Math 5365

Data Mining 1

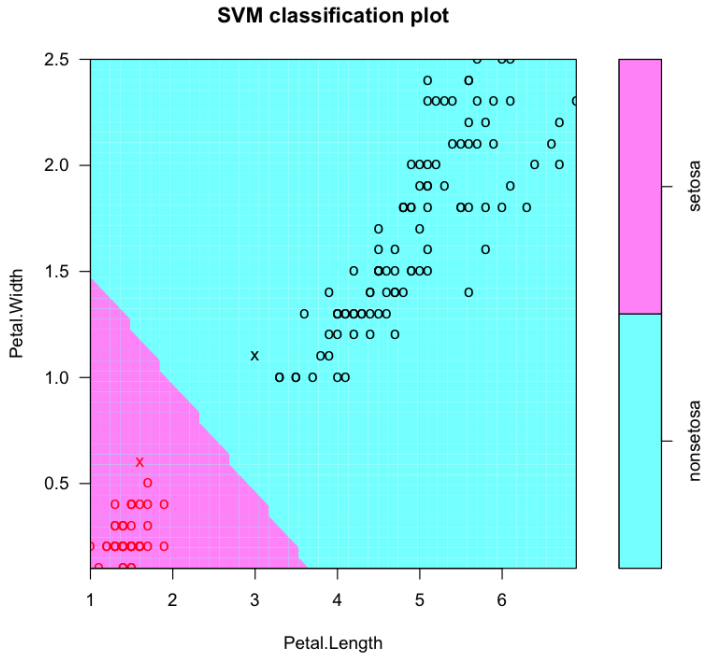
Homework 12

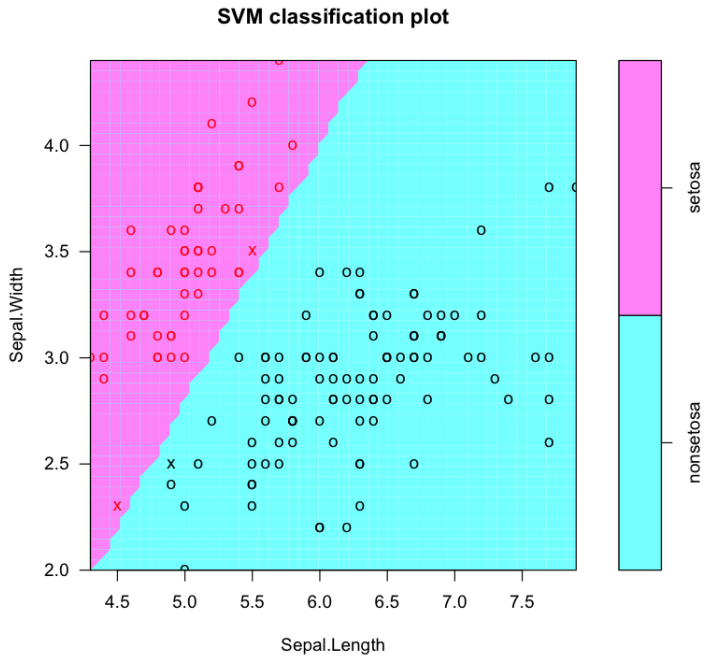
Mary Barker

1. (a) Create a version of the iris data set, where the class labels "versicolor" and "virginica" are replaced by "nonsetosa". This problem involves building an SVM to classify iris flowers as setosa or nonsetosa

```
new_iris <- iris
idx <- (new_iris$Species == 'setosa') * 1
Species <- rep('',length(idx))
Species[idx == 1] <- 'setosa'
Species[idx != 1] <- 'nonsetosa'
new_iris$Species <- Species
levels(new_iris$Species) <- c('setosa', 'nonsetosa')
```

   (b) Fit a linear support vector machine to the data with cost = 1000 and plot the SVM





(c) 1c Is the data set linearly seperable?

The data is linearly separable in each case.

(d) How many support vectors are there?

For the petal model, there are 2 support vectors. For the sepal model, there are 3.

(e) Fnd the parameters w and b that define the decision boundary

For the petal model, w = $-1.497422, -1.238551$, b = -1.804382.

For the petal model, w = $-7.095041, 3.112485$, b = -5.102501.

2. Split `wdbc.data` into 70% training and 30% test data

```
wdbc <- read.table("~/Dropbox/Tarleton/data_mining/dfiles/wdbc.data",
        header = FALSE, sep = ",")
wdbc <- wdbc[,-1]
splitset <- splitdata(wdbc, 0.7, FALSE)
train <- splitset$train
```

(a) Fit an SVM to the training data.

```
wdbcmodel <- svm(V2~., wdbc[train,])
```

(b) What type of kernel was used?

radial

(c) Find the classification accuracy of this SVM on the training and test data

test accuracy: 97.07602%

train accuracy: 98.49246%

The confusionmatrix for the train accuracy is

	B	M
B	240	1
M	5	152

(d) Use the tune.svm command to tune the values of cost and γ. It may take some experimentation to find suitable ranges for these parameters.

```
ogamma <- wdbcmodel$gamma
ocost <- wdbcmodel$cost
tunewdbc <- tune.svm(V2~., data=wdbc[train,],
                     gamma=10^(-4:1), cost=10^(-1:2))

ngamma <- (tunewdbc$best.parameters)[1]
ncost <- (tunewdbc$best.parameters)[2]
```

The original values for γ and cost were $0.0\bar{3}$ and 1 respectively. After tuning, the best values were 0.01 and 1 respectively.

