 55.7% test accuracy for registered users

# Transformed Linear Regreession

Train\_cpy = Train  
Test\_cpy = Test

# Convert the cnt to Log

Train\_cpy$log\_cnt = log(Train$cnt)  
Test\_cpy$log\_cnt = log(Test$cnt)  
xtrain\_log = subset(Train\_cpy, select = -c(registered, casual, cnt, instant, dteday, countclass))  
xtest\_log = subset(Test\_cpy, select = -c(casual, registered, cnt, instant, dteday, countclass))

lmMod\_log <- lm(xtrain\_log$log\_cnt~. , data=xtrain\_log)  
log\_predict = predict(lmMod\_log, xtest\_log)  
actuals\_preds\_log <- data.frame(cbind(actuals=xtest\_log$log\_cnt, predicteds=log\_predict))  
cor(actuals\_preds\_log)

## actuals predicteds  
## actuals 1.0000000 0.6891219  
## predicteds 0.6891219 1.0000000

## 68.91% accuracy

# creating train and test task for the classifier analysis

# Drop features

library(mlr)

## Warning: package 'mlr' was built under R version 3.4.2

## Loading required package: ParamHelpers

## Warning: package 'ParamHelpers' was built under R version 3.4.2

traintaskf = makeClassifTask(data = Train ,target = "countclass")

## Warning in makeTask(type = type, data = data, weights = weights, blocking =  
## blocking, : Empty factor levels were dropped for columns: dteday

traintask = makeClassifTask(data = Train ,target = "countclass" , positive = "TRUE")

## Warning in makeTask(type = type, data = data, weights = weights, blocking =  
## blocking, : Empty factor levels were dropped for columns: dteday

traintask = dropFeatures(task = traintaskf, features = c("dteday","instant","cnt","registered","casual"))  
testtaskf = makeClassifTask(data = Test ,target = "countclass")

## Warning in makeTask(type = type, data = data, weights = weights, blocking =  
## blocking, : Empty factor levels were dropped for columns: dteday

testtask = dropFeatures(task = testtaskf, features = c("dteday","instant","cnt","registered","casual"))

# classifir -1 QDA - Quadratic Discriminant Analysis

cycleqda<- makeLearner("classif.qda", predict.type = "response")  
cyclequdatrain = train(cycleqda, task = traintask)  
qdapredict = predict(cyclequdatrain, testtask)  
table(Test$countclass, qdapredict$data$response)

##   
## FALSE TRUE  
## FALSE 1609 296  
## TRUE 302 963

Here the Accuracy is 81 %

# classifier - 2 Logistic Regression

cyclelr = makeLearner("classif.logreg", predict.type = "response")  
cyclequdatrain = train(cyclelr, task = traintask)  
logrpredict = predict(cyclequdatrain, testtask)  
table(Test$countclass, logrpredict$data$response)

##   
## FALSE TRUE  
## FALSE 1603 302  
## TRUE 412 853

Accuracy is 77%

# classifier 3 - Desicion tree

cycletree = makeLearner("classif.rpart", predict.type = "response")  
   
  
treecv = makeResampleDesc("CV",iters = 10L)  
  
  
param = makeParamSet(  
makeIntegerParam("minsplit",lower = 10, upper = 20),  
makeIntegerParam("minbucket", lower = 5, upper = 10),  
makeNumericParam("cp", lower = 0.001, upper = 0.1)  
)  
  
  
control = makeTuneControlGrid()  
  
treetune <- tuneParams(learner = cycletree, resampling = treecv, task = traintask, par.set = param, control = control, measures = acc)

## [Tune] Started tuning learner classif.rpart for parameter set:

## Type len Def Constr Req Tunable Trafo  
## minsplit integer - - 10 to 20 - TRUE -  
## minbucket integer - - 5 to 10 - TRUE -  
## cp numeric - - 0.001 to 0.1 - TRUE -

