STAD68 A1 Q3

Yang Guan Jian Guo

September 28, 2016

Part 1

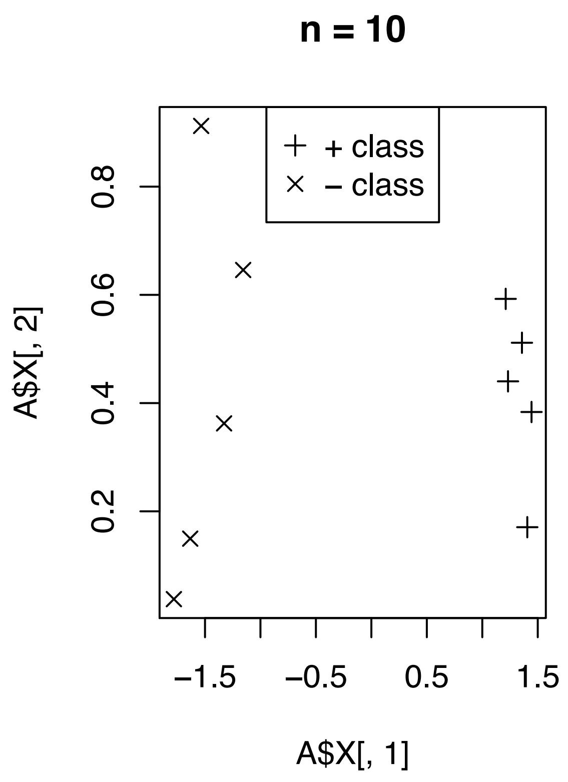
Write code to create a synthetic ("made up") data set that is linearly separable. In particular, write a procedure that accepts two argument: d the number of dimensions and n the number of examples. Your code should then output the labeled data. You can choose the format, but a natural one would be a $n \times d$-matrix for the d-dimensional data x and a n-vector y for the label.

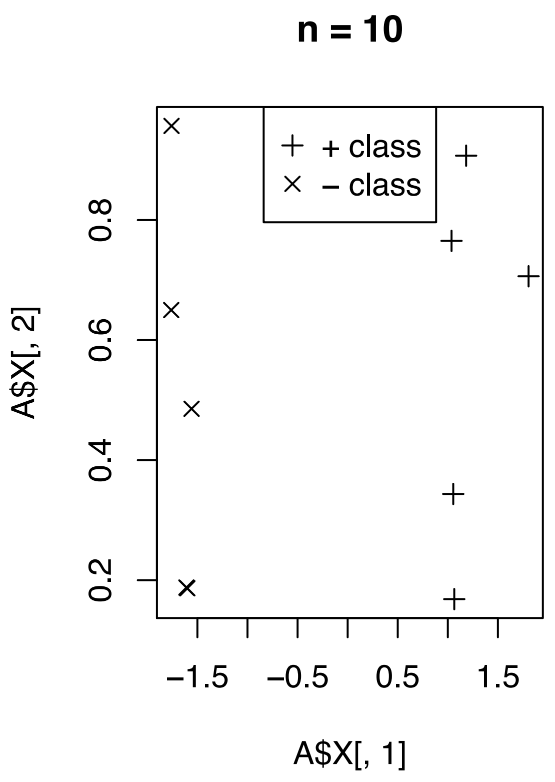
```
gen_ls <- function(n,d){
  A = matrix(runif(n*d,0,1),n,d);
  A[,1] = A[,1] - c(rep(-1,floor(n/2)), rep(2,n-floor(n/2) ));
  list(X= A,y= sign(A[,1]))
  }
```

