EoS\_SO2\_SVM\_02.R

Marcelo

Fri Mar 03 15:13:12 2017

setwd("C:/Users/Marcelo/Desktop/EoS SO2/")  
  
rm(list = ls())  
# Function  
rmse <- function(error)  
{  
 sqrt(mean(error^2))  
}  
  
# Loadind data sets  
library(data.table)  
library(dplyr)

## -------------------------------------------------------------------------

## data.table + dplyr code now lives in dtplyr.  
## Please library(dtplyr)!

## -------------------------------------------------------------------------

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':  
##   
## between, first, last

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

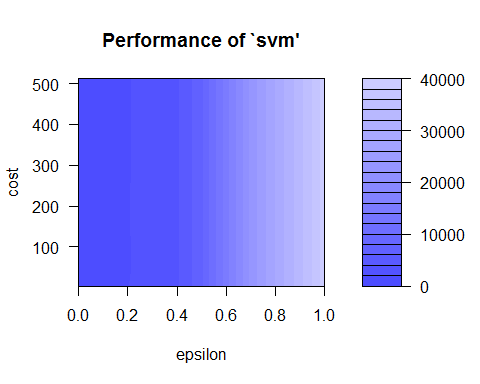
library(ggplot2)  
train <- read.table("data\_train.txt", sep = " ", header = F, stringsAsFactors = F)  
train <- train[,1:3]  
names(train) <- c("temp", "pressao", "densid")  
  
test <- read.table("data\_test.txt", sep = " ", header = F, stringsAsFactors = F)  
test <- test[,1:3]  
names(test) <- c("temp", "pressao", "densid")  
  
# train\_norm <- as.data.frame(scale(train)) também funciona  
# train\_scaled <- train %>%  
# mutate\_each\_(funs(scale(.) %>% as.vector),  
# vars = c("temp", "pressao", "densid"))  
  
#test\_scaled <- test %>%  
# mutate\_each\_(funs(scale(.) %>% as.vector),  
# vars = c("temp", "pressao", "densid"))  
  
library(caTools)  
set.seed(123)  
# Split trains into trains and val  
split = sample.split(train$temp, SplitRatio = 0.85)  
  
# Separating train and test datasets  
train\_nn <- subset(train, split == TRUE)  
val\_nn <- subset(train, split == FALSE)  
  
library(e1071)  
  
model\_svm <- svm(densid ~ . , data = train\_nn, scale = TRUE, kernel = 'radial',  
 cachesize = 400, cross = 5, epsilon = 0.2)  
  
svm\_val <- predict(model\_svm, val\_nn[1:2])  
  
error <- val\_nn$densid - svm\_val  
RMSE <- rmse(error)  
RMSE

## [1] 39.36183

# tunning model  
tuneResult <- tune(svm, densid ~ . , data = train\_nn,scale = TRUE, kernel = 'radial',  
 cachesize = 400, ranges = list(epsilon = seq(0,1,0.1), cost = 2^(2:9)))  
  
print(tuneResult)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## epsilon cost  
## 0 512  
##   
## - best performance: 501.2545

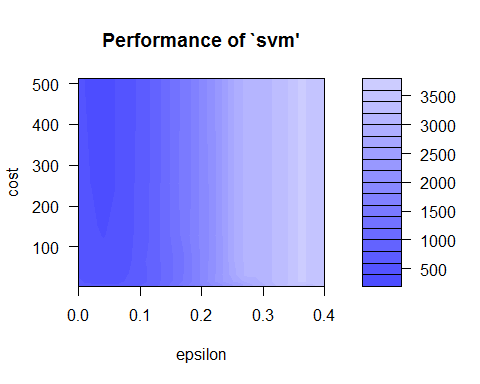
plot(tuneResult)



# Refining model  
tuneResult <- tune(svm, densid ~ . , data = train\_nn,scale = TRUE, kernel = 'radial',  
 cachesize = 400, ranges = list(epsilon = seq(0,0.4,0.01), cost = 2^(2:9)))  
  
print(tuneResult)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## epsilon cost  
## 0.03 512  
##   
## - best performance: 324.0777

