SDSC3006 Lab
9-SVM and PCA

Langming LIU langmiliu2-c@my.cityu.edu.hk

School of Data Science
City University of Hong Kong

Contents

- Suport Vector Machine
- Principal Components Regression
- Principal Components Analysis

Suport Vector Machine

Implemention

Notice:
1. SVM is an extension of the support vector classifier, using different kernels(non-linear).
2. Use the e1071 library to demonstrate the SVM on a two-dimensional example.

```
library(e1071)
##Generate data with nonlinear boundary
set.seed(1)
x=matrix(rnorm(200*2),ncol=2)
x[1:100,]=x[1:100,]+2
x[101:150,]=x[101:150,]-2
y=c(rep(1,150),rep(2,50))
dat=data.frame(x=x,y=as.factor(y))
plot(x,col=(3-y))
```

Implemention

```
##Use SVM with a radial kernel
train=sample(200,100)
svmfit=svm(y~.,data=dat[train,],kernel="radial",gamma=1,cost=1)
##"cost" is similar to tuning parameter C, but with opposite
##effects: small "cost", wide margin; large "cost", narrow margin

plot(svmfit,dat[train,])
summary(svmfit)
```

Implemention

```
##Select best values for "gamma" and "cost" by CV
tune.out=tune(svm,y~.,data=dat[train,],kernel="radial",ranges
=list(cost=c(0.1,1,10,100,1000),gamma=c(0.5,1,2,3,4)))
summary(tune.out)

##Use SVM with a polynomial kernel
svmfit=svm(y~.,data=dat[train,],kernel="polynomial"
, degree=2,ranges=list(cost=c(0.001,0.01,0.1,1,5,10,100)))
plot(svmfit,dat[train,])
```

Principal Components Regression

Implemention

Notice:

1. PCR can be performed using pcr() fuction in pls library
2. We now apply PCR to the Hitters data, in order to predict Salary.

```
install.packages('pls')
library(ISLR)
library(pls)
attach(Hitters)
Hitters = na.omit (Hitters)              ##remove rows with missing values
set.seed (2)
pcr.fit = pcr(Salary ~ ., data = Hitters , scale = TRUE, validation = "CV")
##10-fold cross-validation
summary (pcr.fit)
##plot the CV scores
validationplot (pcr.fit , val.type = "MSEP")
```

Implemention

```
##create training set and test set
x=model.matrix(Salary~.,Hitters)[,-1]
y=Hitters$Salary
train=sample(1:nrow(x), nrow(x)/2)
test=(-train)
y.test=y[test]
##perform PCR on training set
set.seed(1)
pcr.fit = pcr(Salary ~ ., data = Hitters , subset = train , scale = TRUE , validation
= "CV")
validationplot (pcr.fit , val.type = "MSEP")      ##find the best M
##compute test MSE
pcr.pred = predict(pcr.fit , x[test , ], ncomp = 5)
mean ((pcr.pred - y.test)^2)              ##compare with shrinkage method?
##perform PCR on full data
pcr.fit = pcr (y ~ x, scale = TRUE , ncomp = 5)
```

Principal Components Analysis

Implemention

Notice:

1. We perform PCA on the USArrests data set, which is part of the base R package.
2. Using the prcomp() function, which is one of several functions in R that perform PCA.
3. By default, the prcomp() function centers the variables to have mean zero. 'Option scale = TRUE' scale the variables to have sd 1.

Implemention

```
names (USArrests)
pr.out = prcomp (USArrests , scale = TRUE)
names (pr.out)
##rotation: principal component loadings
pr.out$rotation
biplot (pr.out , scale = 0)
##sdev: standard deviation of principal components
pr.out$sdev
pr.var = pr.out$sdev^2
```

Implemention

##compute the proportion of variance explained by each principal component
pve = pr.var / sum (pr.var)
par (mfrow = c(1, 2))
plot (pve , xlab = " Principal Component ", ylab = " Proportion of Variance Explained ", ylim = c(0, 1), type = "b")
plot (cumsum (pve), xlab = " Principal Component ", ylab = " Cumulative Proportion of Variance Explained ", ylim = c(0, 1), type = "b")