Statistical Learning Assignment 3

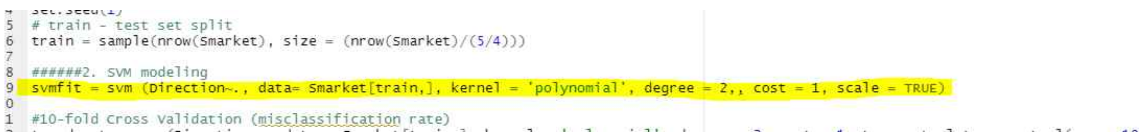
2020047029
JunYeong Ahn

1.write a paragraph (about 1/3 page) how SVM is similar to logistic regression (40%), and only how it is similar.

SVM and logistic regression have something in common in that the SVM's decision rule is based only on a potentially small subset of the training observations (the support vectors), which means that it is quite robust to the behavior of observations that are far away from the hyperplane. Similarly, logistic regression has very low sensitivity to observations far from the decision boundary due to its sigmoid curve shape.

This similarity of their actions also come from their loss function's similarity. L(X, y, β) is a loss function quantifying the extent to which the model, parametrized by β, fits the data (X, y). Since the loss function of SVM (so-called 'hinge loss') and that of logistic regression are similar to each other, SVM and logistic regression often tend to give very similar results.

2.make a SVM for the Smarket data with a polynomial kernel (degree=2) (40%)

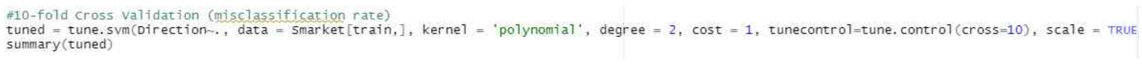


```
set.seed(1)
# train - test set split
train = sample(nrow(Smarket), size = (nrow(Smarket)/(5/4)))

######2. SVM modeling
svmfit = svm (Direction~., data= Smarket[train,], kernel = 'polynomial', degree = 2,, cost = 1, scale = TRUE)

#10-fold Cross Validation (misclassification rate)
```

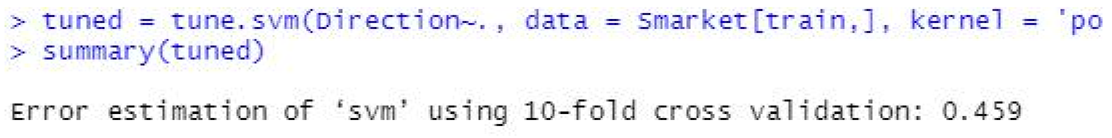
<SVM modeling>

- At first, split the data into train & test to draw Test Data ROC curve later on. Then just fit the SVM model with Smarket[train,] (train set) and call it 'svmfit'.

1, the value of the cost, is just randomly selected because our main task is not to identify the role of cost in modeling.



```
#10-fold Cross Validation (misclassification rate)
tuned = tune.svm(Direction~., data = Smarket[train,], kernel = 'polynomial', degree = 2, cost = 1, tunecontrol=tune.control(cross=10), scale = TRUE
summary(tuned)
```



```
> tuned = tune.svm(Direction~., data = Smarket[train,], kernel = 'po
> summary(tuned)

Error estimation of 'svm' using 10-fold cross validation: 0.459
```

<10-fold CV>

- For model evaluation I conducted 10-fold CV using tune.svm() function of e1071 package, who gives us error estimate of SVM (average misclassification rate). Because this is not the case for regression, calculating misclassification rate of each fold and averaging them is much more meaningful than something like calculating MSE.

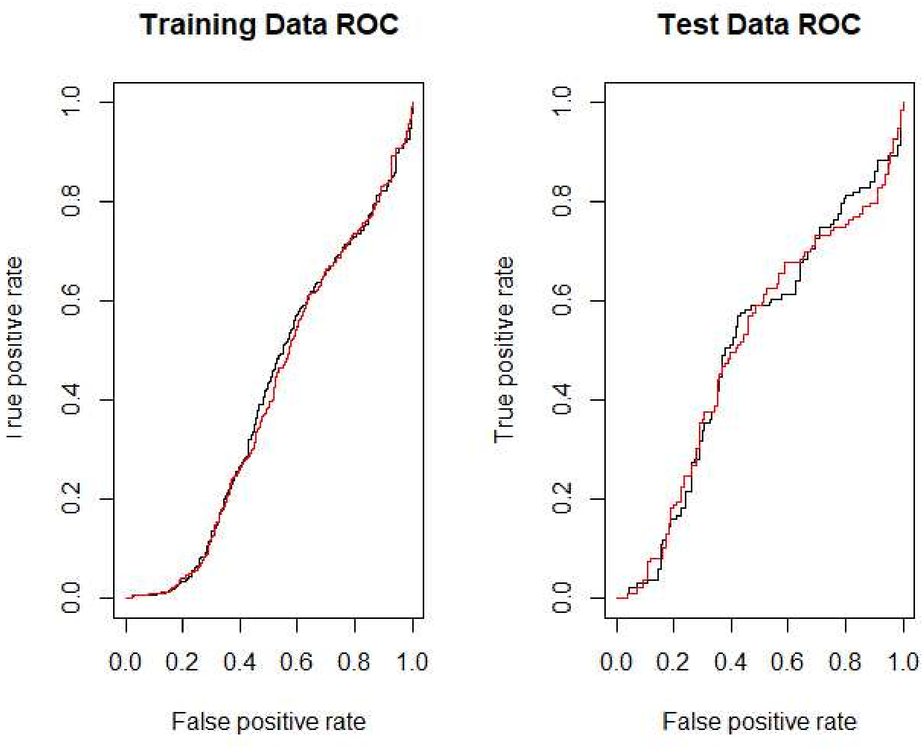
Anyway since there was no any optimization of the value of cost, this model tend to often misclassify the data with the figure of around 0.45.