Part 2

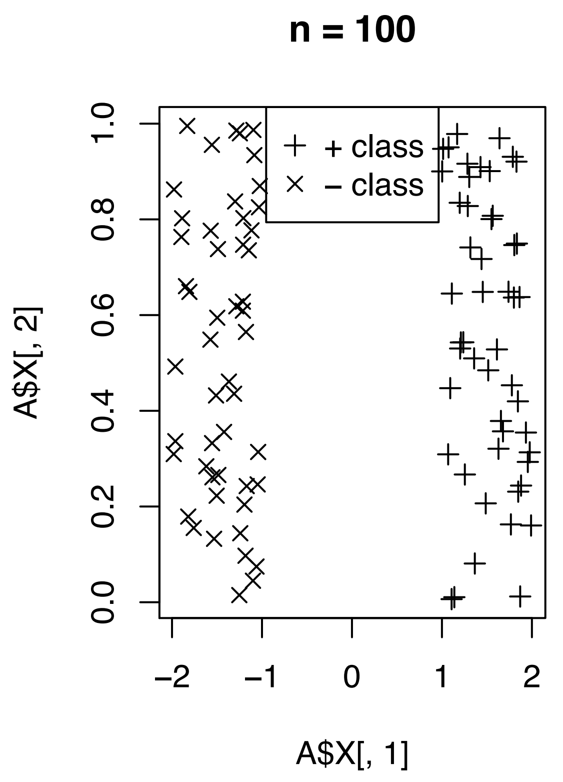
Visualize the output of your procedure in 2 dimensions and $n \in \{10, 100\}$ data points using a scatter plot with + symbols for positively labeled examples and $-$ symbols otherwise. Show two random data sets for each setting.

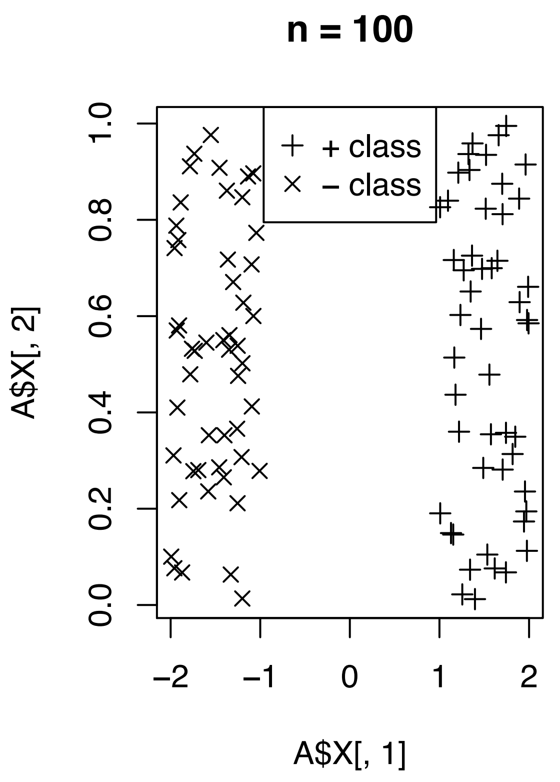
```
par(mfrow = c(1,2))
n = 10;d = 2;
A = gen_ls(n,d)
plot(A$X[,1],A$X[,2],
     pch = mapply((function(x){ if (x == 1) 3 else 4 }),A$y),main = "n = 10")
legend("top",legend = c("+ class","- class"),pch = c(3,4))
A = gen_ls(n,d)
plot(A$X[,1],A$X[,2],
     pch = mapply((function(x){ if (x == 1) 3 else 4 }),A$y),main = "n = 10")
legend("top",legend = c("+ class","- class"),pch = c(3,4))
```





```
n = 100
A = gen_ls(n,d)
plot(A$X[,1],A$X[,2],
     pch = mapply((function(x){ if (x == 1) 3 else 4 }),A$y),main = "n = 100")
legend("top",legend = c("+ class","- class"),pch = c(3,4))
A = gen_ls(n,d)
plot(A$X[,1],A$X[,2],
     pch = mapply((function(x){ if (x == 1) 3 else 4 }),A$y),main = "n = 100")
legend("top",legend = c("+ class","- class"),pch = c(3,4))
```





Part 3

Write code that takes as input 1) a linear classifier v_H and 2) n observations $x_1, \ldots, x_n$ (e.g., in the form of a matrix), and returns the vector of classifications.