## With control class: TuneControlGrid

## Imputation value: -0

## [Tune-x] 1: minsplit=10; minbucket=5; cp=0.001

## [Tune-y] 1: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 2: minsplit=11; minbucket=5; cp=0.001

## [Tune-y] 2: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 3: minsplit=12; minbucket=5; cp=0.001

## [Tune-y] 3: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 4: minsplit=13; minbucket=5; cp=0.001

## [Tune-y] 4: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 5: minsplit=14; minbucket=5; cp=0.001

## [Tune-y] 5: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 6: minsplit=16; minbucket=5; cp=0.001

## [Tune-y] 6: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 7: minsplit=17; minbucket=5; cp=0.001

## [Tune-y] 7: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 8: minsplit=18; minbucket=5; cp=0.001

## [Tune-y] 8: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 9: minsplit=19; minbucket=5; cp=0.001

## [Tune-y] 9: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 10: minsplit=20; minbucket=5; cp=0.001

## [Tune-y] 10: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 11: minsplit=10; minbucket=6; cp=0.001

## [Tune-y] 11: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 12: minsplit=11; minbucket=6; cp=0.001

## [Tune-y] 12: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 13: minsplit=12; minbucket=6; cp=0.001

## [Tune-y] 13: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 14: minsplit=13; minbucket=6; cp=0.001

## [Tune-y] 14: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 15: minsplit=14; minbucket=6; cp=0.001

## [Tune-y] 15: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 16: minsplit=16; minbucket=6; cp=0.001

## [Tune-y] 16: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 17: minsplit=17; minbucket=6; cp=0.001

## [Tune-y] 17: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 18: minsplit=18; minbucket=6; cp=0.001

## [Tune-y] 18: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 19: minsplit=19; minbucket=6; cp=0.001

## [Tune-y] 19: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 20: minsplit=20; minbucket=6; cp=0.001

## [Tune-y] 20: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 21: minsplit=10; minbucket=7; cp=0.001

## [Tune-y] 21: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 22: minsplit=11; minbucket=7; cp=0.001

## [Tune-y] 22: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 23: minsplit=12; minbucket=7; cp=0.001

## [Tune-y] 23: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 24: minsplit=13; minbucket=7; cp=0.001

## [Tune-y] 24: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 25: minsplit=14; minbucket=7; cp=0.001

## [Tune-y] 25: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 26: minsplit=16; minbucket=7; cp=0.001

## [Tune-y] 26: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 27: minsplit=17; minbucket=7; cp=0.001

## [Tune-y] 27: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 28: minsplit=18; minbucket=7; cp=0.001

## [Tune-y] 28: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 29: minsplit=19; minbucket=7; cp=0.001

## [Tune-y] 29: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 30: minsplit=20; minbucket=7; cp=0.001

## [Tune-y] 30: acc.test.mean=0.919; time: 0.0 min

## [Tune-x] 31: minsplit=10; minbucket=8; cp=0.001

## [Tune-y] 31: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 32: minsplit=11; minbucket=8; cp=0.001

## [Tune-y] 32: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 33: minsplit=12; minbucket=8; cp=0.001

## [Tune-y] 33: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 34: minsplit=13; minbucket=8; cp=0.001

## [Tune-y] 34: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 35: minsplit=14; minbucket=8; cp=0.001

## [Tune-y] 35: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 36: minsplit=16; minbucket=8; cp=0.001

## [Tune-y] 36: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 37: minsplit=17; minbucket=8; cp=0.001

## [Tune-y] 37: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 38: minsplit=18; minbucket=8; cp=0.001

## [Tune-y] 38: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 39: minsplit=19; minbucket=8; cp=0.001

## [Tune-y] 39: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 40: minsplit=20; minbucket=8; cp=0.001

## [Tune-y] 40: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 41: minsplit=10; minbucket=9; cp=0.001

## [Tune-y] 41: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 42: minsplit=11; minbucket=9; cp=0.001

## [Tune-y] 42: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 43: minsplit=12; minbucket=9; cp=0.001

## [Tune-y] 43: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 44: minsplit=13; minbucket=9; cp=0.001

## [Tune-y] 44: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 45: minsplit=14; minbucket=9; cp=0.001

## [Tune-y] 45: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 46: minsplit=16; minbucket=9; cp=0.001

## [Tune-y] 46: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 47: minsplit=17; minbucket=9; cp=0.001

## [Tune-y] 47: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 48: minsplit=18; minbucket=9; cp=0.001

## [Tune-y] 48: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 49: minsplit=19; minbucket=9; cp=0.001

## [Tune-y] 49: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 50: minsplit=20; minbucket=9; cp=0.001

## [Tune-y] 50: acc.test.mean=0.918; time: 0.0 min

## [Tune-x] 51: minsplit=10; minbucket=10; cp=0.001

## [Tune-y] 51: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 52: minsplit=11; minbucket=10; cp=0.001

## [Tune-y] 52: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 53: minsplit=12; minbucket=10; cp=0.001

## [Tune-y] 53: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 54: minsplit=13; minbucket=10; cp=0.001

## [Tune-y] 54: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 55: minsplit=14; minbucket=10; cp=0.001

## [Tune-y] 55: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 56: minsplit=16; minbucket=10; cp=0.001

## [Tune-y] 56: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 57: minsplit=17; minbucket=10; cp=0.001

## [Tune-y] 57: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 58: minsplit=18; minbucket=10; cp=0.001

## [Tune-y] 58: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 59: minsplit=19; minbucket=10; cp=0.001

## [Tune-y] 59: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 60: minsplit=20; minbucket=10; cp=0.001

## [Tune-y] 60: acc.test.mean=0.917; time: 0.0 min

## [Tune-x] 61: minsplit=10; minbucket=5; cp=0.012

## [Tune-y] 61: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 62: minsplit=11; minbucket=5; cp=0.012

## [Tune-y] 62: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 63: minsplit=12; minbucket=5; cp=0.012

## [Tune-y] 63: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 64: minsplit=13; minbucket=5; cp=0.012

## [Tune-y] 64: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 65: minsplit=14; minbucket=5; cp=0.012

## [Tune-y] 65: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 66: minsplit=16; minbucket=5; cp=0.012

## [Tune-y] 66: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 67: minsplit=17; minbucket=5; cp=0.012

## [Tune-y] 67: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 68: minsplit=18; minbucket=5; cp=0.012

## [Tune-y] 68: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 69: minsplit=19; minbucket=5; cp=0.012

## [Tune-y] 69: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 70: minsplit=20; minbucket=5; cp=0.012

## [Tune-y] 70: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 71: minsplit=10; minbucket=6; cp=0.012

## [Tune-y] 71: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 72: minsplit=11; minbucket=6; cp=0.012

## [Tune-y] 72: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 73: minsplit=12; minbucket=6; cp=0.012

## [Tune-y] 73: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 74: minsplit=13; minbucket=6; cp=0.012

## [Tune-y] 74: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 75: minsplit=14; minbucket=6; cp=0.012

## [Tune-y] 75: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 76: minsplit=16; minbucket=6; cp=0.012

## [Tune-y] 76: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 77: minsplit=17; minbucket=6; cp=0.012

## [Tune-y] 77: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 78: minsplit=18; minbucket=6; cp=0.012

## [Tune-y] 78: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 79: minsplit=19; minbucket=6; cp=0.012

## [Tune-y] 79: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 80: minsplit=20; minbucket=6; cp=0.012

## [Tune-y] 80: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 81: minsplit=10; minbucket=7; cp=0.012

## [Tune-y] 81: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 82: minsplit=11; minbucket=7; cp=0.012

## [Tune-y] 82: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 83: minsplit=12; minbucket=7; cp=0.012

## [Tune-y] 83: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 84: minsplit=13; minbucket=7; cp=0.012

## [Tune-y] 84: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 85: minsplit=14; minbucket=7; cp=0.012

## [Tune-y] 85: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 86: minsplit=16; minbucket=7; cp=0.012

## [Tune-y] 86: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 87: minsplit=17; minbucket=7; cp=0.012

## [Tune-y] 87: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 88: minsplit=18; minbucket=7; cp=0.012

## [Tune-y] 88: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 89: minsplit=19; minbucket=7; cp=0.012

## [Tune-y] 89: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 90: minsplit=20; minbucket=7; cp=0.012

## [Tune-y] 90: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 91: minsplit=10; minbucket=8; cp=0.012

## [Tune-y] 91: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 92: minsplit=11; minbucket=8; cp=0.012

## [Tune-y] 92: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 93: minsplit=12; minbucket=8; cp=0.012

## [Tune-y] 93: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 94: minsplit=13; minbucket=8; cp=0.012

## [Tune-y] 94: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 95: minsplit=14; minbucket=8; cp=0.012

## [Tune-y] 95: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 96: minsplit=16; minbucket=8; cp=0.012

## [Tune-y] 96: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 97: minsplit=17; minbucket=8; cp=0.012

## [Tune-y] 97: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 98: minsplit=18; minbucket=8; cp=0.012

## [Tune-y] 98: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 99: minsplit=19; minbucket=8; cp=0.012

## [Tune-y] 99: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 100: minsplit=20; minbucket=8; cp=0.012

## [Tune-y] 100: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 101: minsplit=10; minbucket=9; cp=0.012

## [Tune-y] 101: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 102: minsplit=11; minbucket=9; cp=0.012

## [Tune-y] 102: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 103: minsplit=12; minbucket=9; cp=0.012

## [Tune-y] 103: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 104: minsplit=13; minbucket=9; cp=0.012

## [Tune-y] 104: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 105: minsplit=14; minbucket=9; cp=0.012

## [Tune-y] 105: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 106: minsplit=16; minbucket=9; cp=0.012

## [Tune-y] 106: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 107: minsplit=17; minbucket=9; cp=0.012

## [Tune-y] 107: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 108: minsplit=18; minbucket=9; cp=0.012

## [Tune-y] 108: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 109: minsplit=19; minbucket=9; cp=0.012

## [Tune-y] 109: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 110: minsplit=20; minbucket=9; cp=0.012

## [Tune-y] 110: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 111: minsplit=10; minbucket=10; cp=0.012

## [Tune-y] 111: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 112: minsplit=11; minbucket=10; cp=0.012

## [Tune-y] 112: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 113: minsplit=12; minbucket=10; cp=0.012

## [Tune-y] 113: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 114: minsplit=13; minbucket=10; cp=0.012

## [Tune-y] 114: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 115: minsplit=14; minbucket=10; cp=0.012

## [Tune-y] 115: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 116: minsplit=16; minbucket=10; cp=0.012

## [Tune-y] 116: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 117: minsplit=17; minbucket=10; cp=0.012

## [Tune-y] 117: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 118: minsplit=18; minbucket=10; cp=0.012

## [Tune-y] 118: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 119: minsplit=19; minbucket=10; cp=0.012

## [Tune-y] 119: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 120: minsplit=20; minbucket=10; cp=0.012

## [Tune-y] 120: acc.test.mean=0.863; time: 0.0 min

## [Tune-x] 121: minsplit=10; minbucket=5; cp=0.023

## [Tune-y] 121: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 122: minsplit=11; minbucket=5; cp=0.023

## [Tune-y] 122: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 123: minsplit=12; minbucket=5; cp=0.023

## [Tune-y] 123: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 124: minsplit=13; minbucket=5; cp=0.023

## [Tune-y] 124: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 125: minsplit=14; minbucket=5; cp=0.023

## [Tune-y] 125: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 126: minsplit=16; minbucket=5; cp=0.023

## [Tune-y] 126: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 127: minsplit=17; minbucket=5; cp=0.023

## [Tune-y] 127: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 128: minsplit=18; minbucket=5; cp=0.023

## [Tune-y] 128: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 129: minsplit=19; minbucket=5; cp=0.023

## [Tune-y] 129: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 130: minsplit=20; minbucket=5; cp=0.023

## [Tune-y] 130: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 131: minsplit=10; minbucket=6; cp=0.023

## [Tune-y] 131: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 132: minsplit=11; minbucket=6; cp=0.023

## [Tune-y] 