HW4_SVM_xz2735

Xiaofan Zhang(xz2735)

3/23/2019

Problem 1

1.

$d^2(x,x') = ||x-x'||_2^2 =< x-x', x-x' >= \sqrt{(x-x')^T(x-x')} =< x,x > -2 < x,x' > + < x',x' >$

2.

Given $K(x,x') =< \phi(x), \phi(x') >$, the distance is $d_k^2(x,x') = ||\phi(x)-\phi(x')||_2^2 =< \phi(x)-\phi(x'), \phi(x)-\phi(x') >= \sqrt{(\phi(x)-\phi(x'))^T(\phi(x)-\phi(x'))} =< \phi(x),\phi(x) > -2 < \phi(x),\phi(x') > + < \phi(x'),\phi(x') >= K(x,x)-2K(x,x')+K(x',x')$

3.

It calculates the kernel distance between point x and x'.

Problem 2

1.

```r
#Load data
library(leaps)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.2.1 --
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  2.0.1     v dplyr   0.7.7
## v tidyr   0.8.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
## Warning: package 'tibble' was built under R version 3.5.2
## Warning: package 'stringr' was built under R version 3.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
credit = read.csv("Credit.csv")
```

```r
#Define Dummy variables.
credit$Gender = ifelse(credit$Gender == "Female",1,0) %>% as.factor
credit$Student = ifelse(credit$Student == "Yes",1,0) %>% as.factor
credit$African.American = ifelse(credit$Ethnicity == "African American",1,0) %>% as.factor
credit$Asian = ifelse(credit$Ethnicity == "Asian",1,0)%>% as.factor
credit$Married = ifelse(credit$Married == "Yes",1,0)%>% as.factor
credit = subset(credit,select = -c(X,Ethnicity))
head(credit)
```

```
##    Income Limit Rating Cards Age Education Gender Student Married Balance
## 1  14.891  3606    283     2  34        11      0       0       1     333
## 2 106.025  6645    483     3  82        15      1       1       1     903
## 3 104.593  7075    514     4  71        11      0       0       0     580
## 4 148.924  9504    681     3  36        11      1       0       0     964
## 5  55.882  4897    357     2  68        16      0       0       1     331
## 6  80.180  8047    569     4  77        10      0       0       0    1151
##   African.American Asian
## 1                0     0
## 2                0     1
## 3                0     1
## 4                0     1
## 5                0     0
## 6                0     0
```

Best subset selection & Forward stepwise selection & Backward stepwise selection

```
fit.best = regsubsets(Balance~.,nbest = 1, data = credit, method="exhaustive")
fit.fwd = regsubsets(Balance~.,data = credit,method="forward")
fit.bwd = regsubsets(Balance~.,data = credit, method = "backward")
sum.best = summary(fit.best)
sum.fwd = summary(fit.fwd)
sum.bwd = summary(fit.bwd)
```

