module\_9\_assignment\_rmd

# Load optidigits data  
train <- read.table('optdigits.tra', sep = ',')  
table(train$V65)  
# test <- read.table('optdigits.tes', sep = ',')  
# table(test$V65)  
  
# V1-V64 are features, V65 is target vector  
# attributes are ranged 0:16  
# class are ranged 0:9  
# no N/A  
X <- train[,1:(ncol(train)-1)]  
sample\_means <- colMeans(X)  
sample\_sd <- sqrt(diag(cov(X)))  
X\_list <- as.list(X)  
  
# Reference: https://stackoverflow.com/questions/39731068/how-to-let-a-matrix-minus-vector-by-row-rather-than-by-column  
# Reference: https://stackoverflow.com/questions/3444889/how-to-use-the-sweep-function  
X\_centered <- sweep(X, 2, colMeans(X))  
X\_standardized <- sweep(X\_centered, 2, sample\_sd, FUN = "/")  
### The first column is all zeros, cannot be standardized  
X\_standardized$V1 <- 0  
X\_standardized$V40 <- 0  
sum(is.na(X\_standardized))  
X <- X\_standardized  
  
### Normalization stats  
colMeans(X) # roughly zero  
diag(cov(X)) # all ones except cols 1, 40  
### Normalization stats end  
  
S <- cov(X)  
eigens <- eigen(S)  
W <- eigens$vectors # W is the PC matrix that is 64x64  
Z <- as.matrix(X) %\*% as.matrix(W) # projection matrix  
  
# Reconstruction  
# x\_hat <- t(Z[1,]) %\*% t(W) + sample\_means # single observation  
X\_hat <- Z %\*% t(W) + sample\_means # reconstruction matrix  
  
# Create an X-hat for 1-64 PC's  
X\_hat\_levels <- lapply(1:64, function(x) {  
 Z <- as.matrix(X) %\*% as.matrix(W[,1:x])  
 X\_hat <- Z %\*% t(W[,1:x]) + sample\_means  
 X\_hat  
})  
  
reconstruction\_error\_list <- lapply(X\_hat\_levels, function(x) {  
 sum(rowSums((x - X)^2))  
})  
reconstruction\_error\_df <- do.call(rbind, reconstruction\_error\_list)  
plot(1:64, reconstruction\_error\_df, type = 'l',  
 main = 'E(n) vs. n', xlab = '# of PC\'s', ylab = 'Reconstruction Error')  
points(1:64, reconstruction\_error\_df, pch = 1)  
  
# Reconstruction error  
a = X\_hat - X  
total\_reconstruction\_error <- rowSums((X\_hat - X)^2)  
  
  
### analyze Z  
# PC plot  
Z\_sub <- Z[,c(1,2)]  
N <- nrow(train)  
# Reference: https://stackoverflow.com/questions/2579995/control-the-size-of-points-in-an-r-scatterplot  
plot(Z\_sub[,1], Z\_sub[,2], col = scales::alpha('gray', 0.4),  
 xlab = 'PC1', ylab = 'PC2', main = 'Optidigts PCA', pch=19, cex=0.2)  
set.seed(1); sample\_classes <- sort(sample(1:N, 2e2))  
label\_sample <- train$V65[sample\_classes]  
# Reference: https://rpubs.com/RatherBit/188960  
text(x = Z\_sub[sample\_classes,1], y = Z\_sub[sample\_classes,2],  
 labels = label\_sample, col = scales::alpha('red', 0.4))  
### end analysis