132: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 133: minsplit=12; minbucket=6; cp=0.023

## [Tune-y] 133: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 134: minsplit=13; minbucket=6; cp=0.023

## [Tune-y] 134: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 135: minsplit=14; minbucket=6; cp=0.023

## [Tune-y] 135: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 136: minsplit=16; minbucket=6; cp=0.023

## [Tune-y] 136: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 137: minsplit=17; minbucket=6; cp=0.023

## [Tune-y] 137: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 138: minsplit=18; minbucket=6; cp=0.023

## [Tune-y] 138: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 139: minsplit=19; minbucket=6; cp=0.023

## [Tune-y] 139: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 140: minsplit=20; minbucket=6; cp=0.023

## [Tune-y] 140: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 141: minsplit=10; minbucket=7; cp=0.023

## [Tune-y] 141: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 142: minsplit=11; minbucket=7; cp=0.023

## [Tune-y] 142: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 143: minsplit=12; minbucket=7; cp=0.023

## [Tune-y] 143: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 144: minsplit=13; minbucket=7; cp=0.023

## [Tune-y] 144: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 145: minsplit=14; minbucket=7; cp=0.023

## [Tune-y] 145: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 146: minsplit=16; minbucket=7; cp=0.023

## [Tune-y] 146: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 147: minsplit=17; minbucket=7; cp=0.023

## [Tune-y] 147: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 148: minsplit=18; minbucket=7; cp=0.023

## [Tune-y] 148: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 149: minsplit=19; minbucket=7; cp=0.023

## [Tune-y] 149: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 150: minsplit=20; minbucket=7; cp=0.023

## [Tune-y] 150: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 151: minsplit=10; minbucket=8; cp=0.023

## [Tune-y] 151: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 152: minsplit=11; minbucket=8; cp=0.023

## [Tune-y] 152: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 153: minsplit=12; minbucket=8; cp=0.023

## [Tune-y] 153: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 154: minsplit=13; minbucket=8; cp=0.023

## [Tune-y] 154: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 155: minsplit=14; minbucket=8; cp=0.023

## [Tune-y] 155: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 156: minsplit=16; minbucket=8; cp=0.023

## [Tune-y] 156: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 157: minsplit=17; minbucket=8; cp=0.023

## [Tune-y] 157: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 158: minsplit=18; minbucket=8; cp=0.023

## [Tune-y] 158: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 159: minsplit=19; minbucket=8; cp=0.023

## [Tune-y] 159: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 160: minsplit=20; minbucket=8; cp=0.023

## [Tune-y] 160: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 161: minsplit=10; minbucket=9; cp=0.023

## [Tune-y] 161: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 162: minsplit=11; minbucket=9; cp=0.023

## [Tune-y] 162: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 163: minsplit=12; minbucket=9; cp=0.023

## [Tune-y] 163: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 164: minsplit=13; minbucket=9; cp=0.023

## [Tune-y] 164: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 165: minsplit=14; minbucket=9; cp=0.023

## [Tune-y] 165: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 166: minsplit=16; minbucket=9; cp=0.023

## [Tune-y] 166: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 167: minsplit=17; minbucket=9; cp=0.023

## [Tune-y] 167: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 168: minsplit=18; minbucket=9; cp=0.023

## [Tune-y] 168: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 169: minsplit=19; minbucket=9; cp=0.023

## [Tune-y] 169: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 170: minsplit=20; minbucket=9; cp=0.023

## [Tune-y] 170: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 171: minsplit=10; minbucket=10; cp=0.023

## [Tune-y] 171: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 172: minsplit=11; minbucket=10; cp=0.023

## [Tune-y] 172: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 173: minsplit=12; minbucket=10; cp=0.023

## [Tune-y] 173: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 174: minsplit=13; minbucket=10; cp=0.023

## [Tune-y] 174: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 175: minsplit=14; minbucket=10; cp=0.023

## [Tune-y] 175: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 176: minsplit=16; minbucket=10; cp=0.023

## [Tune-y] 176: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 177: minsplit=17; minbucket=10; cp=0.023

## [Tune-y] 177: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 178: minsplit=18; minbucket=10; cp=0.023

## [Tune-y] 178: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 179: minsplit=19; minbucket=10; cp=0.023

## [Tune-y] 179: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 180: minsplit=20; minbucket=10; cp=0.023

## [Tune-y] 180: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 181: minsplit=10; minbucket=5; cp=0.034

## [Tune-y] 181: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 182: minsplit=11; minbucket=5; cp=0.034

## [Tune-y] 182: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 183: minsplit=12; minbucket=5; cp=0.034

## [Tune-y] 183: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 184: minsplit=13; minbucket=5; cp=0.034

## [Tune-y] 184: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 185: minsplit=14; minbucket=5; cp=0.034

## [Tune-y] 185: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 186: minsplit=16; minbucket=5; cp=0.034

## [Tune-y] 186: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 187: minsplit=17; minbucket=5; cp=0.034

## [Tune-y] 187: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 188: minsplit=18; minbucket=5; cp=0.034

## [Tune-y] 188: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 189: minsplit=19; minbucket=5; cp=0.034

## [Tune-y] 189: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 190: minsplit=20; minbucket=5; cp=0.034

## [Tune-y] 190: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 191: minsplit=10; minbucket=6; cp=0.034

## [Tune-y] 191: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 192: minsplit=11; minbucket=6; cp=0.034

## [Tune-y] 192: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 193: minsplit=12; minbucket=6; cp=0.034

## [Tune-y] 193: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 194: minsplit=13; minbucket=6; cp=0.034

## [Tune-y] 194: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 195: minsplit=14; minbucket=6; cp=0.034

## [Tune-y] 195: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 196: minsplit=16; minbucket=6; cp=0.034

## [Tune-y] 196: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 197: minsplit=17; minbucket=6; cp=0.034

## [Tune-y] 197: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 198: minsplit=18; minbucket=6; cp=0.034

## [Tune-y] 198: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 199: minsplit=19; minbucket=6; cp=0.034

## [Tune-y] 199: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 200: minsplit=20; minbucket=6; cp=0.034

## [Tune-y] 200: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 201: minsplit=10; minbucket=7; cp=0.034

## [Tune-y] 201: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 202: minsplit=11; minbucket=7; cp=0.034

## [Tune-y] 202: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 203: minsplit=12; minbucket=7; cp=0.034

## [Tune-y] 203: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 204: minsplit=13; minbucket=7; cp=0.034

## [Tune-y] 204: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 205: minsplit=14; minbucket=7; cp=0.034

## [Tune-y] 205: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 206: minsplit=16; minbucket=7; cp=0.034

## [Tune-y] 206: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 207: minsplit=17; minbucket=7; cp=0.034

## [Tune-y] 207: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 208: minsplit=18; minbucket=7; cp=0.034

## [Tune-y] 208: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 209: minsplit=19; minbucket=7; cp=0.034

## [Tune-y] 209: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 210: minsplit=20; minbucket=7; cp=0.034

## [Tune-y] 210: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 211: minsplit=10; minbucket=8; cp=0.034

## [Tune-y] 211: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 212: minsplit=11; minbucket=8; cp=0.034

## [Tune-y] 212: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 213: minsplit=12; minbucket=8; cp=0.034

## [Tune-y] 213: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 214: minsplit=13; minbucket=8; cp=0.034

## [Tune-y] 214: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 215: minsplit=14; minbucket=8; cp=0.034

## [Tune-y] 215: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 216: minsplit=16; minbucket=8; cp=0.034

## [Tune-y] 216: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 217: minsplit=17; minbucket=8; cp=0.034

## [Tune-y] 217: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 218: minsplit=18; minbucket=8; cp=0.034

## [Tune-y] 218: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 219: minsplit=19; minbucket=8; cp=0.034

## [Tune-y] 219: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 220: minsplit=20; minbucket=8; cp=0.034

## [Tune-y] 220: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 221: minsplit=10; minbucket=9; cp=0.034

## [Tune-y] 221: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 222: minsplit=11; minbucket=9; cp=0.034

## [Tune-y] 222: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 223: minsplit=12; minbucket=9; cp=0.034

## [Tune-y] 223: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 224: minsplit=13; minbucket=9; cp=0.034

## [Tune-y] 224: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 225: minsplit=14; minbucket=9; cp=0.034

## [Tune-y] 225: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 226: minsplit=16; minbucket=9; cp=0.034

## [Tune-y] 226: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 227: minsplit=17; minbucket=9; cp=0.034

## [Tune-y] 227: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 228: minsplit=18; minbucket=9; cp=0.034

## [Tune-y] 228: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 229: minsplit=19; minbucket=9; cp=0.034

## [Tune-y] 229: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 230: minsplit=20; minbucket=9; cp=0.034

## [Tune-y] 230: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 231: minsplit=10; minbucket=10; cp=0.034

## [Tune-y] 231: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 232: minsplit=11; minbucket=10; cp=0.034

## [Tune-y] 232: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 233: minsplit=12; minbucket=10; cp=0.034

## [Tune-y] 233: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 234: minsplit=13; minbucket=10; cp=0.034

## [Tune-y] 234: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 235: minsplit=14; minbucket=10; cp=0.034

## [Tune-y] 235: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 236: minsplit=16; minbucket=10; cp=0.034

## [Tune-y] 236: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 237: minsplit=17; minbucket=10; cp=0.034

## [Tune-y] 237: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 238: minsplit=18; minbucket=10; cp=0.034

## [Tune-y] 238: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 239: minsplit=19; minbucket=10; cp=0.034

## [Tune-y] 239: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 240: minsplit=20; minbucket=10; cp=0.034

## [Tune-y] 240: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 241: minsplit=10; minbucket=5; cp=0.045

## [Tune-y] 241: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 242: minsplit=11; minbucket=5; cp=0.045

## [Tune-y] 242: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 243: minsplit=12; minbucket=5; cp=0.045

## [Tune-y] 243: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 244: minsplit=13; minbucket=5; cp=0.045

## [Tune-y] 244: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 245: minsplit=14; minbucket=5; cp=0.045

## [Tune-y] 245: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 246: minsplit=16; minbucket=5; cp=0.045

## [Tune-y] 246: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 247: minsplit=17; minbucket=5; cp=0.045

## [Tune-y] 247: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 248: minsplit=18; minbucket=5; cp=0.045

## [Tune-y] 248: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 249: minsplit=19; minbucket=5; cp=0.045

## [Tune-y] 249: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 250: minsplit=20; minbucket=5; cp=0.045

## [Tune-y] 250: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 251: minsplit=10; minbucket=6; cp=0.045

## [Tune-y] 251: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 252: minsplit=11; minbucket=6; cp=0.045

## [Tune-y] 252: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 253: minsplit=12; minbucket=6; cp=0.045

## [Tune-y] 253: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 254: minsplit=13; minbucket=6; cp=0.045

## [Tune-y] 254: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 255: minsplit=14; minbucket=6; cp=0.045

## [Tune-y] 255: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 256: minsplit=16; minbucket=6; cp=0.045

## [Tune-y] 256: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 257: minsplit=17; minbucket=6; cp=0.045

## [Tune-y] 257: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 258: minsplit=18; minbucket=6; cp=0.045

## [Tune-y] 258: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 259: minsplit=19; minbucket=6; cp=0.045

## [Tune-y] 259: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 260: minsplit=20; minbucket=6; cp=0.045

## [Tune-y] 260: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 261: minsplit=10; minbucket=7; cp=0.045

## [Tune-y] 261: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 262: minsplit=11; minbucket=7; cp=0.045

## [Tune-y] 