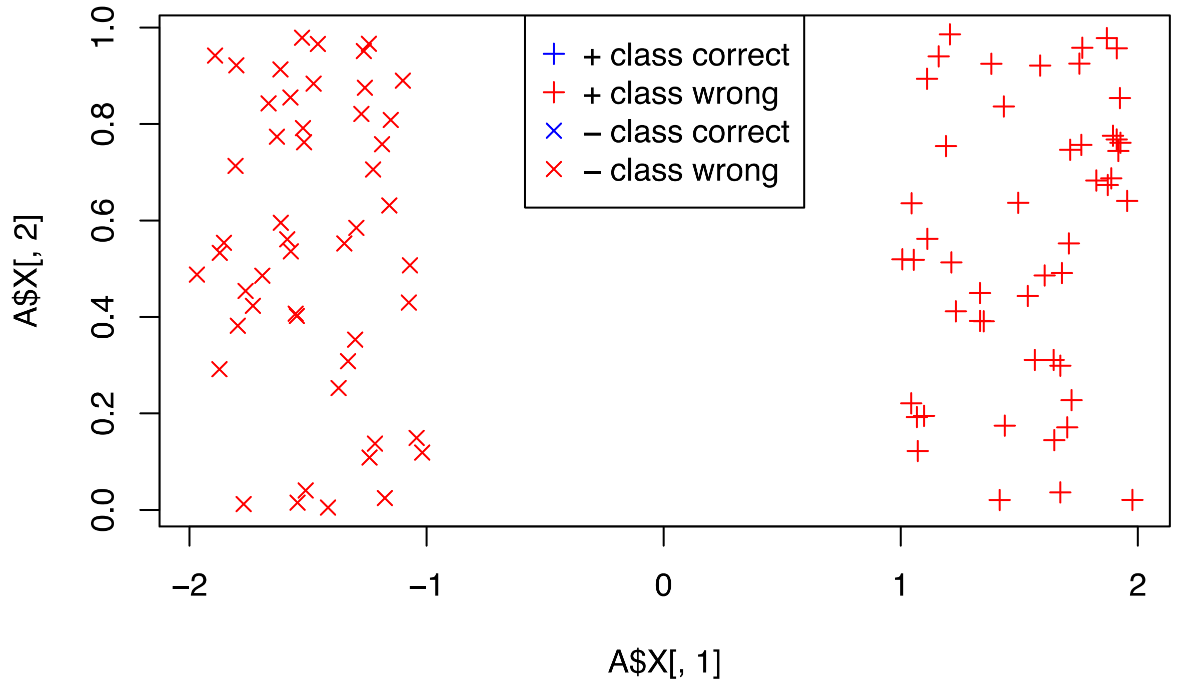
```
classify <- function(X,v){
  if (length(v) == ncol(X))
  sign( X%*%v )
  else
    sign(cbind(rep(1,nrow(A$X)),A$X) %*%v)
}
```

Part 4

Visualize the output of your procedure again on a randomly generated 2D data set and the classifier $v_H = (0\ 0)^T$. Make classifications (+/−) that are correct be BLUE and those that are incorrect, RED.

```
par(mfrow = c(1,1))
v = c(0,0)
A = gen_ls(n,d)
c = classify(A$X,v)

plot(A$X[,1],A$X[,2]
     ,col =mapply((function(x){ if (x == T) "blue" else "red" }),c)
     , pch= mapply((function(x){ if (x == 1) 3 else 4 }),A$y) )
legend("top",
       legend = c("+ class correct","+ class wrong","- class correct","- class wrong"),
       pch = c(3,3,4,4),col = c("blue","red"))
```



Part 5

Write code to take a dataset and produce a random training-test data split. In particular, write code that takes as input 1) a number k and 2) a labelled data set with at least $n > k$ observations, and outputs a random split of the dataset into two halves: k labeled training and $n - k$ labeled test data.