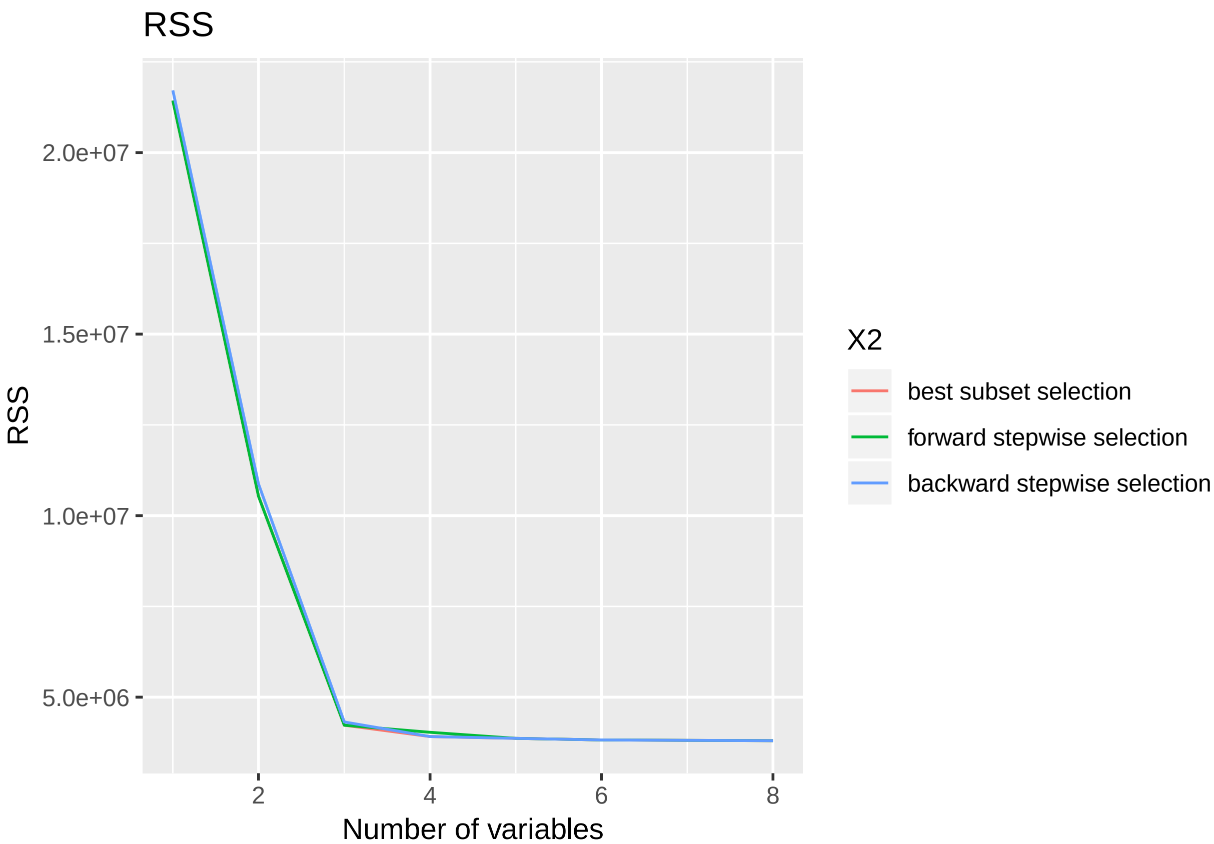
```
library(ggplot2)
library(reshape)
```

```
##
## Attaching package: 'reshape'

## The following object is masked from 'package:dplyr':
##
##     rename

## The following objects are masked from 'package:tidyr':
##
##     expand, smiths
```

```
rss = cbind(sum.best$rss,sum.fwd$rss,sum.bwd$rss) %>% melt
rss$X2 = recode(factor(rss$X2),'1'="best subset selection",'2'="forward stepwise selection",'3'="backwa
ggplot(data=rss, aes(x=X1,y=value,color=X2))+
  geom_line()+
  ggtitle("RSS")+
  xlab("Number of variables")+
  ylab("RSS")
```



2.

Best subset selection

```r
bic.min.best = which.min(sum.best$bic)
cp.min.best = which.min(sum.best$cp)
print(bic.min.best )
```

```
## [1] 4
```

```r
print(cp.min.best)
```

```
## [1] 6
```

```r
sum.best
```

```
## Subset selection object
## Call: regsubsets.formula(Balance ~ ., nbest = 1, data = credit, method = "exhaustive")
## 11 Variables  (and intercept)
##                    Forced in Forced out
## Income                 FALSE      FALSE
## Limit                  FALSE      FALSE
## Rating                 FALSE      FALSE
## Cards                  FALSE      FALSE
## Age                    FALSE      FALSE
## Education              FALSE      FALSE
## Gender1                FALSE      FALSE
## Student1               FALSE      FALSE
```

```
## Married1                FALSE      FALSE
## African.American1       FALSE      FALSE
## Asian1                  FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          Income Limit Rating Cards Age Education Gender1 Student1 Married1
## 1  ( 1 ) " "    " "   "*"    " "   " " " "       " "     " "      " "
## 2  ( 1 ) "*"    " "   "*"    " "   " " " "       " "     " "      " "
## 3  ( 1 ) "*"    " "   "*"    " "   " " " "       " "     "*"      " "
## 4  ( 1 ) "*"    "*"   " "    "*"   " " " "       " "     "*"      " "
## 5  ( 1 ) "*"    "*"   "*"    "*"   " " " "       " "     "*"      " "
## 6  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       " "     "*"      " "
## 7  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       "*"     "*"      " "
## 8  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       "*"     "*"      " "
##          African.American1 Asian1
## 1  ( 1 ) " "               " "
## 2  ( 1 ) " "               " "
## 3  ( 1 ) " "               " "
## 4  ( 1 ) " "               " "
## 5  ( 1 ) " "               " "
## 6  ( 1 ) " "               " "
## 7  ( 1 ) " "               " "
## 8  ( 1 ) "*"               " "
```

So for BIC, the optimal best subset model has 4 variables such that income, limit, cards, student. So for cp, the optimal best subset model has 6 variables such that income, limit, rating, cards, age student.