262: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 263: minsplit=12; minbucket=7; cp=0.045

## [Tune-y] 263: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 264: minsplit=13; minbucket=7; cp=0.045

## [Tune-y] 264: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 265: minsplit=14; minbucket=7; cp=0.045

## [Tune-y] 265: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 266: minsplit=16; minbucket=7; cp=0.045

## [Tune-y] 266: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 267: minsplit=17; minbucket=7; cp=0.045

## [Tune-y] 267: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 268: minsplit=18; minbucket=7; cp=0.045

## [Tune-y] 268: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 269: minsplit=19; minbucket=7; cp=0.045

## [Tune-y] 269: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 270: minsplit=20; minbucket=7; cp=0.045

## [Tune-y] 270: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 271: minsplit=10; minbucket=8; cp=0.045

## [Tune-y] 271: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 272: minsplit=11; minbucket=8; cp=0.045

## [Tune-y] 272: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 273: minsplit=12; minbucket=8; cp=0.045

## [Tune-y] 273: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 274: minsplit=13; minbucket=8; cp=0.045

## [Tune-y] 274: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 275: minsplit=14; minbucket=8; cp=0.045

## [Tune-y] 275: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 276: minsplit=16; minbucket=8; cp=0.045

## [Tune-y] 276: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 277: minsplit=17; minbucket=8; cp=0.045

## [Tune-y] 277: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 278: minsplit=18; minbucket=8; cp=0.045

## [Tune-y] 278: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 279: minsplit=19; minbucket=8; cp=0.045

## [Tune-y] 279: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 280: minsplit=20; minbucket=8; cp=0.045

## [Tune-y] 280: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 281: minsplit=10; minbucket=9; cp=0.045

## [Tune-y] 281: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 282: minsplit=11; minbucket=9; cp=0.045

## [Tune-y] 282: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 283: minsplit=12; minbucket=9; cp=0.045

## [Tune-y] 283: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 284: minsplit=13; minbucket=9; cp=0.045

## [Tune-y] 284: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 285: minsplit=14; minbucket=9; cp=0.045

## [Tune-y] 285: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 286: minsplit=16; minbucket=9; cp=0.045

## [Tune-y] 286: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 287: minsplit=17; minbucket=9; cp=0.045

## [Tune-y] 287: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 288: minsplit=18; minbucket=9; cp=0.045

## [Tune-y] 288: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 289: minsplit=19; minbucket=9; cp=0.045

## [Tune-y] 289: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 290: minsplit=20; minbucket=9; cp=0.045

## [Tune-y] 290: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 291: minsplit=10; minbucket=10; cp=0.045

## [Tune-y] 291: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 292: minsplit=11; minbucket=10; cp=0.045

## [Tune-y] 292: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 293: minsplit=12; minbucket=10; cp=0.045

## [Tune-y] 293: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 294: minsplit=13; minbucket=10; cp=0.045

## [Tune-y] 294: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 295: minsplit=14; minbucket=10; cp=0.045

## [Tune-y] 295: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 296: minsplit=16; minbucket=10; cp=0.045

## [Tune-y] 296: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 297: minsplit=17; minbucket=10; cp=0.045

## [Tune-y] 297: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 298: minsplit=18; minbucket=10; cp=0.045

## [Tune-y] 298: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 299: minsplit=19; minbucket=10; cp=0.045

## [Tune-y] 299: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 300: minsplit=20; minbucket=10; cp=0.045

## [Tune-y] 300: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 301: minsplit=10; minbucket=5; cp=0.056

## [Tune-y] 301: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 302: minsplit=11; minbucket=5; cp=0.056

## [Tune-y] 302: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 303: minsplit=12; minbucket=5; cp=0.056

## [Tune-y] 303: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 304: minsplit=13; minbucket=5; cp=0.056

## [Tune-y] 304: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 305: minsplit=14; minbucket=5; cp=0.056

## [Tune-y] 305: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 306: minsplit=16; minbucket=5; cp=0.056

## [Tune-y] 306: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 307: minsplit=17; minbucket=5; cp=0.056

## [Tune-y] 307: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 308: minsplit=18; minbucket=5; cp=0.056

## [Tune-y] 308: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 309: minsplit=19; minbucket=5; cp=0.056

## [Tune-y] 309: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 310: minsplit=20; minbucket=5; cp=0.056

## [Tune-y] 310: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 311: minsplit=10; minbucket=6; cp=0.056

## [Tune-y] 311: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 312: minsplit=11; minbucket=6; cp=0.056

## [Tune-y] 312: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 313: minsplit=12; minbucket=6; cp=0.056

## [Tune-y] 313: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 314: minsplit=13; minbucket=6; cp=0.056

## [Tune-y] 314: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 315: minsplit=14; minbucket=6; cp=0.056

## [Tune-y] 315: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 316: minsplit=16; minbucket=6; cp=0.056

## [Tune-y] 316: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 317: minsplit=17; minbucket=6; cp=0.056

## [Tune-y] 317: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 318: minsplit=18; minbucket=6; cp=0.056

## [Tune-y] 318: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 319: minsplit=19; minbucket=6; cp=0.056

## [Tune-y] 319: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 320: minsplit=20; minbucket=6; cp=0.056

## [Tune-y] 320: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 321: minsplit=10; minbucket=7; cp=0.056

## [Tune-y] 321: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 322: minsplit=11; minbucket=7; cp=0.056

## [Tune-y] 322: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 323: minsplit=12; minbucket=7; cp=0.056

## [Tune-y] 323: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 324: minsplit=13; minbucket=7; cp=0.056

## [Tune-y] 324: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 325: minsplit=14; minbucket=7; cp=0.056

## [Tune-y] 325: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 326: minsplit=16; minbucket=7; cp=0.056

## [Tune-y] 326: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 327: minsplit=17; minbucket=7; cp=0.056

## [Tune-y] 327: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 328: minsplit=18; minbucket=7; cp=0.056

## [Tune-y] 328: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 329: minsplit=19; minbucket=7; cp=0.056

## [Tune-y] 329: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 330: minsplit=20; minbucket=7; cp=0.056

## [Tune-y] 330: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 331: minsplit=10; minbucket=8; cp=0.056

## [Tune-y] 331: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 332: minsplit=11; minbucket=8; cp=0.056

## [Tune-y] 332: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 333: minsplit=12; minbucket=8; cp=0.056

## [Tune-y] 333: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 334: minsplit=13; minbucket=8; cp=0.056

## [Tune-y] 334: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 335: minsplit=14; minbucket=8; cp=0.056

## [Tune-y] 335: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 336: minsplit=16; minbucket=8; cp=0.056

## [Tune-y] 336: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 337: minsplit=17; minbucket=8; cp=0.056

## [Tune-y] 337: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 338: minsplit=18; minbucket=8; cp=0.056

## [Tune-y] 338: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 339: minsplit=19; minbucket=8; cp=0.056

## [Tune-y] 339: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 340: minsplit=20; minbucket=8; cp=0.056

## [Tune-y] 340: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 341: minsplit=10; minbucket=9; cp=0.056

## [Tune-y] 341: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 342: minsplit=11; minbucket=9; cp=0.056

## [Tune-y] 342: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 343: minsplit=12; minbucket=9; cp=0.056

## [Tune-y] 343: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 344: minsplit=13; minbucket=9; cp=0.056

## [Tune-y] 344: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 345: minsplit=14; minbucket=9; cp=0.056

## [Tune-y] 345: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 346: minsplit=16; minbucket=9; cp=0.056

## [Tune-y] 346: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 347: minsplit=17; minbucket=9; cp=0.056

## [Tune-y] 347: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 348: minsplit=18; minbucket=9; cp=0.056

## [Tune-y] 348: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 349: minsplit=19; minbucket=9; cp=0.056

## [Tune-y] 349: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 350: minsplit=20; minbucket=9; cp=0.056

## [Tune-y] 350: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 351: minsplit=10; minbucket=10; cp=0.056

## [Tune-y] 351: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 352: minsplit=11; minbucket=10; cp=0.056

## [Tune-y] 352: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 353: minsplit=12; minbucket=10; cp=0.056

## [Tune-y] 353: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 354: minsplit=13; minbucket=10; cp=0.056

## [Tune-y] 354: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 355: minsplit=14; minbucket=10; cp=0.056

## [Tune-y] 355: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 356: minsplit=16; minbucket=10; cp=0.056

## [Tune-y] 356: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 357: minsplit=17; minbucket=10; cp=0.056

## [Tune-y] 357: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 358: minsplit=18; minbucket=10; cp=0.056

## [Tune-y] 358: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 359: minsplit=19; minbucket=10; cp=0.056

## [Tune-y] 359: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 360: minsplit=20; minbucket=10; cp=0.056

## [Tune-y] 360: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 361: minsplit=10; minbucket=5; cp=0.067

## [Tune-y] 361: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 362: minsplit=11; minbucket=5; cp=0.067

## [Tune-y] 362: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 363: minsplit=12; minbucket=5; cp=0.067

## [Tune-y] 363: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 364: minsplit=13; minbucket=5; cp=0.067

## [Tune-y] 364: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 365: minsplit=14; minbucket=5; cp=0.067

## [Tune-y] 365: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 366: minsplit=16; minbucket=5; cp=0.067

## [Tune-y] 366: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 367: minsplit=17; minbucket=5; cp=0.067

## [Tune-y] 367: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 368: minsplit=18; minbucket=5; cp=0.067

## [Tune-y] 368: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 369: minsplit=19; minbucket=5; cp=0.067

## [Tune-y] 369: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 370: minsplit=20; minbucket=5; cp=0.067

## [Tune-y] 370: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 371: minsplit=10; minbucket=6; cp=0.067

## [Tune-y] 371: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 372: minsplit=11; minbucket=6; cp=0.067

## [Tune-y] 372: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 373: minsplit=12; minbucket=6; cp=0.067

## [Tune-y] 373: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 374: minsplit=13; minbucket=6; cp=0.067

## [Tune-y] 374: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 375: minsplit=14; minbucket=6; cp=0.067

## [Tune-y] 375: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 376: minsplit=16; minbucket=6; cp=0.067

## [Tune-y] 376: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 377: minsplit=17; minbucket=6; cp=0.067

## [Tune-y] 377: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 378: minsplit=18; minbucket=6; cp=0.067

## [Tune-y] 378: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 379: minsplit=19; minbucket=6; cp=0.067

## [Tune-y] 379: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 380: minsplit=20; minbucket=6; cp=0.067

## [Tune-y] 380: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 381: minsplit=10; minbucket=7; cp=0.067

## [Tune-y] 381: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 382: minsplit=11; minbucket=7; cp=0.067

## [Tune-y] 382: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 383: minsplit=12; minbucket=7; cp=0.067

## [Tune-y] 383: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 384: minsplit=13; minbucket=7; cp=0.067

## [Tune-y] 384: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 385: minsplit=14; minbucket=7; cp=0.067

## [Tune-y] 385: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 386: minsplit=16; minbucket=7; cp=0.067

## [Tune-y] 386: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 387: minsplit=17; minbucket=7; cp=0.067

## [Tune-y] 387: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 388: minsplit=18; minbucket=7; cp=0.067

## [Tune-y] 388: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 389: minsplit=19; minbucket=7; cp=0.067

## [Tune-y] 389: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 390: minsplit=20; minbucket=7; cp=0.067

## [Tune-y] 390: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 391: minsplit=10; minbucket=8; cp=0.067

## [Tune-y] 391: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 392: minsplit=11; minbucket=8; cp=0.067

## [Tune-y] 392: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 