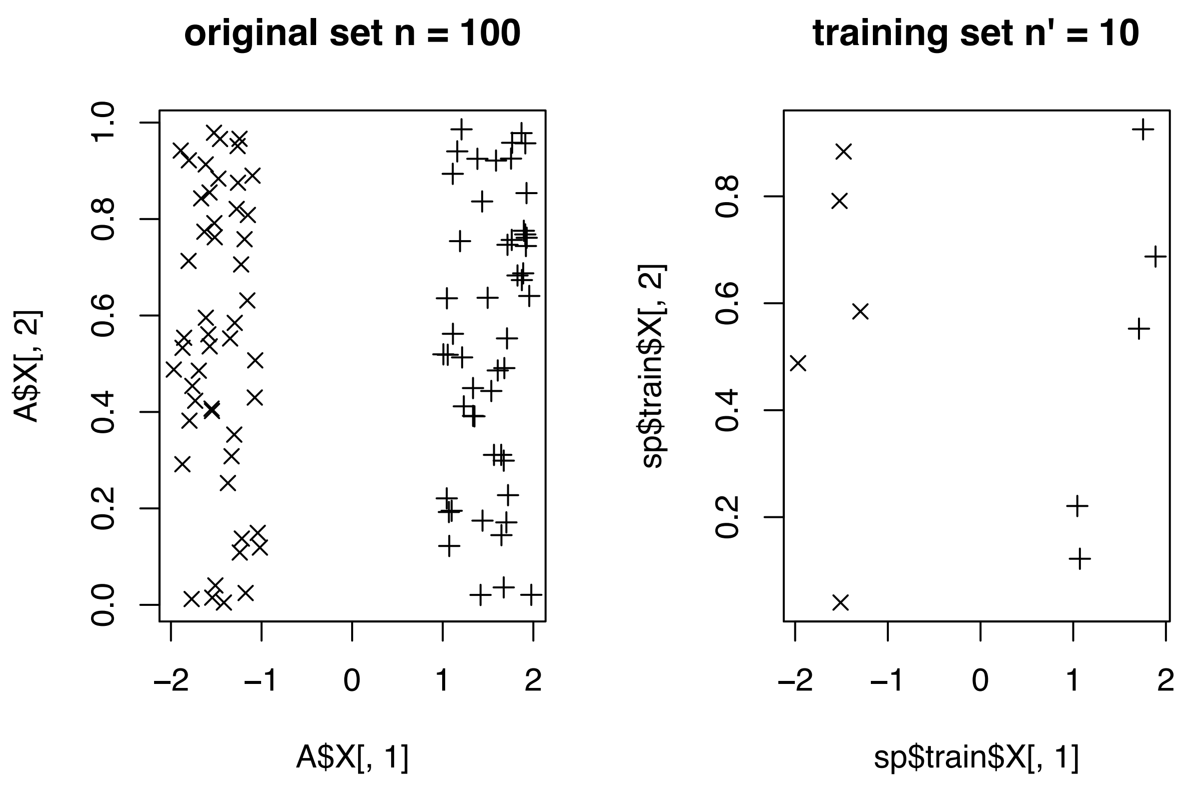
```
datasplit <- function(A,k){
  s = sample(1:nrow(A[[1]]),k)
  list(train = list(X = A[[1]][s,],y = A[[2]][s]) ,
       test =list( X = A[[1]][-s,],y = A[[2]][-s]) )
}
```

Part 6

Visualize a full data set of $n = 100$ examples and a random size $n' = 10$ training data set. You can make the visualization side by side, or plot them in the sample plot using color to distinguish those points that are in the training data set.

```
par(mfrow = c(1,2))
sp = datasplit(A,10)
plot(A$X[,1],A$X[,2],
     pch = mapply((function(x){ if (x == 1) 3 else 4 }),A$y),
     main = "original set n = 100")
```

```
plot(sp$train$X[,1],sp$train$X[,2],
     pch = mapply((function(x){ if (x == 1) 3 else 4 }),sp$train$y),
     main = "training set n' = 10")
```



Part 7 Write a procedure that takes 1) an initial classifier v_H and 2) a labeled data set, and implements the perceptron algorithm with the step size rule $\alpha(k) = 1$ k , where k is the number of the current iteration. The procedure should return the learned classifier vH if the data are linearly separable, or should return an error message otherwise, stating the data are not linearly separable.

