SVM project

MGMT6770

9/23/2019

Table of Contents

# Introduction

This is an implementation of a support vector machine designed to identify short-term stock price changes, based on previous stock parameters: open, low, high, close, volume

## Stock selected for this exercise:

## get data:

crcData = read.csv('CRC.csv')  
attach(crcData)  
# set up data frame  
crcDataFrame = data.frame(open=price.open, high=price.high,  
 low=price.low, close=price.adjusted,volume= volume/10000,change = direction )  
head(crcDataFrame)

## open high low close volume change  
## 1 39.94 40.99 37.750 39.30 200.37 U  
## 2 39.84 40.29 37.900 38.50 136.20 D  
## 3 38.84 39.48 37.860 39.36 86.22 D  
## 4 38.79 38.96 36.500 37.51 137.20 D  
## 5 35.97 36.93 35.511 35.97 161.65 D  
## 6 36.48 36.93 34.690 35.10 117.78 D

tail(crcDataFrame)

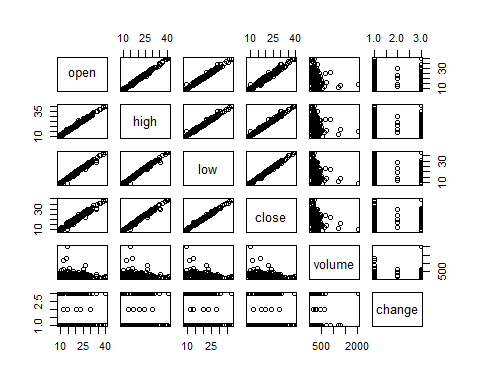
## open high low close volume change  
## 242 9.99 10.290 9.77 10.20 188.10 D  
## 243 10.27 10.660 9.66 9.75 213.30 D  
## 244 9.70 10.024 9.11 9.13 334.55 D  
## 245 9.02 9.290 8.40 9.07 380.58 D  
## 246 9.13 9.270 8.67 8.70 261.00 D  
## 247 8.80 9.020 8.47 8.51 338.23 D

remove(crcData)

levels(direction)

## [1] "D" "N" "U"

plot(crcDataFrame)



## classification model, categorical response

### traditional interface:

x <- subset(crcDataFrame, select = -change)  
y <- crcDataFrame$change  
model\_test <- svm(x, y)  
  
print(model\_test)

##   
## Call:  
## svm.default(x = x, y = y)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 227

summary(model\_test)

##   
## Call:  
## svm.default(x = x, y = y)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 227  
##   
## ( 106 116 5 )  
##   
##   
## Number of Classes: 3   
##   
## Levels:   
## D N U

## how well did the model fit?

pred <- predict(model\_test, x)  
  
# Check accuracy:  
table(pred, y)

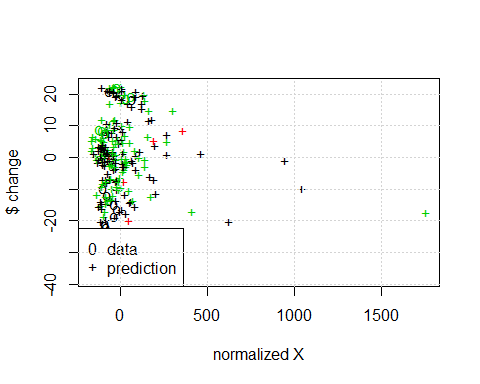
## y  
## pred D N U  
## D 96 4 57  
## N 0 0 0  
## U 36 1 53

## decision values

# compute decision values and probabilities:  
pred <- predict(model\_test, x, decision.values = TRUE)  
attr(pred, "decision.values")[1:6,]

## U/D U/N D/N  
## 1 -0.8641679 1.039344 1.0070841  
## 2 -0.9480064 1.047470 1.0067704  
## 3 -1.0001977 1.049312 1.0063641  
## 4 -0.9996962 1.050764 1.0050364  
## 5 -1.0584606 1.037416 0.9998011  
## 6 -1.0673171 1.053591 1.0030688