Forward stepwise selection

```
bic.min.fwd = which.min(sum.fwd$bic)
cp.min.fwd = which.min(sum.fwd$cp)
bic.min.fwd
```

```
## [1] 5
```

```
cp.min.fwd
```

```
## [1] 6
```

```
sum.fwd
```

```
## Subset selection object
## Call: regsubsets.formula(Balance ~ ., data = credit, method = "forward")
## 11 Variables  (and intercept)
##                     Forced in Forced out
## Income                  FALSE      FALSE
## Limit                   FALSE      FALSE
## Rating                  FALSE      FALSE
## Cards                   FALSE      FALSE
## Age                     FALSE      FALSE
## Education               FALSE      FALSE
## Gender1                 FALSE      FALSE
## Student1                FALSE      FALSE
## Married1                FALSE      FALSE
## African.American1       FALSE      FALSE
## Asian1                  FALSE      FALSE
```

```
## 1 subsets of each size up to 8
## Selection Algorithm: forward
##          Income Limit Rating Cards Age Education Gender1 Student1 Married1
## 1  ( 1 ) " "    " "   "*"    " "   " " " "       " "     " "      " "
## 2  ( 1 ) "*"    " "   "*"    " "   " " " "       " "     " "      " "
## 3  ( 1 ) "*"    " "   "*"    " "   " " " "       " "     "*"      " "
## 4  ( 1 ) "*"    "*"   "*"    " "   " " " "       " "     "*"      " "
## 5  ( 1 ) "*"    "*"   "*"    "*"   " " " "       " "     "*"      " "
## 6  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       " "     "*"      " "
## 7  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       "*"     "*"      " "
## 8  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       "*"     "*"      " "
##          African.American1 Asian1
## 1  ( 1 ) " "               " "
## 2  ( 1 ) " "               " "
## 3  ( 1 ) " "               " "
## 4  ( 1 ) " "               " "
## 5  ( 1 ) " "               " "
## 6  ( 1 ) " "               " "
## 7  ( 1 ) " "               " "
## 8  ( 1 ) "*"               " "
```

So for BIC, the optimal forward stepwise model has 5 variables such that income, limit, rating, cards, student. So for cp, the optimal forward stepwise model has 6 variables such that income, limit, rating, cards, age student.

Backward stepwise selection

```
bic.min.bwd = which.min(sum.bwd$bic)
cp.min.bwd = which.min(sum.bwd$cp)
bic.min.bwd
```

```
## [1] 4
```

```
cp.min.bwd
```

```
## [1] 6
```

```
sum.bwd
```

```
## Subset selection object
## Call: regsubsets.formula(Balance ~ ., data = credit, method = "backward")
## 11 Variables  (and intercept)
##                   Forced in Forced out
## Income                FALSE      FALSE
## Limit                 FALSE      FALSE
## Rating                FALSE      FALSE
## Cards                 FALSE      FALSE
## Age                   FALSE      FALSE
## Education             FALSE      FALSE
## Gender1               FALSE      FALSE
## Student1              FALSE      FALSE
## Married1              FALSE      FALSE
## African.American1     FALSE      FALSE
## Asian1                FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: backward
```

```
##          Income Limit Rating Cards Age Education Gender1 Student1 Married1
## 1  ( 1 ) " "    "*"   " "    " "   " " " "       " "     " "      " "
## 2  ( 1 ) "*"    "*"   " "    " "   " " " "       " "     " "      " "
## 3  ( 1 ) "*"    "*"   " "    " "   " " " "       " "     "*"      " "
## 4  ( 1 ) "*"    "*"   " "    "*"   " " " "       " "     "*"      " "
## 5  ( 1 ) "*"    "*"   "*"    "*"   " " " "       " "     "*"      " "
## 6  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       " "     "*"      " "
## 7  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       "*"     "*"      " "
## 8  ( 1 ) "*"    "*"   "*"    "*"   "*" " "       "*"     "*"      " "
##          African.American1 Asian1
## 1  ( 1 ) " "               " "
## 2  ( 1 ) " "               " "
## 3  ( 1 ) " "               " "
## 4  ( 1 ) " "               " "
## 5  ( 1 ) " "               " "
## 6  ( 1 ) " "               " "
## 7  ( 1 ) " "               " "
## 8  ( 1 ) "*"               " "
```

So for BIC, the optimal backward stepwise model has 4 variables such that income, limit, cards, student. So for cp, the optimal backward stepwise model has 6 variables such that income, limit, rating, cards, age student.

