hw6 p7

For fractions:

```
library(MASS)
```

R Markdown

```
#import data
data <- read.table("T1-6.DAT")

#set up X
X_1<- as.matrix(data)[1:69,1:5]
X_2<- as.matrix(data)[70:98,1:5]

names <- c('(Age)','(SlL + SIR)','|S1L - S1R|','(S2L + S2R)','|S2L - S2R|')
colnames(X_1)<- names
colnames(X_2) <- names

n1 <- length(X_1[,1])
n2 <- length(X_2[,1])
r <- length(names)
```

Classifier function

```
sampleMean<-function(X, n) {
  Ones <- rep(1,n)
  return (1/n * t(X)%*%Ones)
}
sampleCovariance<-function(X, n, sample_mean) {
  Ones <- rep(1,n)
  return (1/(n-1) * t(X - Ones%*%t(sample_mean))%*%(X - Ones%*%t(sample_mean)))
}

# If x_0 is NULL, the classifier is printed and nothing else is done
fisher<-function(X_1, X_2, x_0=NULL) {
  n1 <- length(X_1[,1])
  n2 <- length(X_2[,1])

  # Compute sample mean and S
  x1_sample_mean <- sampleMean(X_1, n1)
  x2_sample_mean <- sampleMean(X_2, n2)
  S1 <- sampleCovariance(X_1, n1, x1_sample_mean)
  S2 <- sampleCovariance(X_2, n2, x2_sample_mean)
  Spooled <- (n1-1)/(n1+n2-2) * S1 + (n2-1)/(n1+n2-2) * S2

  # Compute the classification
  w <- t(x1_sample_mean - x2_sample_mean) %*% solve(Spooled)
  mid <- 1/2*(x1_sample_mean + x2_sample_mean)

  # Print our classifier or classify the value
  if (is.null(x_0))
    cat(w, '*(x_0 -', mid, ') >= 0')
  else {
    result <- w%*%(x_0 - mid)
    if (result >= 0) {
      return (1)
    }
    return (2)
  }
}
```

Print the classiffier obtained

```
# Use the classifier with our whole population, without predicting anything
fisher(X_1, X_2)
```

```
## 0.02340633 -0.03446657 0.2102708 -0.08393327 -0.2534507 *(x_0 - 40.02724 162.779
4 6.91909 216.267 7.351524 ) >= 0
```

Compute the apparent error rate (AER):

```
# Returns 0 if then prediction is correct, 1 otherwise.
classify<-function(X_1, X_2, x_0, expeted_class) {
  if ( fisher(X_1, X_2, x_0) != expeted_class ) {
    return (1)
  }
  return (0)
}

# Use the classifier with our whole population, without predicting anything
errors <- 0
for (i in 1:n1) {
  errors <- errors + classify(X_1, X_2, X_1[i,], 1)
}
for (i in 1:n2) {
  errors <- errors + classify(X_1, X_2, X_2[i,], 2)
}
AER <- errors / (n1+n2)
cat( fractions(AER), '=', errors, '/', (n1+n2))
```

```
## 0.1020408 = 10 / 98
```

Compute the expected actual error rate (EAER):

```
# Use the classifier with our whole population, without predicting anything
errors <- 0
for (i in 1:n1) {
  errors <- errors + classify(X_1[-i,], X_2, X_1[i,], 1)
}
for (i in 1:n2) {
  errors <- errors + classify(X_1, X_2[-i,], X_2[i,], 2)
}
EAER <- errors / (n1+n2)
cat( fractions(EAER), '=', errors, '/', (n1+n2))
```

```
## 0.1326531 = 13 / 98
```