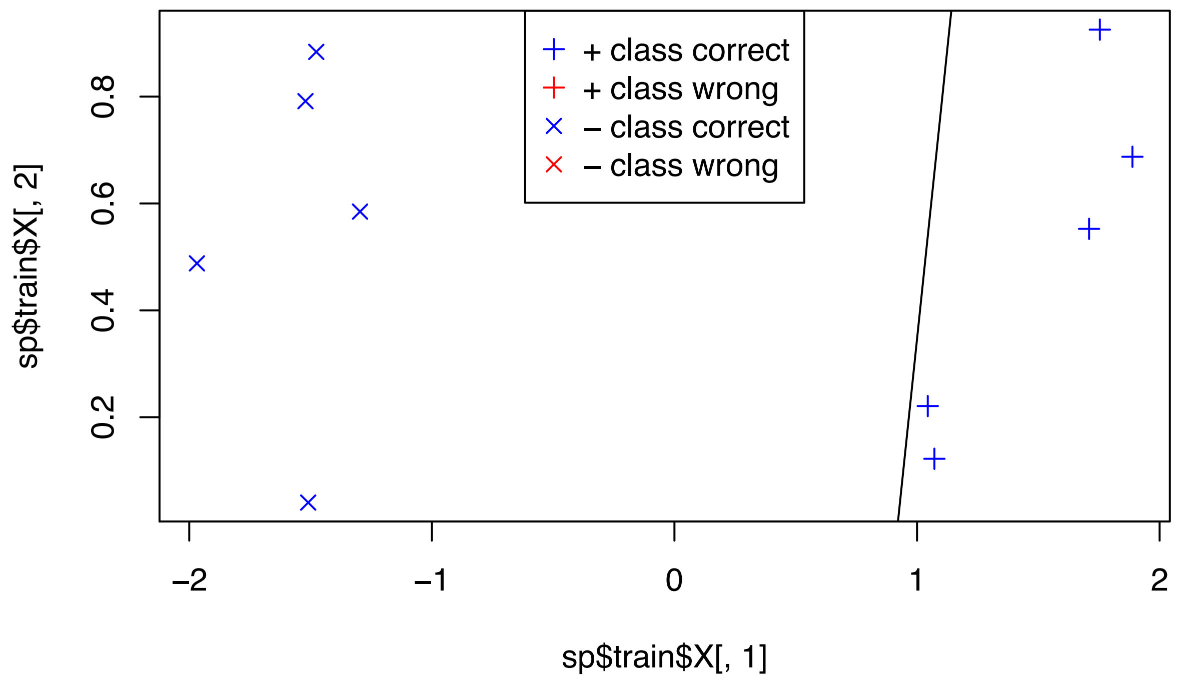
```
cost <- function(A,z){
  f_x =  cbind(rep(1,nrow(A$X)),A$X) %*%z
  sum(mapply((function(x){ if (x == F) 1 else 0 }),sign(f_x) == A$y) * abs(f_x) )
}

grad_cost <- function(A,z){
  f_x =  cbind(rep(1,nrow(A$X)),A$X) %*%z
  t((mapply((function(x){ if (x == F) 1 else 0 }),sign(f_x) == A$y) *(-A$y))%*% cbind(rep(1,nrow(A$X)),
}

perceptron <- function(A,z_0){
  max_it = 10000
  k = 1
  cp = cost(A,z_0)
  z = z_0
  while (cp != 0 & k < max_it ) {
    z = z - (1/k)*grad_cost(A,z)
    cp = cost(A,z)
    k =k +1
  }
  if (cp == 0) z else "error not linearly seperatble"}
```

