Group\_Project\_SVM.R

songyijin

2021-12-03

rm(list=ls()) # Clear environment  
cat("\014") # Clear Console

dev.off() # Clear plots

## null device   
## 1

# packages  
library(dplyr) # For data preparation

## Registered S3 methods overwritten by 'tibble':  
## method from   
## format.tbl pillar  
## print.tbl pillar

##   
## Attaching package: 'dplyr'

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

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

library(ROCR) # For evaluation metrics  
library(caret) # For confusion matrix

## Loading required package: ggplot2

## Loading required package: lattice

library(e1071) # For svm  
  
#read data  
hotel\_data <- read.csv("hotel\_bookings.csv")  
#colnames(hotel\_data)  
#dim(hotel\_data)  
  
#data preparation  
# Move out the obvirous outier  
hotel\_data<-subset(hotel\_data,hotel\_data$adr<4000)  
# classification data set as factor  
hotel\_data<-hotel\_data%>%  
 mutate(  
 hotel=as.factor(hotel),   
 is\_canceled=as.factor(is\_canceled),  
 meal=as.factor(meal),  
 country=as.factor(country),  
 market\_segment=as.factor(market\_segment),  
 distribution\_channel=as.factor(distribution\_channel),  
 is\_repeated\_guest=as.factor(is\_repeated\_guest),  
 reserved\_room\_type=as.factor(reserved\_room\_type),  
 assigned\_room\_type=as.factor(assigned\_room\_type),  
 deposit\_type=as.factor(deposit\_type),  
 customer\_type=as.factor(customer\_type),  
 reservation\_status=as.factor(reservation\_status),  
 agent=as.factor(agent),  
 company=as.factor(company),  
 arrival\_date\_day\_of\_month=as.factor(arrival\_date\_day\_of\_month),  
 arrival\_date\_month=as.factor(arrival\_date\_month),  
 arrival\_date\_year=as.factor(arrival\_date\_year)  
   
)  
hotel\_data[is.na(hotel\_data)]<-0  
  
# training data and test data preparation  
a = 1:nrow(hotel\_data)  
set.seed(1)  
ind = sample(a, floor(nrow(hotel\_data)\*0.9), replace=FALSE)   
train\_choose = hotel\_data[ind,]  
test = hotel\_data[-ind,]  
# SVM model encounter with problem with large scale of data  
# choose actual train data again(reduce the scale)  
b = 1:nrow(train\_choose)  
set.seed(1)  
ind = sample(a, floor(nrow(train\_choose)\*0.2), replace=FALSE)   
train = train\_choose[ind,]  
  
# show the changes  
#table(hotel\_data$hotel)  
#table(hotel\_data$is\_canceled)  
#table(hotel\_data$lead\_time)  
#table(hotel\_data$adults)  
#table(hotel\_data$children)  
#table(hotel\_data$babies)  
#table(hotel\_data$meal)  
#table(hotel\_data$market\_segment)  
#table(hotel\_data$distribution\_channel)  
#table(hotel\_data$is\_repeated\_guest)  
#table(hotel\_data$previous\_cancellations)  
#table(hotel\_data$previous\_bookings\_not\_canceled)  
#table(hotel\_data$reserved\_room\_type)  
#table(hotel\_data$deposit\_type)  
#table(hotel\_data$days\_in\_waiting\_list)  
#table(hotel\_data$customer\_type)  
#table(hotel\_data$adr)  
#table(hotel\_data$required\_car\_parking\_spaces)  
#table(hotel\_data$is\_repeated\_guest)  
  
  
train\_data <- train[c('hotel','is\_canceled','lead\_time','adults','children','babies','meal',  
 'market\_segment','distribution\_channel','is\_repeated\_guest',  
 'previous\_cancellations','previous\_bookings\_not\_canceled','reserved\_room\_type',  
 'deposit\_type','days\_in\_waiting\_list','customer\_type','adr',  
 'required\_car\_parking\_spaces')]  
test\_data <- test[c('hotel','is\_canceled','lead\_time','adults','children','babies','meal',  
 'market\_segment','distribution\_channel','is\_repeated\_guest',  
 'previous\_cancellations','previous\_bookings\_not\_canceled','reserved\_room\_type',  
 'deposit\_type','days\_in\_waiting\_list','customer\_type','adr',  
 'required\_car\_parking\_spaces')]  
  
tune\_out\_prob=tune(svm,  
 is\_canceled~.,  
 data=train\_data,   
 ranges =list(cost=c(0.01,5,10,100)), probability=TRUE)  
summary(tune\_out\_prob)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost  
## 100  
##   
## - best performance: 0.2139986   
##   
## - Detailed performance results:  
## cost error dispersion  
## 1 1e-02 0.2552056 0.005271494  
## 2 5e+00 0.2166880 0.005761204  
## 3 1e+01 0.2164345 0.005945663  
## 4 1e+02 0.2139986 0.006385018

#apply to the test data  
best\_mod\_prob = tune\_out\_prob$best.model # best\_mod is trained based on entire data  
summary(best\_mod\_prob)

##   
## Call:  
## best.tune(method = svm, train.x = is\_canceled ~ ., data = train\_data,   
## ranges = list(cost = c(0.01, 5, 10, 100)), probability = TRUE)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 100   
##   
## Number of Support Vectors: 8832  
##   
## ( 4561 4271 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## 0 1

best.pred = predict(best\_mod\_prob, newdata = test\_data, probability=TRUE)  
acc\_best\_prob = mean(as.numeric(best.pred==test\_data$is\_canceled))   
acc\_best\_prob