3.and create a ROC curve using the model (20%)

-A short review

A receiver operating characteristic curve, or ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \qquad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$$



<On the left panel>

- Not that meaningful, because we trained our model with this data – this cannot give us better prediction on totally new data than one <On the right panel>.

Black line : Drawing ROC curve with training data, the cost = 1

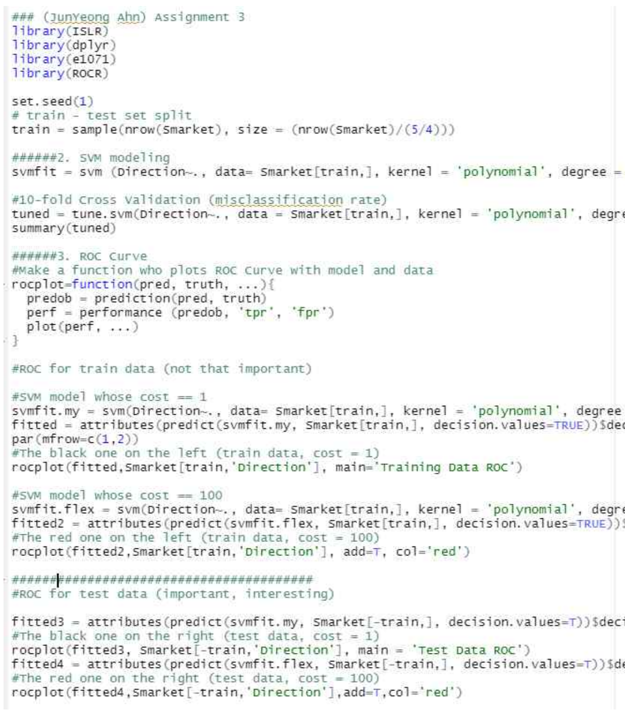
Red line (Just for visualizing that ROC differs when cost differs, not the main task) : “, the cost = 100

<On the right panel>

- Meaningful, because we are drawing ROC curve with completely new data (test set which I got earlier by splitting the given Smarket data) unlike on <On the left panel>.

Black line : Drawing ROC curve with test data, the cost = 1

Red line ("): ", the cost = 100



```
### (JunYeong Ahn) Assignment 3
library(ISLR)
library(dplyr)
library(e1071)
library(ROCR)

set.seed(1)
# train - test set split
train = sample(nrow(Smarket), size = (nrow(Smarket)/(5/4)))

######2. SVM modeling
svmfit = svm (Direction~., data= Smarket[train,], kernel = 'polynomial', degree = 2,, cost = 1, scale = TRUE)

#10-fold Cross Validation (misclassification rate)
tuned = tune.svm(Direction~., data = Smarket[train,], kernel = 'polynomial', degree = 2, cost = 1, tunecontrol=tune.control(cross=10), scale = TRUE)
summary(tuned)

######3. ROC Curve
#Make a function who plots ROC Curve with model and data
rocplot=function(pred, truth, ...){
  predob = prediction(pred, truth)
  perf = performance (predob, 'tpr', 'fpr')
  plot(perf, ...)
}

#ROC for train data (not that important)

#SVM model whose cost == 1
svmfit.my = svm(Direction~., data= Smarket[train,], kernel = 'polynomial', degree = 2, cost = 1, scale = TRUE, decision.values=TRUE)
fitted = attributes(predict(svmfit.my, Smarket[train,], decision.values=TRUE))$decision.values
par(mfrow=c(1,2))
#The black one on the left (train data, cost = 1)
rocplot(fitted,Smarket[train,'Direction'], main='Training Data ROC')

#SVM model whose cost == 100
svmfit.flex = svm(Direction~., data= Smarket[train,], kernel = 'polynomial', degree = 2, cost = 100, scale = TRUE, decision.values=TRUE)
fitted2 = attributes(predict(svmfit.flex, Smarket[train,], decision.values=TRUE))$decision.values
#The red one on the left (train data, cost = 100)
rocplot(fitted2,Smarket[train,'Direction'], add=T, col='red')

##########################################
#ROC for test data (important, interesting)

fitted3 = attributes(predict(svmfit.my, Smarket[-train,], decision.values=T))$decision.values
#The black one on the right (test data, cost = 1)
rocplot(fitted3, Smarket[-train,'Direction'], main = 'Test Data ROC')
fitted4 = attributes(predict(svmfit.flex, Smarket[-train,], decision.values=T))$decision.values
#The red one on the right (test data, cost = 100)
rocplot(fitted4,Smarket[-train,'Direction'],add=T,col='red')
```

<The entire codes>