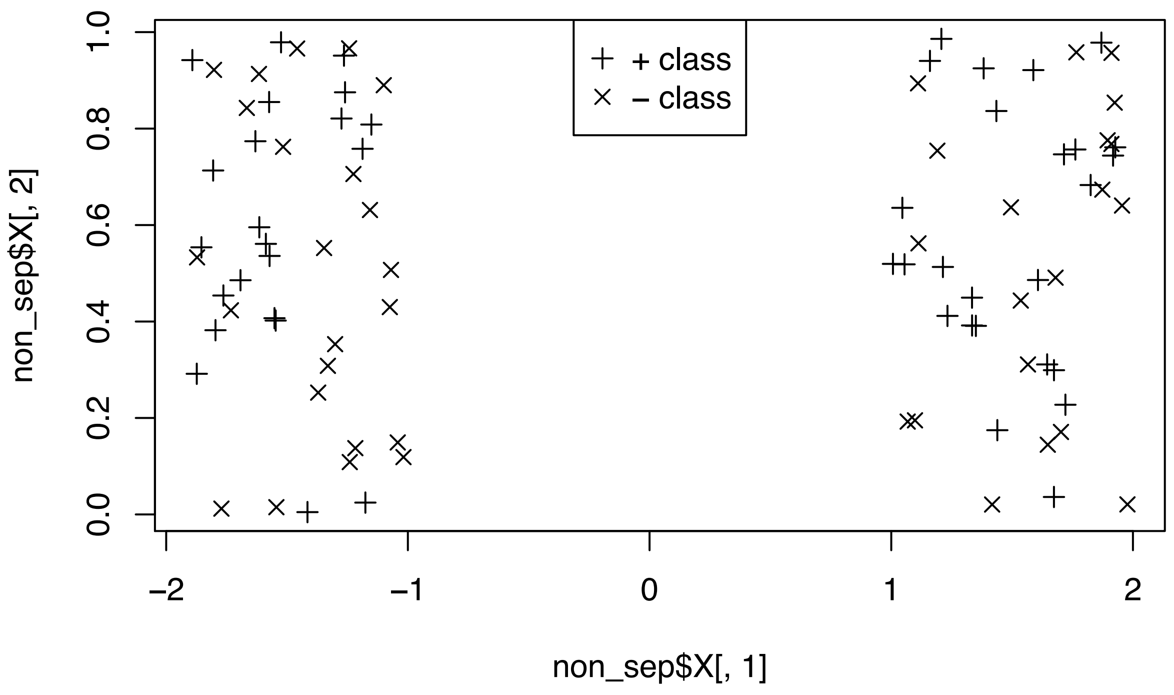
Part 8

```
z_0 = c(1,0,1)
z = perceptron(sp$train,z_0)
plot(sp$train$X[,1],sp$train$X[,2],
     pch = mapply((function(x){ if (x == 1) 3 else 4 }),sp$train$y),
     col = mapply((function(x){ if (x == T) "blue" else "red" }),classify(sp$train$X,z)))
legend("top",
       legend = c("+ class correct","+ class wrong","- class correct","- class wrong"),
       pch = c(3,3,4,4),col = c("blue","red"))
abline(a = z[1],b = -z[2]/z[3])
```



```
non_sep = list(X = sp$test$X, y = mapply((function(x){ if (x == T) 1 else -1 }),runif(length(sp$test$y)

plot(non_sep$X[,1],non_sep$X[,2],
     pch = mapply((function(x){ if (x == 1) 3 else 4 }),non_sep$y))
legend("top",legend = c("+ class","- class"),pch = c(3,4))
```



```
perceptron(non_sep,z_0)
```

```
## [1] "error not linearly seperatble"
```