## [1] 0.7869168

test\_data$Yhat\_svm = attr(best.pred,'probabilities')[,1]   
head(test\_data)

## hotel is\_canceled lead\_time adults children babies meal  
## 2 Resort Hotel 0 737 2 0 0 BB  
## 9 Resort Hotel 1 85 2 0 0 BB  
## 28 Resort Hotel 1 60 2 0 0 BB  
## 31 Resort Hotel 0 118 1 0 0 BB  
## 34 Resort Hotel 0 69 2 0 0 BB  
## 35 Resort Hotel 1 45 3 0 0 BB  
## market\_segment distribution\_channel is\_repeated\_guest previous\_cancellations  
## 2 Direct Direct 0 0  
## 9 Online TA TA/TO 0 0  
## 28 Online TA TA/TO 0 0  
## 31 Direct Direct 0 0  
## 34 Offline TA/TO TA/TO 0 0  
## 35 Online TA TA/TO 0 0  
## previous\_bookings\_not\_canceled reserved\_room\_type deposit\_type  
## 2 0 C No Deposit  
## 9 0 A No Deposit  
## 28 0 E No Deposit  
## 31 0 A No Deposit  
## 34 0 A No Deposit  
## 35 0 D No Deposit  
## days\_in\_waiting\_list customer\_type adr required\_car\_parking\_spaces  
## 2 0 Transient 0.0 0  
## 9 0 Transient 82.0 0  
## 28 0 Transient 107.0 0  
## 31 0 Transient 62.0 0  
## 34 0 Transient 65.5 0  
## 35 0 Transient 108.8 0  
## Yhat\_svm  
## 2 0.004207927  
## 9 0.658008929  
## 28 0.615796712  
## 31 0.775526777  
## 34 0.779993735  
## 35 0.760787283

#test\_data$Yhat\_svm

attach(test\_data)  
  
#try cut\_off = 0.5  
class\_svm <- function(x){ifelse(Yhat\_svm > x, 1, 0)}  
predicted\_svm1 = class\_svm(0.5)  
CF1 <- confusionMatrix(factor(predicted\_svm1), is\_canceled, positive=as.character(1))  
CF1

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 410 2250  
## 1 7145 2134  
##   
## Accuracy : 0.2131   
## 95% CI : (0.2058, 0.2205)  
## No Information Rate : 0.6328   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.3718   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.48677   
## Specificity : 0.05427   
## Pos Pred Value : 0.22998   
## Neg Pred Value : 0.15414   
## Prevalence : 0.36720   
## Detection Rate : 0.17874   
## Detection Prevalence : 0.77720   
## Balanced Accuracy : 0.27052   
##   
## 'Positive' Class : 1   
##

CF1$table

## Reference  
## Prediction 0 1  
## 0 410 2250  
## 1 7145 2134

CF1$overall

## Accuracy Kappa AccuracyLower AccuracyUpper AccuracyNull   
## 0.2130832 -0.3718325 0.2057667 0.2205398 0.6328001   
## AccuracyPValue McnemarPValue   
## 1.0000000 0.0000000

CF1$byClass

## Sensitivity Specificity Pos Pred Value   
## 0.4867701 0.0542687 0.2299817   
## Neg Pred Value Precision Recall   
## 0.1541353 0.2299817 0.4867701   
## F1 Prevalence Detection Rate   
## 0.3123765 0.3671999 0.1787419   
## Detection Prevalence Balanced Accuracy   
## 0.7772008 0.2705194

# create a prediction object  
predict\_svm <- prediction(Yhat\_svm, is\_canceled)  
  
# accuracy curve  
acc\_svm = performance(predict\_svm,'acc')  
plot.new()  
plot(acc\_svm, col='cyan3', lwd=2)  
title("Accuracy curves")  
legend(0.7, 0.5 ,"SVM",   
 lty = c(1,1,1),   
 lwd = c(2,2,2),  
 col = "cyan3",  
 ncol=1, cex=0.9, y.intersp=1.2)  
  
#ROC Curve  
ROC\_svm <- performance(predict\_svm, "tpr", "fpr")  
plot.new()  
plot(ROC\_svm, add = TRUE, col= "cyan3")  
abline(0,1, col = "black")  
title("ROC curves")  
legend(0.7, 0.5 ,"SVM",   
 lty = c(1,1,1),   
 lwd = c(2,2,2),  
 col = "cyan3",  
 ncol=1, cex=0.9, y.intersp=1.2)  
  
#AUC value  
auc\_svm = performance(predict\_svm,"auc")  
auc\_svm@y.values

## [[1]]  
## [1] 0.1613585

#CRC Curve  
CRC\_svm <- performance(predict\_svm, "tpr", "rpp")  
plot.new()  
plot(CRC\_svm, add = TRUE, col= "cyan3")  
abline(0,1, col = "black")  
title("CRC curves")  
legend(0.7, 0.5 ,"SVM",   
 lty = c(1,1,1),   
 lwd = c(2,2,2),  
 col = "cyan3",  
 ncol=1, cex=0.9, y.intersp=1.2)  
  
#Lift Curves   
LIFT\_svm <- performance(predict\_svm, "lift", "rpp")  
plot.new()  
plot(LIFT\_svm, add = TRUE, col= "cyan3")  
abline(1,0, col = "black")  
title("Lift curves")  
legend(0.7, 0.5 ,"SVM",   
 lty = c(1,1,1),   
 lwd = c(2,2,2),  
 col = "cyan3",  
 ncol=1, cex=0.9, y.intersp=1.2)