(e) Refit the SVM using the tuned cost and gamma values

```
wdbcmodel2 <- svm(V2~., data=wdbc[train,],gamma=ngamma,cost=ncost)
predwdbc2 <- predict(wdbcmodel2, newdata = wdbc[-train,])
confmatrix(wdbc$V2[-train], predwdbc2)
```

(f) Find the classification accuracy of the tuned SVM on the training and test data

test accuracy: 95.32164%

train accuracy: 98.49246%

The confusion matrix for the train accuracy is

	B	M
B	241	0
M	6	151

At a glance, it can be seen that the accuracy for the predicted values on the test set actually decreased with the tuned values. This is most likely due to overfitting, since the training set did not show decreased accuracy.

3. Create a data set similar to the one below, where there are four normally distributed

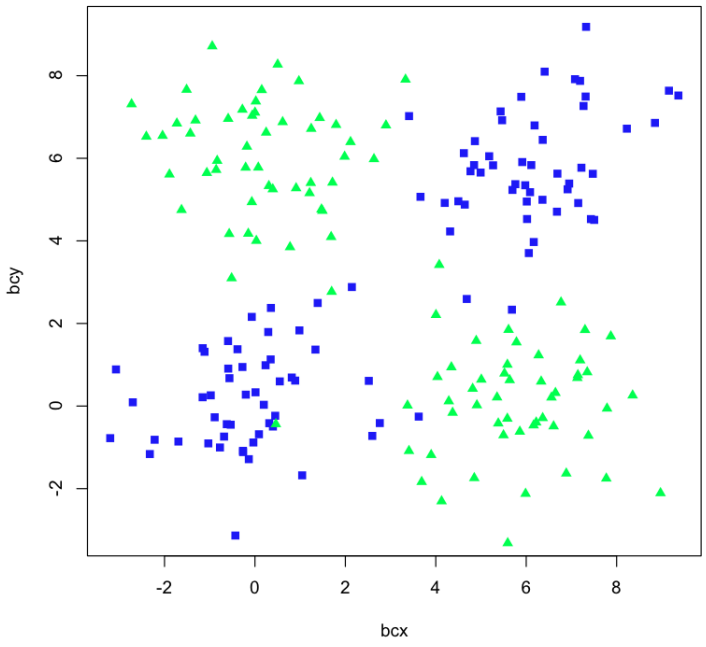
clusters, each containing 50 points, centered at

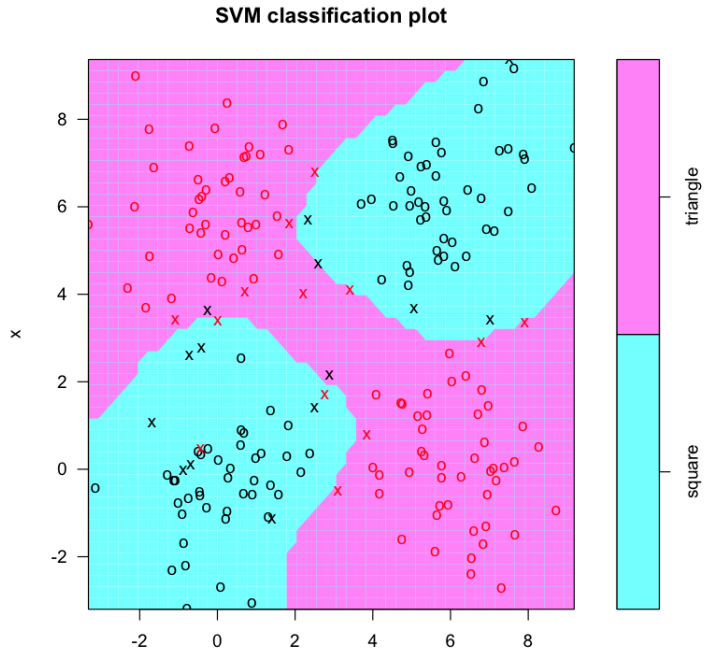
$$(0,0),(0,6),(6,0),(6,6)$$

for all four clusters, $\sigma_x = \sigma_y = 1.5$.

```
bcx <- rnorm(50, 0, 1.5)
bcy <- rnorm(50, 0, 1.5)
bcx <- c(bcx, rnorm(50, 6, 1.5))
bcy <- c(bcy, rnorm(50, 6, 1.5))

rtx <- rnorm(50, 6, 1.5)
rty <- rnorm(50, 0, 1.5)
rtx <- c(rtx, rnorm(50, 0, 1.5))
rty <- c(rty, rnorm(50, 6, 1.5))

x <- c(bcx, rtx)
y <- c(bcy, rty)
type <- rep('square', 100)
type <- c(type, rep('triangle', 100))

plot(bcx, bcy, type='p',col='blue', pch = 15)
lines(rtx, rty, type='p',col='green', pch=17)

points <- data.frame(x = x, y = y, type = type)
```

4. Create an SVM for distinguishing between the blue and green nodes, plot the SVM, and calculate its classification accuracy.