393: minsplit=12; minbucket=8; cp=0.067

## [Tune-y] 393: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 394: minsplit=13; minbucket=8; cp=0.067

## [Tune-y] 394: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 395: minsplit=14; minbucket=8; cp=0.067

## [Tune-y] 395: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 396: minsplit=16; minbucket=8; cp=0.067

## [Tune-y] 396: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 397: minsplit=17; minbucket=8; cp=0.067

## [Tune-y] 397: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 398: minsplit=18; minbucket=8; cp=0.067

## [Tune-y] 398: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 399: minsplit=19; minbucket=8; cp=0.067

## [Tune-y] 399: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 400: minsplit=20; minbucket=8; cp=0.067

## [Tune-y] 400: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 401: minsplit=10; minbucket=9; cp=0.067

## [Tune-y] 401: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 402: minsplit=11; minbucket=9; cp=0.067

## [Tune-y] 402: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 403: minsplit=12; minbucket=9; cp=0.067

## [Tune-y] 403: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 404: minsplit=13; minbucket=9; cp=0.067

## [Tune-y] 404: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 405: minsplit=14; minbucket=9; cp=0.067

## [Tune-y] 405: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 406: minsplit=16; minbucket=9; cp=0.067

## [Tune-y] 406: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 407: minsplit=17; minbucket=9; cp=0.067

## [Tune-y] 407: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 408: minsplit=18; minbucket=9; cp=0.067

## [Tune-y] 408: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 409: minsplit=19; minbucket=9; cp=0.067

## [Tune-y] 409: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 410: minsplit=20; minbucket=9; cp=0.067

## [Tune-y] 410: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 411: minsplit=10; minbucket=10; cp=0.067

## [Tune-y] 411: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 412: minsplit=11; minbucket=10; cp=0.067

## [Tune-y] 412: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 413: minsplit=12; minbucket=10; cp=0.067

## [Tune-y] 413: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 414: minsplit=13; minbucket=10; cp=0.067

## [Tune-y] 414: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 415: minsplit=14; minbucket=10; cp=0.067

## [Tune-y] 415: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 416: minsplit=16; minbucket=10; cp=0.067

## [Tune-y] 416: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 417: minsplit=17; minbucket=10; cp=0.067

## [Tune-y] 417: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 418: minsplit=18; minbucket=10; cp=0.067

## [Tune-y] 418: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 419: minsplit=19; minbucket=10; cp=0.067

## [Tune-y] 419: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 420: minsplit=20; minbucket=10; cp=0.067

## [Tune-y] 420: acc.test.mean=0.818; time: 0.0 min

## [Tune-x] 421: minsplit=10; minbucket=5; cp=0.078

## [Tune-y] 421: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 422: minsplit=11; minbucket=5; cp=0.078

## [Tune-y] 422: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 423: minsplit=12; minbucket=5; cp=0.078

## [Tune-y] 423: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 424: minsplit=13; minbucket=5; cp=0.078

## [Tune-y] 424: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 425: minsplit=14; minbucket=5; cp=0.078

## [Tune-y] 425: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 426: minsplit=16; minbucket=5; cp=0.078

## [Tune-y] 426: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 427: minsplit=17; minbucket=5; cp=0.078

## [Tune-y] 427: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 428: minsplit=18; minbucket=5; cp=0.078

## [Tune-y] 428: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 429: minsplit=19; minbucket=5; cp=0.078

## [Tune-y] 429: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 430: minsplit=20; minbucket=5; cp=0.078

## [Tune-y] 430: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 431: minsplit=10; minbucket=6; cp=0.078

## [Tune-y] 431: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 432: minsplit=11; minbucket=6; cp=0.078

## [Tune-y] 432: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 433: minsplit=12; minbucket=6; cp=0.078

## [Tune-y] 433: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 434: minsplit=13; minbucket=6; cp=0.078

## [Tune-y] 434: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 435: minsplit=14; minbucket=6; cp=0.078

## [Tune-y] 435: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 436: minsplit=16; minbucket=6; cp=0.078

## [Tune-y] 436: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 437: minsplit=17; minbucket=6; cp=0.078

## [Tune-y] 437: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 438: minsplit=18; minbucket=6; cp=0.078

## [Tune-y] 438: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 439: minsplit=19; minbucket=6; cp=0.078

## [Tune-y] 439: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 440: minsplit=20; minbucket=6; cp=0.078

## [Tune-y] 440: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 441: minsplit=10; minbucket=7; cp=0.078

## [Tune-y] 441: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 442: minsplit=11; minbucket=7; cp=0.078

## [Tune-y] 442: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 443: minsplit=12; minbucket=7; cp=0.078

## [Tune-y] 443: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 444: minsplit=13; minbucket=7; cp=0.078

## [Tune-y] 444: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 445: minsplit=14; minbucket=7; cp=0.078

## [Tune-y] 445: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 446: minsplit=16; minbucket=7; cp=0.078

## [Tune-y] 446: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 447: minsplit=17; minbucket=7; cp=0.078

## [Tune-y] 447: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 448: minsplit=18; minbucket=7; cp=0.078

## [Tune-y] 448: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 449: minsplit=19; minbucket=7; cp=0.078

## [Tune-y] 449: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 450: minsplit=20; minbucket=7; cp=0.078

## [Tune-y] 450: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 451: minsplit=10; minbucket=8; cp=0.078

## [Tune-y] 451: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 452: minsplit=11; minbucket=8; cp=0.078

## [Tune-y] 452: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 453: minsplit=12; minbucket=8; cp=0.078

## [Tune-y] 453: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 454: minsplit=13; minbucket=8; cp=0.078

## [Tune-y] 454: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 455: minsplit=14; minbucket=8; cp=0.078

## [Tune-y] 455: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 456: minsplit=16; minbucket=8; cp=0.078

## [Tune-y] 456: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 457: minsplit=17; minbucket=8; cp=0.078

## [Tune-y] 457: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 458: minsplit=18; minbucket=8; cp=0.078

## [Tune-y] 458: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 459: minsplit=19; minbucket=8; cp=0.078

## [Tune-y] 459: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 460: minsplit=20; minbucket=8; cp=0.078

## [Tune-y] 460: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 461: minsplit=10; minbucket=9; cp=0.078

## [Tune-y] 461: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 462: minsplit=11; minbucket=9; cp=0.078

## [Tune-y] 462: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 463: minsplit=12; minbucket=9; cp=0.078

## [Tune-y] 463: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 464: minsplit=13; minbucket=9; cp=0.078

## [Tune-y] 464: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 465: minsplit=14; minbucket=9; cp=0.078

## [Tune-y] 465: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 466: minsplit=16; minbucket=9; cp=0.078

## [Tune-y] 466: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 467: minsplit=17; minbucket=9; cp=0.078

## [Tune-y] 467: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 468: minsplit=18; minbucket=9; cp=0.078

## [Tune-y] 468: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 469: minsplit=19; minbucket=9; cp=0.078

## [Tune-y] 469: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 470: minsplit=20; minbucket=9; cp=0.078

## [Tune-y] 470: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 471: minsplit=10; minbucket=10; cp=0.078

## [Tune-y] 471: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 472: minsplit=11; minbucket=10; cp=0.078

## [Tune-y] 472: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 473: minsplit=12; minbucket=10; cp=0.078

## [Tune-y] 473: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 474: minsplit=13; minbucket=10; cp=0.078

## [Tune-y] 474: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 475: minsplit=14; minbucket=10; cp=0.078

## [Tune-y] 475: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 476: minsplit=16; minbucket=10; cp=0.078

## [Tune-y] 476: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 477: minsplit=17; minbucket=10; cp=0.078

## [Tune-y] 477: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 478: minsplit=18; minbucket=10; cp=0.078

## [Tune-y] 478: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 479: minsplit=19; minbucket=10; cp=0.078

## [Tune-y] 479: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 480: minsplit=20; minbucket=10; cp=0.078

## [Tune-y] 480: acc.test.mean=0.814; time: 0.0 min

## [Tune-x] 481: minsplit=10; minbucket=5; cp=0.089

## [Tune-y] 481: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 482: minsplit=11; minbucket=5; cp=0.089

## [Tune-y] 482: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 483: minsplit=12; minbucket=5; cp=0.089

## [Tune-y] 483: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 484: minsplit=13; minbucket=5; cp=0.089

## [Tune-y] 484: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 485: minsplit=14; minbucket=5; cp=0.089

## [Tune-y] 485: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 486: minsplit=16; minbucket=5; cp=0.089

## [Tune-y] 486: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 487: minsplit=17; minbucket=5; cp=0.089

## [Tune-y] 487: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 488: minsplit=18; minbucket=5; cp=0.089

## [Tune-y] 488: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 489: minsplit=19; minbucket=5; cp=0.089

## [Tune-y] 489: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 490: minsplit=20; minbucket=5; cp=0.089

## [Tune-y] 490: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 491: minsplit=10; minbucket=6; cp=0.089

## [Tune-y] 491: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 492: minsplit=11; minbucket=6; cp=0.089

## [Tune-y] 492: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 493: minsplit=12; minbucket=6; cp=0.089

## [Tune-y] 493: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 494: minsplit=13; minbucket=6; cp=0.089

## [Tune-y] 494: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 495: minsplit=14; minbucket=6; cp=0.089

## [Tune-y] 495: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 496: minsplit=16; minbucket=6; cp=0.089

## [Tune-y] 496: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 497: minsplit=17; minbucket=6; cp=0.089

## [Tune-y] 497: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 498: minsplit=18; minbucket=6; cp=0.089

## [Tune-y] 498: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 499: minsplit=19; minbucket=6; cp=0.089

## [Tune-y] 499: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 500: minsplit=20; minbucket=6; cp=0.089

## [Tune-y] 500: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 501: minsplit=10; minbucket=7; cp=0.089

## [Tune-y] 501: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 502: minsplit=11; minbucket=7; cp=0.089

## [Tune-y] 502: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 503: minsplit=12; minbucket=7; cp=0.089

## [Tune-y] 503: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 504: minsplit=13; minbucket=7; cp=0.089

## [Tune-y] 504: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 505: minsplit=14; minbucket=7; cp=0.089

## [Tune-y] 505: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 506: minsplit=16; minbucket=7; cp=0.089

## [Tune-y] 506: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 507: minsplit=17; minbucket=7; cp=0.089

## [Tune-y] 507: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 508: minsplit=18; minbucket=7; cp=0.089

## [Tune-y] 508: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 509: minsplit=19; minbucket=7; cp=0.089

## [Tune-y] 509: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 510: minsplit=20; minbucket=7; cp=0.089

## [Tune-y] 510: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 511: minsplit=10; minbucket=8; cp=0.089

## [Tune-y] 511: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 512: minsplit=11; minbucket=8; cp=0.089

## [Tune-y] 512: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 513: minsplit=12; minbucket=8; cp=0.089

## [Tune-y] 513: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 514: minsplit=13; minbucket=8; cp=0.089

## [Tune-y] 514: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 515: minsplit=14; minbucket=8; cp=0.089

## [Tune-y] 515: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 516: minsplit=16; minbucket=8; cp=0.089

## [Tune-y] 516: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 517: minsplit=17; minbucket=8; cp=0.089

## [Tune-y] 517: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 518: minsplit=18; minbucket=8; cp=0.089

## [Tune-y] 518: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 519: minsplit=19; minbucket=8; cp=0.089

## [Tune-y] 519: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 520: minsplit=20; minbucket=8; cp=0.089

## [Tune-y] 520: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 521: minsplit=10; minbucket=9; cp=0.089

## [Tune-y] 521: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 522: minsplit=11; minbucket=9; cp=0.089

## [Tune-y] 522: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 523: minsplit=12; minbucket=9; cp=0.089

## [Tune-y] 523: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 524: minsplit=13; minbucket=9; cp=0.089

## [Tune-y] 524: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 525: minsplit=14; minbucket=9; cp=0.089

## [Tune-y] 525: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 526: minsplit=16; minbucket=9; cp=0.089

## [Tune-y] 526: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 527: minsplit=17; minbucket=9; cp=0.089

## [Tune-y] 527: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 528: minsplit=18; minbucket=9; cp=0.089

## [Tune-y] 528: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 529: minsplit=19; minbucket=9; cp=0.089

## [Tune-y] 529: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 530: minsplit=20; minbucket=9; cp=0.089

## [Tune-y] 530: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 531: minsplit=10; minbucket=10; cp=0.089

## [Tune-y] 531: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 532: minsplit=11; minbucket=10; cp=0.089

## [Tune-y] 532: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 533: minsplit=12; minbucket=10; cp=0.089

## [Tune-y] 533: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 534: minsplit=13; minbucket=10; cp=0.089

## [Tune-y] 534: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 535: minsplit=14; minbucket=10; cp=0.089

## [Tune-y] 535: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 536: minsplit=16; minbucket=10; cp=0.089

## [Tune-y] 536: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 537: minsplit=17; minbucket=10; cp=0.089

## [Tune-y] 537: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 538: minsplit=18; minbucket=10; cp=0.089

## [Tune-y] 538: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 539: minsplit=19; minbucket=10; cp=0.089

## [Tune-y] 539: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 540: minsplit=20; minbucket=10; cp=0.089

## [Tune-y] 540: acc.test.mean=0.79; time: 0.0 min

## [Tune-x] 541: minsplit=10; minbucket=5; cp=0.1

## [Tune-y] 541: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 542: minsplit=11; minbucket=5; cp=0.1

## [Tune-y] 542: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 543: minsplit=12; minbucket=5; cp=0.1

## [Tune-y] 543: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 544: minsplit=13; minbucket=5; cp=0.1

## [Tune-y] 544: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 545: minsplit=14; minbucket=5; cp=0.1

## [Tune-y] 545: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 546: minsplit=16; minbucket=5; cp=0.1

## [Tune-y] 546: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 547: minsplit=17; minbucket=5; cp=0.1

## [Tune-y] 547: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 548: minsplit=18; minbucket=5; cp=0.1

## [Tune-y] 548: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 549: minsplit=19; minbucket=5; cp=0.1

## [Tune-y] 549: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 550: minsplit=20; minbucket=5; cp=0.1

## [Tune-y] 550: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 551: minsplit=10; minbucket=6; cp=0.1

## [Tune-y] 551: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 552: minsplit=11; minbucket=6; cp=0.1

## [Tune-y] 552: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 553: minsplit=12; minbucket=6; cp=0.1

## [Tune-y] 553: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 554: minsplit=13; minbucket=6; cp=0.1

## [Tune-y] 554: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 555: minsplit=14; minbucket=6; cp=0.1

## [Tune-y] 555: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 556: minsplit=16; minbucket=6; cp=0.1

## [Tune-y] 556: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 557: minsplit=17; minbucket=6; cp=0.1

## [Tune-y] 557: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 558: minsplit=18; minbucket=6; cp=0.1

## [Tune-y] 558: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 559: minsplit=19; minbucket=6; cp=0.1

## [Tune-y] 559: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 560: minsplit=20; minbucket=6; cp=0.1

## [Tune-y] 560: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 561: minsplit=10; minbucket=7; cp=0.1

## [Tune-y] 561: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 562: minsplit=11; minbucket=7; cp=0.1

## [Tune-y] 562: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 563: minsplit=12; minbucket=7; cp=0.1

## [Tune-y] 563: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 564: minsplit=13; minbucket=7; cp=0.1

## [Tune-y] 564: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 565: minsplit=14; minbucket=7; cp=0.1

## [Tune-y] 565: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 566: minsplit=16; minbucket=7; cp=0.1

## [Tune-y] 566: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 567: minsplit=17; minbucket=7; cp=0.1

## [Tune-y] 567: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 568: minsplit=18; minbucket=7; cp=0.1

## [Tune-y] 568: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 569: minsplit=19; minbucket=7; cp=0.1

## [Tune-y] 569: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 570: minsplit=20; minbucket=7; cp=0.1

## [Tune-y] 570: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 571: minsplit=10; minbucket=8; cp=0.1

## [Tune-y] 571: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 572: minsplit=11; minbucket=8; cp=0.1

## [Tune-y] 572: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 573: minsplit=12; minbucket=8; cp=0.1

## [Tune-y] 573: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 574: minsplit=13; minbucket=8; cp=0.1

## [Tune-y] 574: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 575: minsplit=14; minbucket=8; cp=0.1

## [Tune-y] 575: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 576: minsplit=16; minbucket=8; cp=0.1

## [Tune-y] 576: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 577: minsplit=17; minbucket=8; cp=0.1

## [Tune-y] 577: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 578: minsplit=18; minbucket=8; cp=0.1

## [Tune-y] 578: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 579: minsplit=19; minbucket=8; cp=0.1

## [Tune-y] 579: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 580: minsplit=20; minbucket=8; cp=0.1

## [Tune-y] 580: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 581: minsplit=10; minbucket=9; cp=0.1

## [Tune-y] 581: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 582: minsplit=11; minbucket=9; cp=0.1

## [Tune-y] 582: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 583: minsplit=12; minbucket=9; cp=0.1

## [Tune-y] 583: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 584: minsplit=13; minbucket=9; cp=0.1

## [Tune-y] 584: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 585: minsplit=14; minbucket=9; cp=0.1

## [Tune-y] 585: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 586: minsplit=16; minbucket=9; cp=0.1

## [Tune-y] 586: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 587: minsplit=17; minbucket=9; cp=0.1

## [Tune-y] 587: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 588: minsplit=18; minbucket=9; cp=0.1

## [Tune-y] 588: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 589: minsplit=19; minbucket=9; cp=0.1

## [Tune-y] 589: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 590: minsplit=20; minbucket=9; cp=0.1

## [Tune-y] 590: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 591: minsplit=10; minbucket=10; cp=0.1

## [Tune-y] 591: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 592: minsplit=11; minbucket=10; cp=0.1

## [Tune-y] 592: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 593: minsplit=12; minbucket=10; cp=0.1

## [Tune-y] 593: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 594: minsplit=13; minbucket=10; cp=0.1

## [Tune-y] 594: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 595: minsplit=14; minbucket=10; cp=0.1

## [Tune-y] 595: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 596: minsplit=16; minbucket=10; cp=0.1

## [Tune-y] 596: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 597: minsplit=17; minbucket=10; cp=0.1

## [Tune-y] 597: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 598: minsplit=18; minbucket=10; cp=0.1

## [Tune-y] 598: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 599: minsplit=19; minbucket=10; cp=0.1

## [Tune-y] 599: acc.test.mean=0.784; time: 0.0 min

## [Tune-x] 600: minsplit=20; minbucket=10; cp=0.1

## [Tune-y] 600: acc.test.mean=0.784; time: 0.0 min

## [Tune] Result: minsplit=12; minbucket=5; cp=0.001 : acc.test.mean=0.919

treetune$x

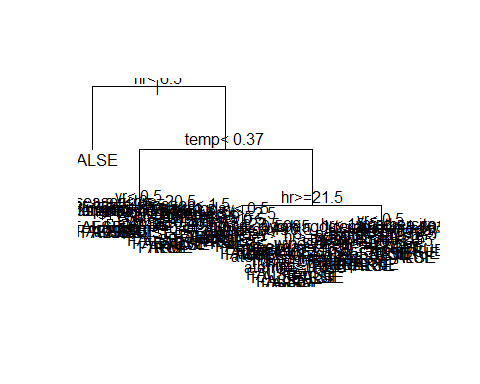
## $minsplit  
## [1] 12  
##   
## $minbucket  
## [1] 5  
##   
## $cp  
## [1] 0.001

tree = setHyperPars(cycletree, par.vals = treetune$x)  
traintree = train(tree, traintask)  
predicttree = predict(traintree, testtask)  
table(Test$countclass,predicttree$data$response)

##   
## FALSE TRUE  
## FALSE 1765 140  
## TRUE 115 1150

Here the Acuuracy is 91%

plot(traintree$learner.model)  
text(traintree$learner.model)

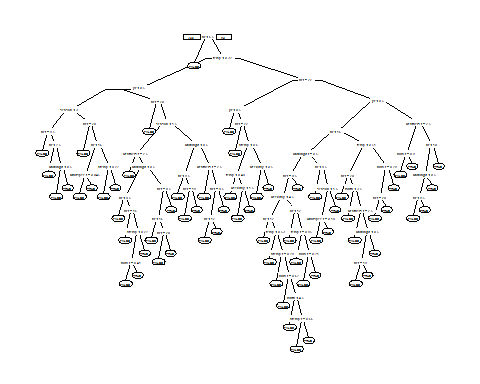


library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 3.4.2

## Loading required package: rpart

prp(traintree$learner.model)



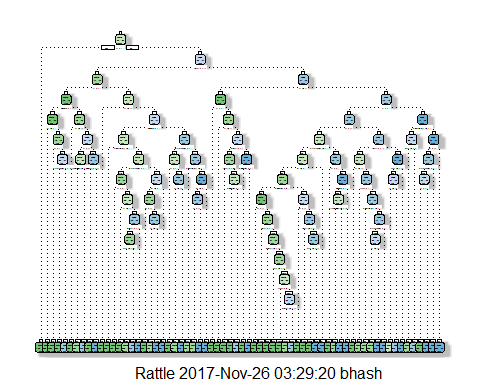
library(rattle)

## Warning: package 'rattle' was built under R version 3.4.2

## Rattle: A free graphical interface for data science with R.  
## Version 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

fancyRpartPlot(traintree$learner.model)

## Warning: labs do not fit even at cex 0.15, there may be some overplotting



# random forest classfier

randomforest = makeLearner("classif.randomForest",predict.type = "response", par.vals = list(ntree = 200, mtry = 3))  
randomforest$par.vals = list(importance = TRUE)  
  
randomparam <- makeParamSet(  
makeIntegerParam("ntree",lower = 50, upper = 450),  
makeIntegerParam("mtry", lower = 3, upper = 10),  
makeIntegerParam("nodesize", lower = 10, upper = 40)  
)  
randomcontrol = makeTuneControlRandom(maxit = 30L)  
randomcross = makeResampleDesc("CV",iter = 10L)  
randomtune <- tuneParams(learner = randomforest, resampling = randomcross, task = traintask, par.set = randomparam, control = randomcontrol, measures = acc)

## [Tune] Started tuning learner classif.randomForest for parameter set:

## Type len Def Constr Req Tunable Trafo  
## ntree integer - - 50 to 450 - TRUE -  
## mtry integer - - 3 to 10 - TRUE -  
## nodesize integer - - 10 to 40 - TRUE -

## With control class: TuneControlRandom

## Imputation value: -0

## [Tune-x] 1: ntree=383; mtry=4; nodesize=11

## [Tune-y] 1: acc.test.mean=0.926; time: 1.4 min

## [Tune-x] 2: ntree=59; mtry=10; nodesize=11

## [Tune-y] 2: acc.test.mean=0.935; time: 0.2 min

## [Tune-x] 3: ntree=436; mtry=4; nodesize=30

## [Tune-y] 3: acc.test.mean=0.915; time: 1.5 min

## [Tune-x] 4: ntree=221; mtry=7; nodesize=33

## [Tune-y] 4: acc.test.mean=0.923; time: 0.7 min

## [Tune-x] 5: ntree=265; mtry=3; nodesize=20

## [Tune-y] 5: acc.test.mean=0.912; time: 1.0 min

## [Tune-x] 6: ntree=444; mtry=9; nodesize=30

## [Tune-y] 6: acc.test.mean=0.927; time: 1.5 min

## [Tune-x] 7: ntree=338; mtry=3; nodesize=33

## [Tune-y] 7: acc.test.mean=0.905; time: 1.2 min

## [Tune-x] 8: ntree=161; mtry=3; nodesize=11

## [Tune-y] 8: acc.test.mean=0.918; time: 0.6 min

## [Tune-x] 9: ntree=124; mtry=4; nodesize=32

## [Tune-y] 9: acc.test.mean=0.914; time: 0.4 min

## [Tune-x] 10: ntree=79; mtry=7; nodesize=34

## [Tune-y] 10: acc.test.mean=0.924; time: 0.3 min

## [Tune-x] 11: ntree=111; mtry=5; nodesize=17

## [Tune-y] 11: acc.test.mean=0.926; time: 0.4 min

## [Tune-x] 12: ntree=58; mtry=7; nodesize=14

## [Tune-y] 12: acc.test.mean=0.931; time: 0.2 min

## [Tune-x] 13: ntree=169; mtry=10; nodesize=34

## [Tune-y] 13: acc.test.mean=0.926; time: 0.6 min

## [Tune-x] 14: ntree=441; mtry=3; nodesize=19

## [Tune-y] 14: acc.test.mean=0.915; time: 1.6 min

## [Tune-x] 15: ntree=404; mtry=7; nodesize=20

## [Tune-y] 15: acc.test.mean=0.929; time: 1.4 min

## [Tune-x] 16: ntree=110; mtry=6; nodesize=10

## [Tune-y] 16: acc.test.mean=0.933; time: 0.4 min

## [Tune-x] 17: ntree=409; mtry=7; nodesize=35

## [Tune-y] 17: acc.test.mean=0.923; time: 1.4 min

## [Tune-x] 18: ntree=351; mtry=10; nodesize=24

## [Tune-y] 18: acc.test.mean=0.93; time: 1.2 min

## [Tune-x] 19: ntree=198; mtry=6; nodesize=21

## [Tune-y] 19: acc.test.mean=0.926; time: 0.7 min

## [Tune-x] 20: ntree=79; mtry=3; nodesize=18

## [Tune-y] 20: acc.test.mean=0.912; time: 0.3 min

## [Tune-x] 21: ntree=53; mtry=3; nodesize=13

## [Tune-y] 21: acc.test.mean=0.916; time: 0.2 min

## [Tune-x] 22: ntree=269; mtry=5; nodesize=37

## [Tune-y] 22: acc.test.mean=0.917; time: 0.9 min

## [Tune-x] 23: ntree=363; mtry=10; nodesize=11

## [Tune-y] 23: acc.test.mean=0.935; time: 1.3 min

## [Tune-x] 24: ntree=340; mtry=3; nodesize=35

## [Tune-y] 24: acc.test.mean=0.904; time: 1.2 min

## [Tune-x] 25: ntree=294; mtry=10; nodesize=11

## [Tune-y] 25: acc.test.mean=0.935; time: 1.1 min

## [Tune-x] 26: ntree=199; mtry=10; nodesize=14

## [Tune-y] 26: acc.test.mean=0.934; time: 0.7 min

## [Tune-x] 27: ntree=343; mtry=10; nodesize=24

## [Tune-y] 27: acc.test.mean=0.93; time: 1.2 min

## [Tune-x] 28: ntree=182; mtry=5; nodesize=39

## [Tune-y] 28: acc.test.mean=0.915; time: 0.6 min

## [Tune-x] 29: ntree=154; mtry=3; nodesize=28

## [Tune-y] 29: acc.test.mean=0.905; time: 0.5 min

## [Tune-x] 30: ntree=192; mtry=7; nodesize=21

## [Tune-y] 30: acc.test.mean=0.928; time: 0.7 min

## [Tune] Result: ntree=294; mtry=10; nodesize=11 : acc.test.mean=0.935

randomtune$y

## acc.test.mean   
## 0.9354789

randomtune$x

## $ntree  
## [1] 294  
##   
## $mtry  
## [1] 10  
##   
## $nodesize  
## [1] 11

randomtree = setHyperPars(randomforest, par.vals = randomtune$x)  
randomtrain = train(randomtree, traintask)  
getLearnerModel(randomtrain)

##   
## Call:  
## randomForest(formula = f, data = data, classwt = classwt, cutoff = cutoff, importance = TRUE, ntree = 294L, mtry = 10L, nodesize = 11L)   
## Type of random forest: classification  
## Number of trees: 294  
## No. of variables tried at each split: 10  
##   
## OOB estimate of error rate: 6.45%  
## Confusion matrix:  
## FALSE TRUE class.error  
## FALSE 7258 423 0.05507095  
## TRUE 403 4718 0.07869557

randompredict = predict(randomtrain,testtask)  
table(Test$countclass, randompredict$data$response)