# visualize (classes by color, SV by crosses):  
plot(cmdscale(dist(crcDataFrame[,-6])),  
 col = as.integer(crcDataFrame[,6]),xlab="normalized X", ylab = "$ change",  
 pch = c("o","+")[1:200 %in% model\_test$index + 1])  
legend('bottomleft',c('data','prediction'),pch=c('0','+'))  
grid()



# train and test set:

pctTrain=0.66  
nObs = length(x[,1])  
nTrain = round(pctTrain\*nObs,0)  
TrainSetIndex = 1:nTrain  
  
  
  
# or ?  
#scramble=sample(1:nObs)  
#TrainSetIndex=scramble[1:nTrain]  
#TestSetIndex= scramble[(nTrain+1):nObs]

## set up training set, test set

# first the training set  
  
XtrainSet = crcDataFrame[TrainSetIndex,-6]  
YtrainSet = crcDataFrame$change[TrainSetIndex]

## get the model from training set..

model\_train<- svm(XtrainSet, YtrainSet)   
  
print(model\_train)

##   
## Call:  
## svm.default(x = XtrainSet, y = YtrainSet)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 146

summary(model\_train)

##   
## Call:  
## svm.default(x = XtrainSet, y = YtrainSet)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 146  
##   
## ( 67 75 4 )  
##   
##   
## Number of Classes: 3   
##   
## Levels:   
## D N U

## now the test set..

XtestSet = crcDataFrame[(nTrain:nObs),-6]  
YtestSet = crcDataFrame$change[nTrain:nObs]

## and evaluate with the test data

pred2 <- predict(model\_train, XtestSet)  
# And the accuracy..  
  
table(pred2, YtestSet)

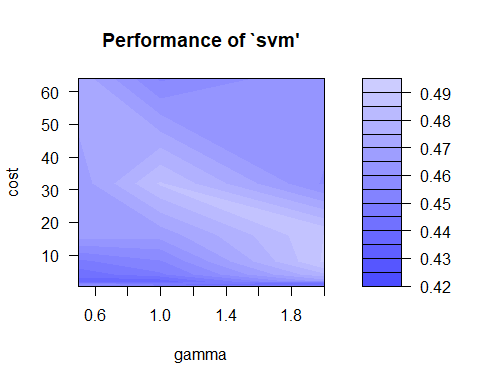
## YtestSet  
## pred2 D N U  
## D 18 1 13  
## N 0 0 0  
## U 29 0 24

## tune

stockTuned <- tune.svm(XtrainSet, YtrainSet, gamma = 2^(-1:1), cost = 2^(-1:6))  
  
summary(stockTuned)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## gamma cost  
## 2 1  
##   
## - best performance: 0.4242647   
##   
## - Detailed performance results:  
## gamma cost error dispersion  
## 1 0.5 0.5 0.4551471 0.07327981  
## 2 1.0 0.5 0.4496324 0.09573032  
## 3 2.0 0.5 0.4794118 0.06292388  
## 4 0.5 1.0 0.4613971 0.09877557  
## 5 1.0 1.0 0.4488971 0.10400723  
## 6 2.0 1.0 0.4242647 0.09652943  
## 7 0.5 2.0 0.4367647 0.11481476  
## 8 1.0 2.0 0.4367647 0.11769556  
## 9 2.0 2.0 0.4669118 0.11707120  
## 10 0.5 4.0 0.4419118 0.11240892  
## 11 1.0 4.0 0.4488971 0.11912433  
## 12 2.0 4.0 0.4794118 0.13059609  
## 13 0.5 8.0 0.4488971 0.11159971  
## 14 1.0 8.0 0.4544118 0.07976426  
## 15 2.0 8.0 0.4908088 0.12079940  
## 16 0.5 16.0 0.4669118 0.08634272  
## 17 1.0 16.0 0.4669118 0.13106444  
## 18 2.0 16.0 0.4900735 0.12467579  
## 19 0.5 32.0 0.4669118 0.09430702  
## 20 1.0 32.0 0.4856618 0.14050393  
## 21 2.0 32.0 0.4591912 0.13679951  
## 22 0.5 64.0 0.4735294 0.09145803  
## 23 1.0 64.0 0.4540441 0.14490094  
## 24 2.0 64.0 0.4647059 0.14137505

plot(stockTuned)



#use optimized parameters...   
model\_tuned <- svm(XtrainSet, YtrainSet, gamma=1, cost=1)   
  
print(model\_tuned)