Problem 3

1.

Load data and split the dataset

```r
#Load data
df1 = read.csv("train.5-1.txt")
colnames(df1) = 1:256
df2 = read.csv("train.6.txt")
colnames(df2) = 1:256
df = rbind(df1,df2)
y = as.factor(c(rep(-1, dim(df1)[1]),rep(1,dim(df2)[1])))

#Spilt data into train and test set
set.seed(123)
index = 1:nrow(df)
index = sample(index,size = floor(0.2*nrow(df)),replace = FALSE)
x.train = df[-index,]
y.train = y[-index]
x.test = df[index,]
y.test = y[index]
train = cbind(x.train,y.train)
```

(a)

```r
#linear SVM with soft margin
library(e1071)
```

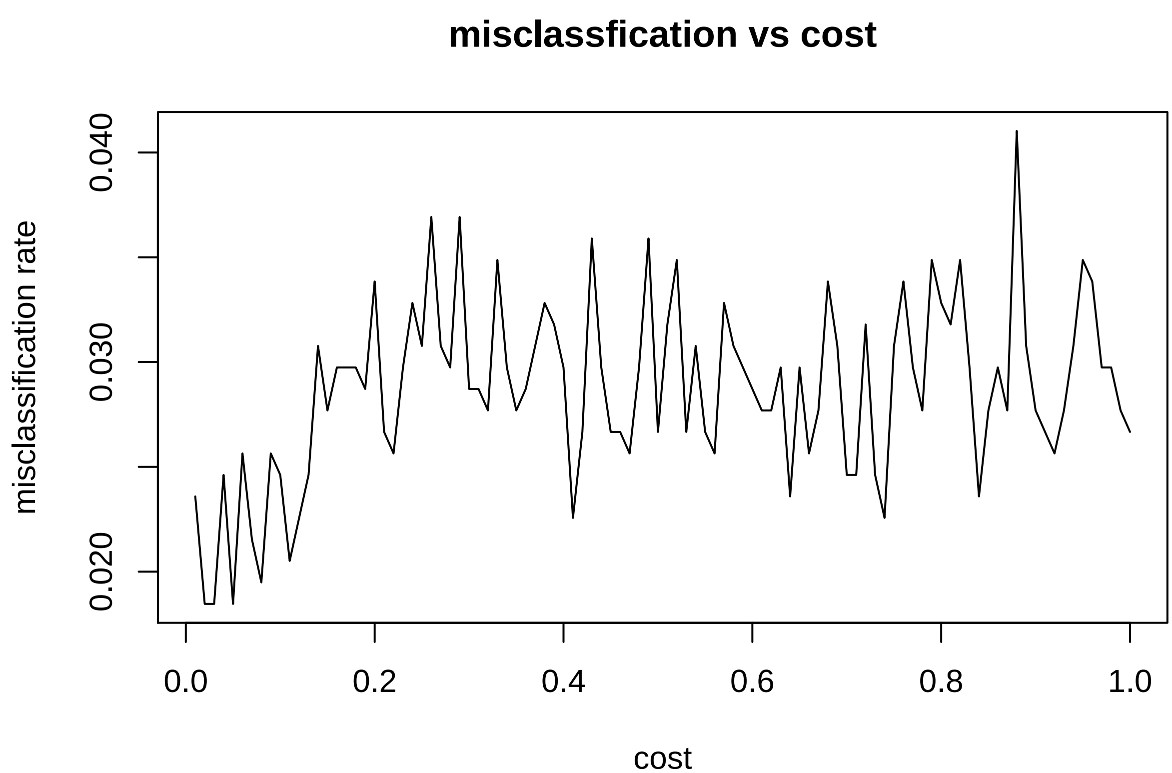
```
## Warning: package 'e1071' was built under R version 3.5.2
```

```r
library(rpart)
cost = seq(0.01,1,0.01)
j=1
mis.rate=NULL
#Cross valiate andtuning parameter.
for(i in cost){
  svm.fit <- svm(y.train ~., train, type = "C-classification", kernel = "linear", cost= i,cross =    10
  mis.rate[j] = (100- svm.fit$tot.accuracy)/100
  j = j+1
}
optimal.cost = cost[order(min(mis.rate))]
svm.linear =  svm(y.train ~., train, type = "C-classification", kernel = "linear", cost= optimal.cost ,
print(optimal.cost)
```

```
## [1] 0.01
```

plot cost and misclassification rate

```r
plot(cost,mis.rate,type="l",main="misclassfication vs cost",ylab = "misclassification rate")
```




```r
#Compute test error of linear svm
y_pred1 = predict(svm.linear,x.test)
test.error1 = mean(y.test!=y_pred1)
test.error1
```

```
## [1] 0.02469136
```

The best cost is 0.01.