##   
## FALSE TRUE  
## FALSE 1801 104  
## TRUE 68 1197

Here the accuracy is 94.5 %

svmlearner = makeLearner("classif.ksvm", predict.type = "response")  
  
randomcross = makeResampleDesc("CV",iters = 10L)  
  
svmparameter<- makeParamSet(makeNumericParam("C", lower = -5, upper = 5, trafo = function(x) 2^x),  
 makeDiscreteParam("sigma", values = 2^c(-8,-4,0,4))) #RBF Kernel Parameter  
  
svmcontrol = makeTuneControlGrid()  
svmtune= tuneParams("classif.ksvm", task = traintask, resampling = randomcross, par.set = svmparameter, control = svmcontrol, measures = acc)

## [Tune] Started tuning learner classif.ksvm for parameter set:

## Type len Def Constr Req Tunable Trafo  
## C numeric - - -5 to 5 - TRUE Y  
## sigma discrete - - 0.00390625,0.0625,1,16 - TRUE -

## With control class: TuneControlGrid

## Imputation value: -0

## [Tune-x] 1: C=0.0312; sigma=0.00390625

## [Tune-y] 1: acc.test.mean=0.772; time: 1.9 min

## [Tune-x] 2: C=0.0675; sigma=0.00390625

## [Tune-y] 2: acc.test.mean=0.776; time: 1.8 min

## [Tune-x] 3: C=0.146; sigma=0.00390625

## [Tune-y] 3: acc.test.mean=0.78; time: 1.7 min

## [Tune-x] 4: C=0.315; sigma=0.00390625

## [Tune-y] 4: acc.test.mean=0.783; time: 1.6 min

## [Tune-x] 5: C=0.68; sigma=0.00390625

## [Tune-y] 5: acc.test.mean=0.789; time: 1.6 min

## [Tune-x] 6: C=1.47; sigma=0.00390625

## [Tune-y] 6: acc.test.mean=0.795; time: 1.5 min

## [Tune-x] 7: C=3.17; sigma=0.00390625

## [Tune-y] 7: acc.test.mean=0.804; time: 1.5 min

## [Tune-x] 8: C=6.86; sigma=0.00390625

## [Tune-y] 8: acc.test.mean=0.812; time: 1.4 min

## [Tune-x] 9: C=14.8; sigma=0.00390625

## [Tune-y] 9: acc.test.mean=0.821; time: 1.4 min

## [Tune-x] 10: C=32; sigma=0.00390625

## [Tune-y] 10: acc.test.mean=0.833; time: 1.3 min

## [Tune-x] 11: C=0.0312; sigma=0.0625

## [Tune-y] 11: acc.test.mean=0.796; time: 1.2 min

## [Tune-x] 12: C=0.0675; sigma=0.0625

## [Tune-y] 12: acc.test.mean=0.805; time: 1.1 min

## [Tune-x] 13: C=0.146; sigma=0.0625

## [Tune-y] 13: acc.test.mean=0.816; time: 1.0 min

## [Tune-x] 14: C=0.315; sigma=0.0625

## [Tune-y] 14: acc.test.mean=0.83; time: 1.0 min

## [Tune-x] 15: C=0.68; sigma=0.0625

## [Tune-y] 15: acc.test.mean=0.845; time: 0.9 min

## [Tune-x] 16: C=1.47; sigma=0.0625

## [Tune-y] 16: acc.test.mean=0.854; time: 0.9 min

## [Tune-x] 17: C=3.17; sigma=0.0625

## [Tune-y] 17: acc.test.mean=0.863; time: 0.9 min

## [Tune-x] 18: C=6.86; sigma=0.0625

## [Tune-y] 18: acc.test.mean=0.868; time: 0.9 min

## [Tune-x] 19: C=14.8; sigma=0.0625

## [Tune-y] 19: acc.test.mean=0.873; time: 1.0 min

## [Tune-x] 20: C=32; sigma=0.0625

## [Tune-y] 20: acc.test.mean=0.876; time: 1.2 min

## [Tune-x] 21: C=0.0312; sigma=1

## [Tune-y] 21: acc.test.mean=0.626; time: 1.9 min

## [Tune-x] 22: C=0.0675; sigma=1

## [Tune-y] 22: acc.test.mean=0.695; time: 1.9 min

## [Tune-x] 23: C=0.146; sigma=1

## [Tune-y] 23: acc.test.mean=0.776; time: 2.0 min

## [Tune-x] 24: C=0.315; sigma=1

## [Tune-y] 24: acc.test.mean=0.825; time: 2.3 min

## [Tune-x] 25: C=0.68; sigma=1

## [Tune-y] 25: acc.test.mean=0.849; time: 2.7 min

## [Tune-x] 26: C=1.47; sigma=1

## [Tune-y] 26: acc.test.mean=0.852; time: 3.2 min

## [Tune-x] 27: C=3.17; sigma=1

## [Tune-y] 27: acc.test.mean=0.852; time: 3.7 min

## [Tune-x] 28: C=6.86; sigma=1

## [Tune-y] 28: acc.test.mean=0.847; time: 4.2 min

## [Tune-x] 29: C=14.8; sigma=1

## [Tune-y] 29: acc.test.mean=0.846; time: 4.3 min

## [Tune-x] 30: C=32; sigma=1

## [Tune-y] 30: acc.test.mean=0.844; time: 4.4 min

## [Tune-x] 31: C=0.0312; sigma=16

## [Tune-y] 31: acc.test.mean= 0.6; time: 3.7 min

## [Tune-x] 32: C=0.0675; sigma=16

## [Tune-y] 32: acc.test.mean= 0.6; time: 3.8 min

## [Tune-x] 33: C=0.146; sigma=16

## [Tune-y] 33: acc.test.mean= 0.6; time: 4.0 min

## [Tune-x] 34: C=0.315; sigma=16

## [Tune-y] 34: acc.test.mean= 0.6; time: 4.2 min

## [Tune-x] 35: C=0.68; sigma=16

## [Tune-y] 35: acc.test.mean=0.616; time: 4.4 min

## [Tune-x] 36: C=1.47; sigma=16

## [Tune-y] 36: acc.test.mean=0.675; time: 6.9 min

## [Tune-x] 37: C=3.17; sigma=16

## [Tune-y] 37: acc.test.mean=0.676; time: 6.9 min

## [Tune-x] 38: C=6.86; sigma=16

## [Tune-y] 38: acc.test.mean=0.676; time: 7.0 min

## [Tune-x] 39: C=14.8; sigma=16

## [Tune-y] 39: acc.test.mean=0.676; time: 7.0 min

## [Tune-x] 40: C=32; sigma=16

## [Tune-y] 40: acc.test.mean=0.676; time: 7.0 min

## [Tune] Result: C=32; sigma=0.0625 : acc.test.mean=0.876

svmtune$y

## acc.test.mean   
## 0.8757983

svmtune$x

## $C  
## [1] 32  
##   
## $sigma  
## [1] 0.0625

svmmodel = setHyperPars(svmlearner,par.vals = svmtune$x)  
svmtrain = train(svmmodel, traintask)  
getLearnerModel(svmtrain)

## Support Vector Machine object of class "ksvm"   
##   
## SV type: C-svc (classification)   
## parameter : cost C = 32   
##   
## Gaussian Radial Basis kernel function.   
## Hyperparameter : sigma = 0.0625   
##   
## Number of Support Vectors : 4098   
##   
## Objective Function Value : -109243.2

predictsvm = predict(svmtrain, testtask)  
table (Test$countclass , predictsvm$data$response)

##   
## FALSE TRUE  
## FALSE 1711 194  
## TRUE 196 1069

0.87

# bossting

boost = makeLearner("classif.gbm", predict.type = "response")  
  
gbmcontrol = makeTuneControlRandom(maxit = 40L)  
  
gbmcv = makeResampleDesc("CV",iters = 10L)  
  
  
  
  
  
  
gbmparam = makeParamSet(makeDiscreteParam("distribution", values = "bernoulli"),  
makeIntegerParam("n.trees", lower = 500, upper = 1000), #number of trees  
makeIntegerParam("interaction.depth", lower = 2, upper = 6), #depth of tree  
makeIntegerParam("n.minobsinnode", lower = 10, upper = 50),  
makeNumericParam("shrinkage",lower = 0.01, upper = 0.7))  
  
  
  
gbmtune = tuneParams(learner = boost, task = traintask, par.set = gbmparam, control = gbmcontrol, measures = acc, resampling = gbmcv)

## [Tune] Started tuning learner classif.gbm for parameter set:

## Type len Def Constr Req Tunable Trafo  
## distribution discrete - - bernoulli - TRUE -  
## n.trees integer - - 500 to 1e+03 - TRUE -  
## interaction.depth integer - - 2 to 6 - TRUE -  
## n.minobsinnode integer - - 10 to 50 - TRUE -  
## shrinkage numeric - - 0.01 to 0.7 - TRUE -

