SVM

Xinying Fang

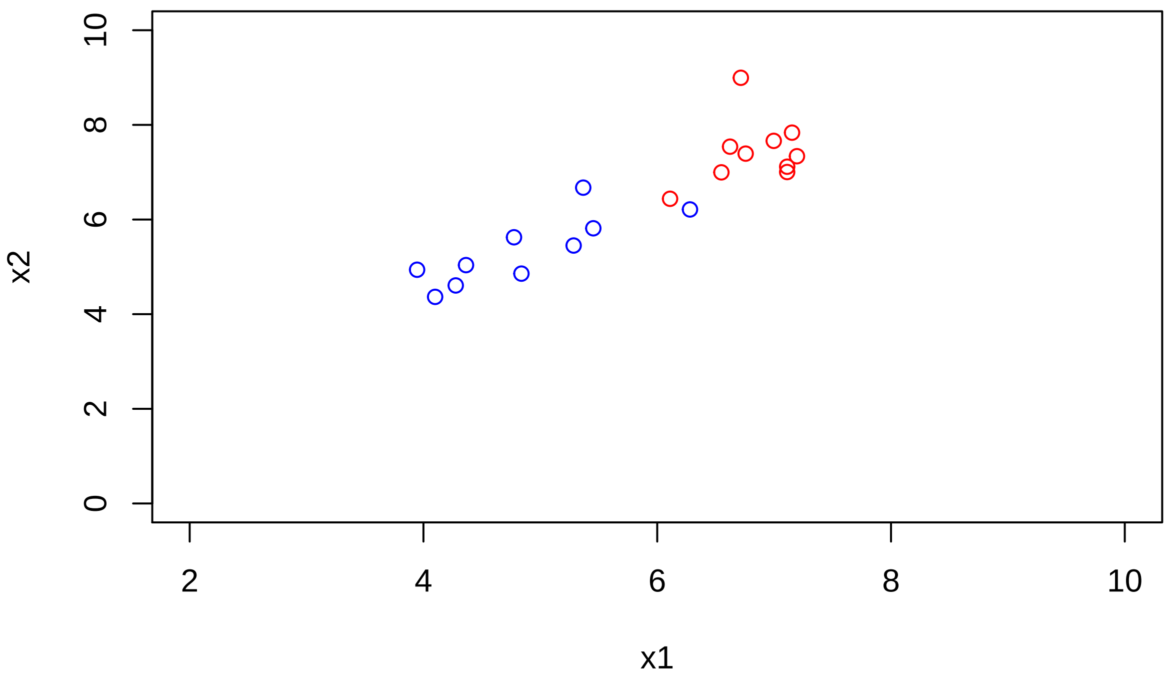
11/29/2021

Data simulation

Here, we simulate two groups of data that have overlap so we are going to derive a soft margin for svm.

```
library(mvtnorm)
set.seed(100)
n=10
X = rmvnorm(n, mean=c(7,7), sigma=matrix(c(1,0.5,0.5,1), ncol = 2))
x1 = X[,1]
x2 = X[,2]
X = rmvnorm(n, mean=c(5,5), sigma=matrix(c(1,0.5,0.5,1), ncol = 2))
x11 = X[,1]
x22 = X[,2]

plot(x1,x2,col="red",xlim=c(2,10),ylim=c(0,10))+ points(x11,x22,col="blue")
```




```
## integer(0)
```

Following the instruction of **quadprog** package, we perform the quadratic optimization by specifying required matrices and vectors:

```
library(quadprog)
library(Matrix)
```

```r
X <- matrix(c(x1,x11,x2,x22),ncol=2)
Y <- matrix(rep(c(1,-1),each=n), ncol=1)
X <- scale(X)

Ymat <- Y %*% t(Y)
Xmat <- X %*% t(X)

Dmat <- Ymat * Xmat
Dmatnear <- nearPD(Dmat)

# dvec       <- t(matrix(1,2*n,1))
dvec <- as.vector(matrix(1,2*n,1))
firstrow <- t(Y)
diagmat <- diag(x = 2*n)
diagmat2 <- -diag(x = 2*n)
Amat_t <- rbind(firstrow, diagmat, diagmat2)
Amat <- t(Amat_t)
cval <- 10
# bvec <- t(as.matrix(c(rep(0,(2*n+1)),rep(-cval,(2*n)))))
bvec <- c(rep(0,(2*n+1)),rep(-cval,(2*n)))
res <- solve.QP(Dmat=Dmatnear$mat,dvec=dvec,Amat=Amat,bvec=bvec, meq=1)

alpha <- res$solution
betas <- t(as.matrix(alpha*Y)) %*% (X);
ind<- which(alpha>1e-5)
# mean(Y[ind] - t(as.matrix(alpha[ind]*Y[ind])) %*% (X[ind,]) %*% t(X[ind,]))

beta0<- mean(Y[ind]-X[ind,]%*%t(betas))

betas;beta0
```

```
##          [,1]     [,2]
## [1,] 1.33078 2.482698

## [1] -1.372601
```

Comparing the results with svm function in **e1071** package:

```r
## compared with SVM function
library(e1071)
svm.model <- svm(as.factor(Y) ~ X, cost = 10, kernel="linear",scale=FALSE)
beta = drop(t(svm.model$coefs)%*%X[svm.model$index,])
beta0.svm = coef(svm.model)[1]
beta;beta0.svm
```

```
## [1] 1.330786 2.482656

## (Intercept)
##  -0.9877366
```

Our coefficient estimates are pretty close to the svm outputs, but the intercept is different. However, the difference is reasonable. By plotting the two svm results out, red line represent the results from **e1071** package, black lines represent the results from the quadratic optimization implemented by myself. Both separation includes one mis-classification. Their results are pretty similar.

```r
library(ggplot2)
dt <- data.frame(X1 = X[,1],
```

```r
                      X2 = X[,2],
                      Y = Y)
ggplot(dt, aes(x=X1, y=X2)) +
  geom_point(aes(color=as.factor(Y)))+
  geom_abline(slope = -betas[1]/betas[2], intercept =-beta0/betas[2])+
  geom_abline(slope = -betas[1]/betas[2], intercept =(1-beta0)/betas[2], linetype="twodash")+
  geom_abline(slope = -betas[1]/betas[2], intercept =(-1-beta0)/betas[2], linetype="twodash") +
  geom_abline(slope= -beta[1]/beta[2], intercept = -beta0.svm/beta[2], color='red')+
  geom_abline(slope= -beta[1]/beta[2], intercept = (1-beta0.svm)/beta[2], color='red',linetype="twodash
  geom_abline(slope= -beta[1]/beta[2], intercept = (-1-beta0.svm)/beta[2], color='red',linetype="twodas
```