##   
## Call:  
## svm.default(x = XtrainSet, y = YtrainSet, gamma = 1, cost = 1)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 147

summary(model\_tuned)

##   
## Call:  
## svm.default(x = XtrainSet, y = YtrainSet, gamma = 1, cost = 1)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 147  
##   
## ( 71 72 4 )  
##   
##   
## Number of Classes: 3   
##   
## Levels:   
## D N U

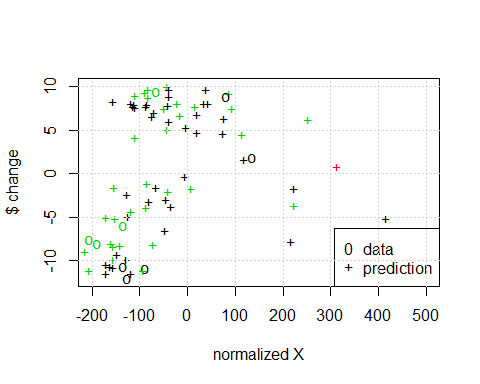
pred3 <- predict(model\_tuned, XtestSet)  
# And the accuracy..  
  
table(pred3, YtestSet)

## YtestSet  
## pred3 D N U  
## D 27 1 16  
## N 0 0 0  
## U 20 0 21

pred= predict(model\_tuned,XtestSet,decision.values = TRUE)  
attr(pred,"decision.values")[1:20,]

## U/D U/N D/N  
## 163 0.8587836 1.0053988 0.9951678  
## 164 0.8658319 1.0064848 1.0000285  
## 165 0.7343034 0.9986608 1.0020145  
## 166 0.8928790 1.0048584 0.9984906  
## 167 0.9562486 1.0121734 1.0008609  
## 168 0.8802783 1.0101437 1.0006149  
## 169 0.6081699 0.9868500 1.0009645  
## 170 0.8803002 1.0058591 1.0108113  
## 171 1.1560227 1.0179686 0.9951153  
## 172 0.8888819 0.9956298 0.9915024  
## 173 0.9966875 1.0032428 0.9864362  
## 174 1.2155280 1.0121818 0.9919761  
## 175 0.9266169 0.9952128 1.0182264  
## 176 0.8169514 1.0103904 1.0196638  
## 177 0.9199182 1.0008983 1.0218916  
## 178 0.6779104 0.9946399 0.9804695  
## 179 0.4571286 0.9968859 0.9804163  
## 180 1.1628840 1.0075606 0.9749657  
## 181 1.0054894 1.0021645 1.0070414  
## 182 0.7718305 0.9900439 1.0125545

plot(cmdscale(dist(XtestSet)),  
 col = as.integer(YtestSet),xlab="normalized X", ylab = "$ change",  
 pch = c("o","+")[1:200 %in% model\_tuned$index + 1],xlim = c(-200,500))  
legend('bottomright',c('data','prediction'),pch=c('0','+'))  
grid()



#SVM Assignment

## complete an rmarkdown file on the stock you selected.

You will turn in the URL for your github accout that has the repo for this assignment.

### Identify the stock you use for this assignment.

Why did you choose this particular stock?

I chose the California Resources Stock becaused fo the variability from day to day. It has also seen pretty steep rises and falls over the last few years.

### Use this template to get the SVM:

1. Training and Test sets:

I chose to use chronological data observations. I believe that because the stock I chose is of an energy company, that some trends could be market related or seasonal. I used 2/3 of my data set to train the SVM. I chose something large enough to include steep climbs and drops.

1. How well did the SVM do?

table(pred3,YtestSet)

## YtestSet  
## pred3 D N U  
## D 27 1 16  
## N 0 0 0  
## U 20 0 21

1. Tune the algorithm- i.e. find optimal gamma and cost values.

### Evaluate SVM performance

1. Did you exmine using other than the ‘open’, ‘low’, ‘high’, ‘close’ prices and volume as predictors?
2. Use the SVM you developed to predict stock moves on 16 - 20 Sept. 2019. Compute the ROI assuming the SVM identifies an opportunity, (for example use around $1000 on each of the SVM predictions)
3. Would you put your own $$ in this SVM?