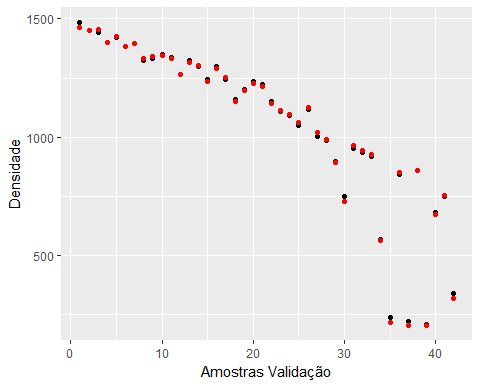
plot(tuneResult)



tuneModel <- tuneResult$best.model  
tuneModelY <- predict(tuneModel, val\_nn[1:2])  
  
errorTunedModel <- val\_nn$densid - tuneModelY  
tunedModelRMSE <-rmse(errorTunedModel)  
tunedModelRMSE

## [1] 9.800437

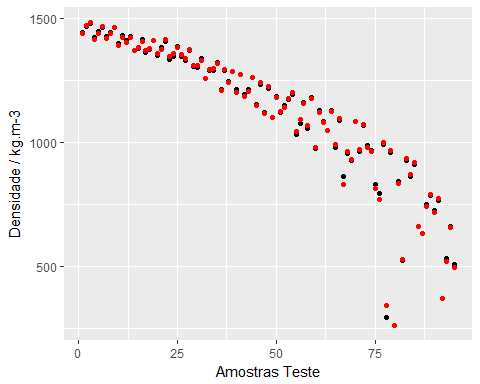
results\_val\_tuned\_Model <- as.data.frame(cbind(val\_nn, tuneModelY))  
names(results\_val\_tuned\_Model) <- c("temp", "pressao", "densid", "test.svm")  
  
ggplot(results\_val\_tuned\_Model, aes(x = 1:nrow(val\_nn), y = densid))+  
 geom\_point()+  
 geom\_point(aes(y = test.svm), col = "red")+  
 xlab("Amostras Validação")+  
 ylab("Densidade")



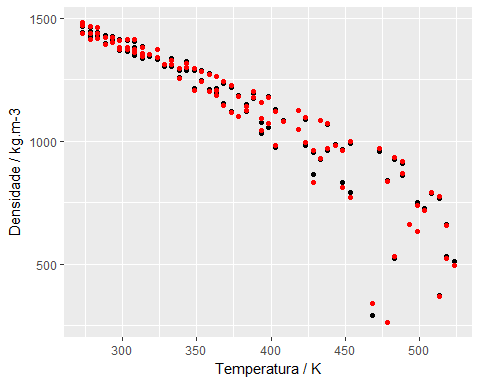
# Predicting density in the test data set  
tunedModel\_test <- predict(tuneModel, test[1:2])  
  
error\_test <- test$densid - tunedModel\_test  
error\_testRMSE <-rmse(error\_test)  
error\_testRMSE

## [1] 9.563887

results\_test <- as.data.frame(cbind(test, tunedModel\_test))  
names(results\_test) <- c("temp", "pressao", "densid", "test.svm")  
  
ggplot(results\_test, aes(x = 1:nrow(test), y = densid))+  
 geom\_point()+  
 geom\_point(aes(y = test.svm), col = "red")+  
 xlab("Amostras Teste")+  
 ylab("Densidade / kg.m-3")



ggplot(results\_test, aes(x = temp, y = densid))+  
 geom\_point()+  
 geom\_point(aes(y = test.svm), col = "red")+  
 xlab("Temperatura / K")+  
 ylab("Densidade / kg.m-3")



ggplot(results\_test, aes(x = pressao, y = densid))+  
 geom\_point()+  
 geom\_point(aes(y = test.svm), col = "red")+  
 xlab("Pressão / MPa")+  
 ylab("Densidade / kg.m-3")

