# SVM Method to Classify the Kddcup99 Data

## Process of the Data:

* The last column is the label, we treat them as two part. One is” normal “and the other is” attack”
* The second, third and fourth column are characters. We use the following code to transfer these character into numbers and treat the last column as factor

for (i in 1:41){

kddcup\_train[,i]=as.numeric(kddcup\_train[,i])

kddcup\_test[,i]=as.numeric(kddcup\_test[,i])

}

* Finally we find that some columns are constant. We use the following command to remove these columns:

kddcup\_train1=subset(kddcup\_train,select=c(x.7,x.21,x.9,x.11,x.15,x.22,x.20,x.14,x.18))

kddcup\_test1=subset(kddcup\_test,select=-c(x.7,x.21,x.9,x.11,x.15,x.22,x.20,x.14,x.18))

## Implement the Data to find the optimal cost

Sample the data with size 5000 and 2000 respectively. 5000 data is the training data and 2000 data is the testing data. Run the algorithm 5 times with the cost parameter ranging from 1 to one hundred.

The figure of error\_rate vs cost parameter is in next page

## Train with larger size training data (20000) and larger size test data(5000)

model\_svm=svm(kddcup\_train2$Y~.,method="class",data=kddcup\_train2,cost=1)

start\_time=proc.time()

pre\_svm2=predict(model\_svm,newdata=kddcup\_test2,type="class")

end\_time=proc.time()

run\_time=end\_time-start\_time

error\_rate=sum(kddcup\_test[,42]!=pre\_svm2)/nrow(kddcup\_test)

The result are in the following:

> error\_rate

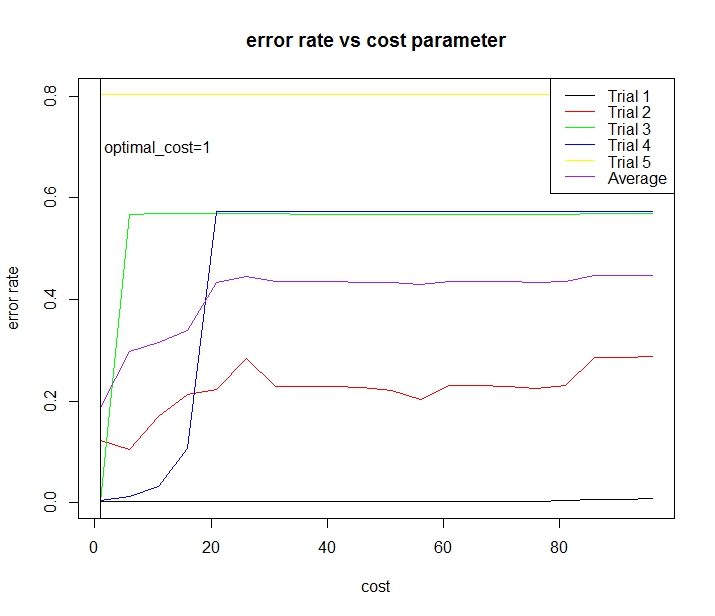
[1] 0.0042

> run\_time

user system elapsed

0.19 0.00 0.19

|  |  |
| --- | --- |
| Average time of one data stream(s) | Error\_rate |
| e(-5) | 0.0042 |



## Code Appendix

# SVM method to classify kddcup99 data

library("e1071")

kddcup <- read.csv("C:/Users/Christina/Desktop/kddcup.data\_10\_percent\_corrected", header=FALSE)

kddcup=as.matrix(kddcup)

error\_rate=matrix(0,nrow=5,ncol=20)

#we run the algorithm 5 times with the cost parameter changing from 1 to 100

for(n in (1:5)){

kddcup\_1=sample((nrow(kddcup)),size=5000,replace = FALSE, prob = NULL)

kddcup\_2=sample((nrow(kddcup)),size=2000,replace = FALSE, prob = NULL)

kddcup\_train=kddcup[kddcup\_1,]

kddcup\_test=kddcup[kddcup\_2,]

for (i in 1:2000){

if (kddcup\_test[i,42]=="normal."){

kddcup\_test[i,42]=0}

else{

kddcup\_test[i,42]=1

}

}

for (i in 1:5000){

if (kddcup\_train[i,42]=="normal."){

kddcup\_train[i,42]=0}

else{

kddcup\_train[i,42]=1

}

}

kddcup\_train=as.data.frame(kddcup\_train)

kddcup\_test=as.data.frame(kddcup\_test)

colnames(kddcup\_train)=c(paste("x.",1:41,sep=""),"Y")

colnames(kddcup\_test)=c(paste("x.",1:41,sep=""),"Y")