The classification accuracy is 97%

The confusion matrix is

	square	*triangle*
square	97	3
triangle	3	97

```
#Data Mining Hw 12
library(e1071)
#1a. Create a version of the iris data set, where the class labels
#    "versicolor" and "virginica" are replaced by "nonsetosa". This
#    problem involves building an SVM to classify iris flowers as
#    setosa or nonsetosa

new_iris <- iris
idx <- (new_iris$Species == 'setosa') * 1
Species <- rep('',length(idx))
Species[idx == 1] <- 'setosa'
Species[idx != 1] <- 'nonsetosa'
new_iris$Species <- Species
levels(new_iris$Species) <- c('setosa', 'nonsetosa')

# 1b Fit a linear support vector machine to the data with cost = 1000
#    and plot the SVM

#Separate model for comparison
petalmodel <- svm(as.factor(Species)~Petal.Length + Petal.Width,
                  new_iris, kernel='linear',cost=1000)
plot(petalmodel, new_iris, Petal.Width~Petal.Length, )
```

```
sepalmodel <- svm(as.factor(Species)~Sepal.Length + Sepal.Width,
                  new_iris, kernel='linear',cost=1000)
plot(sepalmodel, new_iris, Sepal.Width~Sepal.Length, )

speciesmodel <- svm(as.factor(Species)~., new_iris, kernel='linear',cost = 1000)
plot(speciesmodel, new_iris, Petal.Width~Petal.Length)

# 1c Is the data set linearly seperable?

# 1d How many support vectors are there?
sum(speciesmodel$nSV)

#1e Fnd the parameters w and b that define the decision boundary
w = t(speciesmodel$coefs) %*% (speciesmodel$SV)
b = -speciesmodel$rho

#2 Split wdbc.data into 70% training and 30% test data

wdbc <- read.table("~/Dropbox/Tarleton/data_mining/dfiles/wdbc.data",
        header = FALSE, sep = ",")
wdbc <- wdbc[,-1]
splitset <- splitdata(wdbc, 0.7, FALSE)
train <- splitset$train

#2a Fit an SVM to the training data.

wdbcmodel <- svm(V2~., wdbc[train,])
```

```
#2b What type of kernel was used?

#2c Find the classification accuracy of this SVM on the training and test data
predwdbc <- predict(wdbcmodel, newdata = wdbc[-train,])
predwdbc <- predict(wdbcmodel, newdata = wdbc[train,])
confmatrix(wdbc$V2[train], predwdbc)

#2d Use the tune.svm command to tune the values of cost and gamma. It may
#   take some experimentation to find suitable ranges for these parameters.
ogamma <- wdbcmodel$gamma
ocost <- wdbcmodel$cost
tunewdbc <- tune.svm(V2~., data=wdbc[train,], gamma=10^(-4:1), cost=10^(-1:2))

ngamma <- (tunewdbc$best.parameters)[1]
ncost <- (tunewdbc$best.parameters)[2]

#2e Refit the SVM using the tuned cost and gamma values
wdbcmodel2 <- svm(V2~., data=wdbc[train,],gamma=ngamma,cost=ncost)
predwdbc2 <- predict(wdbcmodel2, newdata = wdbc[-train,])
confmatrix(wdbc$V2[-train], predwdbc2)

#2f Find the classification accuracy of the tuned SVM on the training
#   and test data


#3a Create a data set similar to the one below, where there are four
```

```
#   normally distributed clusters, each containing 50 points, centered at
#                     (0, 0), (0, 6), (6, 0), (6, 6)
#   for all four clusters, sigma_x = sigma_y = 1.5.
bcx <- rnorm(50, 0, 1.5)
bcy <- rnorm(50, 0, 1.5)
bcx <- c(bcx, rnorm(50, 6, 1.5))
bcy <- c(bcy, rnorm(50, 6, 1.5))

rtx <- rnorm(50, 6, 1.5)
rty <- rnorm(50, 0, 1.5)
rtx <- c(rtx, rnorm(50, 0, 1.5))
rty <- c(rty, rnorm(50, 6, 1.5))

x <- c(bcx, rtx)
y <- c(bcy, rty)
type <- rep('square', 100)
type <- c(type, rep('triangle', 100))

plot(bcx, bcy, type='p',col='blue', pch = 15)
lines(rtx, rty, type='p',col='green', pch=17)

points <- data.frame(x = x, y = y, type = type)
#3b Create an SVM for distinguishing between the blue and green nodes, plot the
#   SVM, and calculate its classification accuracy.
pointmodel <- svm(as.factor(type)~., points, cost=1000)
plot(pointmodel, points)
predpoint <- predict(pointmodel, newdata = points)
```

```
confmatrix(points$type, predpoint)
```