(b)

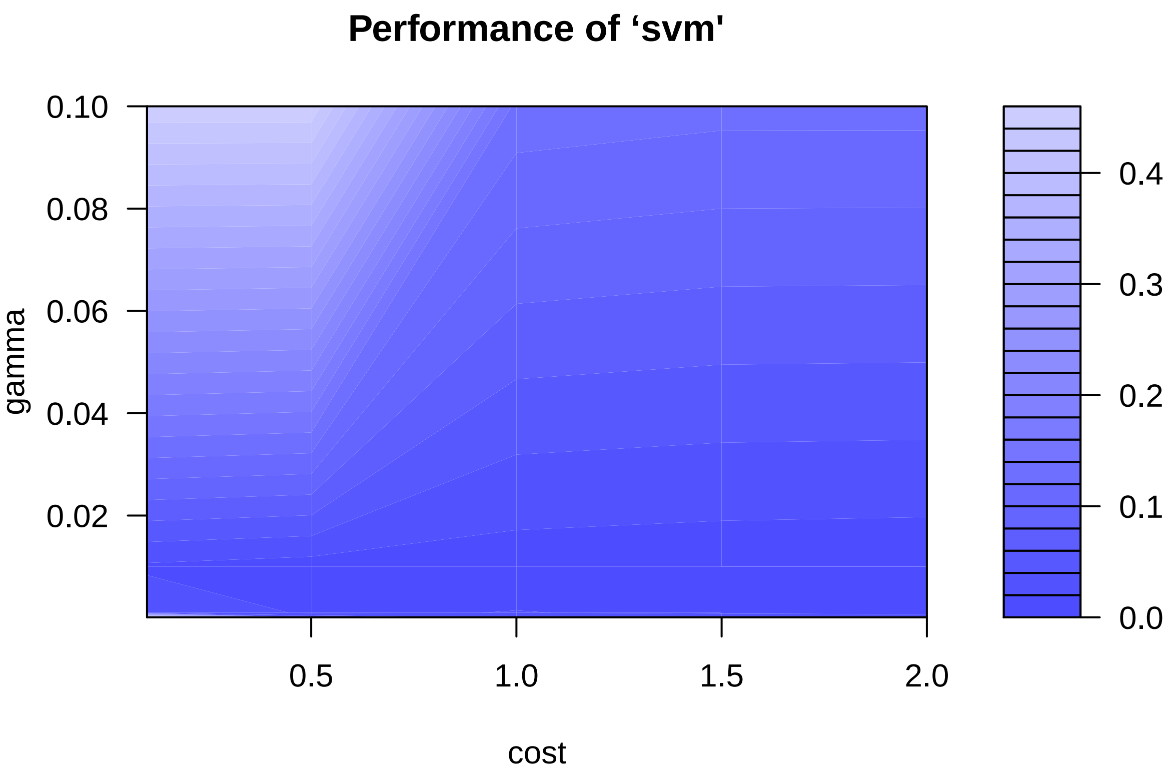
train rbf svm and tuning parameters such as cost and gamma(binwidth).

```
set.seed(1)
tune.out.rbf <- tune(svm, y.train ~., data=train,
                 kernel='radial',
                 ranges = list(cost=c(0.1,0.5,1,1.5,2),
                 gamma=c(0.0001,0.001,0.01,0.1)))
tune.out.rbf
```

The optimal cost is 2, optimal gamma is 0.01.

plot heatmap about cost, gamma and misclassification rate

```
plot(tune.out.rbf)
```




```
#Compute test error of rbf svm
y_pred2 = predict(tune.out.rbf$best.model,x.test)
test.error2 = mean(y_pred2 != y.test)
test.error2
```

```
## [1] 0.008230453
```

summary of both models

```
test.error = data.frame(linear.svm = test.error1, rbf.svm = test.error2)
test.error
```

```
##   linear.svm     rbf.svm
## 1 0.02469136 0.008230453
```

According to the misclassification rate, test error of linear svm is 0.02469136, whereas test error of rbf svm is 0.008230453 rbf svm is better.