#process the data in order to apply the svm method

for (i in 1:41){

kddcup\_train[,i]=as.numeric(kddcup\_train[,i])

kddcup\_test[,i]=as.numeric(kddcup\_test[,i])

}

kddcup\_test[,42]=as.factor(kddcup\_test[,42])

kddcup\_train[,42]=as.factor(kddcup\_train[,42])

kddcup\_train1=subset(kddcup\_train,select=-c(x.7,x.21,x.9,x.11,x.15,x.22,x.20,x.14,x.18))

kddcup\_test1=subset(kddcup\_test,select=-c(x.7,x.21,x.9,x.11,x.15,x.22,x.20,x.14,x.18))

for (j in 1:20)

{#we choose cost from one to one hundred

m=seq(1,100,by=5)

c=m[j]

model\_svm=svm(kddcup\_train$Y~.,method="class",data=kddcup\_train1,cost=c)

pre\_svm=predict(model\_svm,newdata=kddcup\_test1,type="class")

error\_rate[n,j]=sum(kddcup\_test[,42]!=pre\_svm)/nrow(kddcup\_test)}

}

#plot the result to find

mean\_error\_rate=as.matrix(colMeans(error\_rate))

cost.optimal=apply(mean\_error\_rate,2,which.min)

m=as.matrix(m)

matplot(m,cbind(t(error\_rate),mean\_error\_rate),type='l',col=c('black','red','green','blue','yellow','purple'),ylab='error rate',xlab='cost',lty =1,cex=2)

title(main='error rate vs cost parameter')

legend(legend=c('Trial 1','Trial 2','Trial 3','Trial 4','Trial 5','Average'),col=c('black','red','green','blue','yellow','purple'),'topright',lty=1)

abline(v=cost.optimal)

text(x=11,y=0.7,label='optimal\_cost=1')

# we find the optimal cost is 1.The we apply this to large training set and testing set( 20000 and 5000)

kddcup\_1=sample((nrow(kddcup)),size=20000,replace = FALSE, prob = NULL)

kddcup\_2=sample((nrow(kddcup)),size=5000,replace = FALSE, prob = NULL)

kddcup\_train=kddcup[kddcup\_1,]

kddcup\_test=kddcup[kddcup\_2,]

for (i in 1:5000){

if (kddcup\_test[i,42]=="normal."){

kddcup\_test[i,42]=0}

else{

kddcup\_test[i,42]=1

}

}

for (i in 1:20000){

if (kddcup\_train[i,42]=="normal."){

kddcup\_train[i,42]=0}

else{

kddcup\_train[i,42]=1

}

}

kddcup\_train=as.data.frame(kddcup\_train)

kddcup\_test=as.data.frame(kddcup\_test)

colnames(kddcup\_train)=c(paste("x.",1:41,sep=""),"Y")

colnames(kddcup\_test)=c(paste("x.",1:41,sep=""),"Y")

#process the data in order to apply the svm method

for (i in 1:41){

kddcup\_train[,i]=as.numeric(kddcup\_train[,i])

kddcup\_test[,i]=as.numeric(kddcup\_test[,i])

}

kddcup\_test[,42]=as.factor(kddcup\_test[,42])

kddcup\_train[,42]=as.factor(kddcup\_train[,42])

kddcup\_train2=subset(kddcup\_train,select=-c(x.7,x.21,x.9,x.11,x.15,x.22,x.20,x.14,x.18))

kddcup\_test2=subset(kddcup\_test,select=-c(x.7,x.21,x.9,x.11,x.15,x.22,x.20,x.14,x.18))

model\_svm=svm(kddcup\_train2$Y~.,method="class",data=kddcup\_train2,cost=1)

start\_time=proc.time()

pre\_svm2=predict(model\_svm,newdata=kddcup\_test2,type="class")

end\_time=proc.time()

run\_time=end\_time-start\_time

error\_rate=sum(kddcup\_test[,42]!=pre\_svm2)/nrow(kddcup\_test)

> error\_rate

[1] 0.0042

> run\_time

user system elapsed

0.19 0.00 0.19