## With control class: TuneControlRandom

## Imputation value: -0

## [Tune-x] 1: distribution=bernoulli; n.trees=746; interaction.depth=4; n.minobsinnode=16; shrinkage=0.543

## [Tune-y] 1: acc.test.mean=0.921; time: 1.1 min

## [Tune-x] 2: distribution=bernoulli; n.trees=680; interaction.depth=6; n.minobsinnode=34; shrinkage=0.337

## [Tune-y] 2: acc.test.mean=0.932; time: 1.3 min

## [Tune-x] 3: distribution=bernoulli; n.trees=567; interaction.depth=5; n.minobsinnode=38; shrinkage=0.274

## [Tune-y] 3: acc.test.mean=0.937; time: 0.9 min

## [Tune-x] 4: distribution=bernoulli; n.trees=894; interaction.depth=3; n.minobsinnode=27; shrinkage=0.275

## [Tune-y] 4: acc.test.mean=0.935; time: 1.1 min

## [Tune-x] 5: distribution=bernoulli; n.trees=698; interaction.depth=5; n.minobsinnode=10; shrinkage=0.029

## [Tune-y] 5: acc.test.mean=0.932; time: 1.2 min

## [Tune-x] 6: distribution=bernoulli; n.trees=682; interaction.depth=5; n.minobsinnode=20; shrinkage=0.36

## [Tune-y] 6: acc.test.mean=0.929; time: 1.1 min

## [Tune-x] 7: distribution=bernoulli; n.trees=768; interaction.depth=5; n.minobsinnode=11; shrinkage=0.132

## [Tune-y] 7: acc.test.mean=0.939; time: 1.3 min

## [Tune-x] 8: distribution=bernoulli; n.trees=742; interaction.depth=6; n.minobsinnode=28; shrinkage=0.349

## [Tune-y] 8: acc.test.mean=0.928; time: 1.4 min

## [Tune-x] 9: distribution=bernoulli; n.trees=598; interaction.depth=6; n.minobsinnode=35; shrinkage=0.282

## [Tune-y] 9: acc.test.mean=0.933; time: 1.1 min

## [Tune-x] 10: distribution=bernoulli; n.trees=634; interaction.depth=4; n.minobsinnode=30; shrinkage=0.118

## [Tune-y] 10: acc.test.mean=0.94; time: 0.9 min

## [Tune-x] 11: distribution=bernoulli; n.trees=646; interaction.depth=4; n.minobsinnode=23; shrinkage=0.603

## [Tune-y] 11: acc.test.mean=0.916; time: 0.9 min

## [Tune-x] 12: distribution=bernoulli; n.trees=513; interaction.depth=6; n.minobsinnode=39; shrinkage=0.498

## [Tune-y] 12: acc.test.mean=0.929; time: 1.0 min

## [Tune-x] 13: distribution=bernoulli; n.trees=654; interaction.depth=2; n.minobsinnode=38; shrinkage=0.0948

## [Tune-y] 13: acc.test.mean=0.923; time: 0.6 min

## [Tune-x] 14: distribution=bernoulli; n.trees=937; interaction.depth=2; n.minobsinnode=29; shrinkage=0.372

## [Tune-y] 14: acc.test.mean=0.93; time: 0.9 min

## [Tune-x] 15: distribution=bernoulli; n.trees=882; interaction.depth=4; n.minobsinnode=41; shrinkage=0.649

## [Tune-y] 15: acc.test.mean=0.922; time: 1.2 min

## [Tune-x] 16: distribution=bernoulli; n.trees=559; interaction.depth=2; n.minobsinnode=38; shrinkage=0.0144

## [Tune-y] 16: acc.test.mean=0.862; time: 0.5 min

## [Tune-x] 17: distribution=bernoulli; n.trees=558; interaction.depth=5; n.minobsinnode=48; shrinkage=0.128

## [Tune-y] 17: acc.test.mean=0.939; time: 0.9 min

## [Tune-x] 18: distribution=bernoulli; n.trees=811; interaction.depth=3; n.minobsinnode=33; shrinkage=0.239

## [Tune-y] 18: acc.test.mean=0.937; time: 1.0 min

## [Tune-x] 19: distribution=bernoulli; n.trees=806; interaction.depth=5; n.minobsinnode=25; shrinkage=0.628

## [Tune-y] 19: acc.test.mean=0.918; time: 1.3 min

## [Tune-x] 20: distribution=bernoulli; n.trees=976; interaction.depth=6; n.minobsinnode=44; shrinkage=0.34

## [Tune-y] 20: acc.test.mean=0.93; time: 1.8 min

## [Tune-x] 21: distribution=bernoulli; n.trees=784; interaction.depth=4; n.minobsinnode=31; shrinkage=0.464

## [Tune-y] 21: acc.test.mean=0.928; time: 1.1 min

## [Tune-x] 22: distribution=bernoulli; n.trees=972; interaction.depth=6; n.minobsinnode=34; shrinkage=0.492

## [Tune-y] 22: acc.test.mean=0.919; time: 1.8 min

## [Tune-x] 23: distribution=bernoulli; n.trees=564; interaction.depth=6; n.minobsinnode=34; shrinkage=0.492

## [Tune-y] 23: acc.test.mean=0.927; time: 1.1 min

## [Tune-x] 24: distribution=bernoulli; n.trees=954; interaction.depth=6; n.minobsinnode=48; shrinkage=0.507

## [Tune-y] 24: acc.test.mean=0.924; time: 1.8 min

## [Tune-x] 25: distribution=bernoulli; n.trees=884; interaction.depth=5; n.minobsinnode=24; shrinkage=0.361

## [Tune-y] 25: acc.test.mean=0.927; time: 1.4 min

## [Tune-x] 26: distribution=bernoulli; n.trees=954; interaction.depth=6; n.minobsinnode=11; shrinkage=0.292

## [Tune-y] 26: acc.test.mean=0.928; time: 1.7 min

## [Tune-x] 27: distribution=bernoulli; n.trees=845; interaction.depth=3; n.minobsinnode=33; shrinkage=0.696

## [Tune-y] 27: acc.test.mean=0.91; time: 1.0 min

## [Tune-x] 28: distribution=bernoulli; n.trees=771; interaction.depth=5; n.minobsinnode=40; shrinkage=0.0529

## [Tune-y] 28: acc.test.mean=0.939; time: 1.3 min

## [Tune-x] 29: distribution=bernoulli; n.trees=702; interaction.depth=4; n.minobsinnode=46; shrinkage=0.415

## [Tune-y] 29: acc.test.mean=0.934; time: 1.0 min

## [Tune-x] 30: distribution=bernoulli; n.trees=842; interaction.depth=6; n.minobsinnode=35; shrinkage=0.345

## [Tune-y] 30: acc.test.mean=0.93; time: 1.6 min

## [Tune-x] 31: distribution=bernoulli; n.trees=515; interaction.depth=5; n.minobsinnode=22; shrinkage=0.307

## [Tune-y] 31: acc.test.mean=0.934; time: 0.8 min

## [Tune-x] 32: distribution=bernoulli; n.trees=896; interaction.depth=4; n.minobsinnode=19; shrinkage=0.465

## [Tune-y] 32: acc.test.mean=0.928; time: 1.3 min

## [Tune-x] 33: distribution=bernoulli; n.trees=789; interaction.depth=2; n.minobsinnode=20; shrinkage=0.461

## [Tune-y] 33: acc.test.mean=0.927; time: 0.7 min

## [Tune-x] 34: distribution=bernoulli; n.trees=649; interaction.depth=2; n.minobsinnode=21; shrinkage=0.0335

## [Tune-y] 34: acc.test.mean=0.884; time: 0.6 min

## [Tune-x] 35: distribution=bernoulli; n.trees=721; interaction.depth=3; n.minobsinnode=23; shrinkage=0.0154

## [Tune-y] 35: acc.test.mean=0.892; time: 0.9 min

## [Tune-x] 36: distribution=bernoulli; n.trees=941; interaction.depth=4; n.minobsinnode=50; shrinkage=0.233

## [Tune-y] 36: acc.test.mean=0.935; time: 1.3 min

## [Tune-x] 37: distribution=bernoulli; n.trees=518; interaction.depth=4; n.minobsinnode=50; shrinkage=0.321

## [Tune-y] 37: acc.test.mean=0.934; time: 0.7 min

## [Tune-x] 38: distribution=bernoulli; n.trees=782; interaction.depth=6; n.minobsinnode=10; shrinkage=0.567

## [Tune-y] 38: acc.test.mean=0.92; time: 1.4 min

## [Tune-x] 39: distribution=bernoulli; n.trees=699; interaction.depth=6; n.minobsinnode=43; shrinkage=0.347

## [Tune-y] 39: acc.test.mean=0.93; time: 1.3 min

## [Tune-x] 40: distribution=bernoulli; n.trees=636; interaction.depth=3; n.minobsinnode=24; shrinkage=0.202

## [Tune-y] 40: acc.test.mean=0.935; time: 0.8 min

## [Tune] Result: distribution=bernoulli; n.trees=634; interaction.depth=4; n.minobsinnode=30; shrinkage=0.118 : acc.test.mean=0.94

gbmtune$y

## acc.test.mean   
## 0.9396958

gbmtune$x

## $distribution  
## [1] "bernoulli"  
##   
## $n.trees  
## [1] 634  
##   
## $interaction.depth  
## [1] 4  
##   
## $n.minobsinnode  
## [1] 30  
##   
## $shrinkage  
## [1] 0.1184034

gbmboost = setHyperPars(boost,par.vals = gbmtune$x)

gbmtrain = train(gbmboost,traintask)  
gbmpredict = predict(gbmtrain,testtask)  
table(Test$countclass,gbmpredict$data$response)

##   
## FALSE TRUE  
## FALSE 1811 94  
## TRUE 74 1191

94 % Accuracy

# lda

ldalearner = makeLearner("classif.lda", predict.type = "response")  
ldatrain = train(ldalearner, traintask)  
predictlda = predict(ldatrain, testtask)  
table(Test$countclass, predictlda$data$response)

##   
## FALSE TRUE  
## FALSE 1586 319  
## TRUE 401 864

the accuracy is 77% #svm linear

svmlearnerl = makeLearner("classif.ksvm", predict.type = "response")  
  
randomcross = makeResampleDesc("CV",iters = 3L)  
  
svmparameter<- makeParamSet(makeNumericParam("C", lower = -5, upper = 5, trafo = function(x) 2^x))  
  
  
svmcontrol = makeTuneControlGrid()  
svmtunel = tuneParams(svmlearnerl, task = traintask, resampling = randomcross, par.set = svmparameter, control = svmcontrol, measures = acc)

## [Tune] Started tuning learner classif.ksvm for parameter set:

## Type len Def Constr Req Tunable Trafo  
## C numeric - - -5 to 5 - TRUE Y

## With control class: TuneControlGrid

## Imputation value: -0

## [Tune-x] 1: C=0.0312

## [Tune-y] 1: acc.test.mean=0.793; time: 0.3 min

## [Tune-x] 2: C=0.0675

## [Tune-y] 2: acc.test.mean=0.801; time: 0.2 min

## [Tune-x] 3: C=0.146

## [Tune-y] 3: acc.test.mean=0.813; time: 0.2 min

## [Tune-x] 4: C=0.315

## [Tune-y] 4: acc.test.mean=0.825; time: 0.2 min

## [Tune-x] 5: C=0.68

## [Tune-y] 5: acc.test.mean=0.839; time: 0.2 min

## [Tune-x] 6: C=1.47

## [Tune-y] 6: acc.test.mean=0.851; time: 0.2 min

## [Tune-x] 7: C=3.17

## [Tune-y] 7: acc.test.mean=0.859; time: 0.2 min

## [Tune-x] 8: C=6.86

## [Tune-y] 8: acc.test.mean=0.863; time: 0.2 min

## [Tune-x] 9: C=14.8

## [Tune-y] 9: acc.test.mean=0.868; time: 0.2 min

## [Tune-x] 10: C=32

## [Tune-y] 10: acc.test.mean=0.872; time: 0.2 min

## [Tune] Result: C=32 : acc.test.mean=0.872

svmtune$x

## $C  
## [1] 32  
##   
## $sigma  
## [1] 0.0625

svmtunel$y

## acc.test.mean   
## 0.8715826

svmmodell = setHyperPars(svmlearnerl,par.vals = svmtunel$x)  
svmtrainl = train(svmmodell, traintask)  
getLearnerModel(svmtrainl)

## Support Vector Machine object of class "ksvm"   
##   
## SV type: C-svc (classification)   
## parameter : cost C = 32   
##   
## Gaussian Radial Basis kernel function.   
## Hyperparameter : sigma = 0.0621324884133046   
##   
## Number of Support Vectors : 4103   
##   
## Objective Function Value : -109401.7

predictsvml = predict(svmtrainl, testtask)  
table (Test$countclass, predictsvml$data$response)

##   
## FALSE TRUE  
## FALSE 1709 196  
## TRUE 196 1069

the accuracy is 88%

dim(Train)

## [1] 12802 18

library(class)  
  
Train1 = Train[3:14]  
Test1 = Test[3:14]  
model=knn(train=Train1,test = Test1, cl=Train$countclass,k=10)  
summary(model)

## FALSE TRUE   
## 1841 1329

table(Test$countclass,model)

## model  
## FALSE TRUE  
## FALSE 1765 140  
## TRUE 76 1189

For k = 10 accuracy is 93%

for Higher the K value, the results are not better as the cluster stays very close to each other.

names(Train)

## [1] "instant" "dteday" "season" "yr" "mnth"   
## [6] "hr" "holiday" "weekday" "workingday" "weathersit"  
## [11] "temp" "atemp" "hum" "windspeed" "casual"   
## [16] "registered" "cnt" "countclass"