Homework Assignment 03

The Due Date : By 1:30pm, December, 1^{st} (Thursday)
Your solution should include R codes and the answer of each question.
You need to upload your homework on `http://plato.pusan.ac.kr` for full credits.
You may collaborate on this problem but you must write up your own solution.

Perform a simulation study to compare classification performance of statistical learning methods. First, generate the simulation data such that

```
> library(mvtnorm)
> set.seed(112233)
> K <- 100; n <- 200; p <- 20
> x.tran <- x.test <- x.vald <- array(0, c(n, p, K))
> z <- rep(c(1, 2, 3), each=n/2)
> for (i in 1:K) {
+      c <- runif(1, 0, 0.3)
+      cov <- matrix(c, p, p); diag(cov) <- 1
+      t <- sample(1:p, 1); s <- sample(1:p, t)
+      mu <- rep(0,p); mu[s] <- runif(t, -1, 1)
+      x1 <- rmvt(3*n/2, delta=mu, sigma=diag(p), df=9)
+      x2 <- rmvt(3*n/2, delta=rep(0, p), sigma=cov, df=9)
+      x.tran[,,i] <- rbind(x1[z==1,], x2[z==1,])
+      x.test[,,i] <- rbind(x1[z==2,], x2[z==2,])
+      x.vald[,,i] <- rbind(x1[z==3,], x2[z==3,])
+ }
> y <- as.factor(c(rep(1, n/2), rep(-1, n/2)))
```

In this simulation data, a training set (`x.tran`), a test set (`x.test`) and a validation set (`x.vald`) were generated, where the samples of each set consist of 100 cases ($y_i = 1$) for the first 100 observations and 100 controls ($y_i = -1$) for the other 100 observations, i.e., the sample size is $n = 200$. The number of variables is $p = 20$ and the number of simulation replications is $K = 100$. For the k-th simulation replication, you have to build a classifier $c(x)$ from the training set (`x.tran[,,k]`) and then find the optimal tuning parameter from the validation set (`x.vald[,,k]`). Finally, apply the classifier $c(x)$ with the optimal tuning parameter to the test set (`x.test[,,k]`) and compute the classification error rate (CER) of the test set. The final CER of the classifier $c(x)$ should be averaged over 100 simulation replications.

1. Build 4 classifiers including LR(Logistic Regression), LDA(Linear Discriminant Analysis), QDA(Quadratic Discriminant Analysis) and NB(Naive Bayes) from the training set. For each classifier, compute the prediction probability of the i-th validation observation, $p(y_i = 1|x_i)$, and classify $\hat{y}_i = 1$ if $p(y_i = 1|x_i) > \delta$ and otherwise $\hat{y}_i = -1$. The threshold δ begins from 0.1 to 0.9 increased by 0.01. You need to find the optimal threshold from the validation set. If multiple thresholds have the same smallest CER in the validation set, take the sample mean of the multiple thresholds as the optimal threshold. Find the averaged CERs of the test sets of four classifiers.

2. Fit a classification tree for the training set and find the optimal size to prune the tree, using a

`prune.misclass` function for the validation set. Finally, apply the optimal tree to the test set and compute the averaged test CER. If the optimal tree has only one node, fix CER=0.5.

3. Fit random forest for the training set where the number of predictors is considered as 1, 2, ..., 10. Find the optimal number of predictors which has the smallest CER for the validation set. If you have a tie of CER, just select a smaller value of the number of predictors. Finally, apply random forest with the optimal number of predictors to the test set and compute the averaged test CER. Fix the number of trees as 500 (i.e., `ntree=500` in the `randomForest` function).

4. Fit a boosting model with the training set, using a `gbm` function where `interaction.depth` is considered as 1, 2, 3, or 4. Find the optimal number of `interaction.depth` that has the smallest CER for the validation set. Also, you will have the prediction probability of the i-th validation observation, $p(y_i = 1|x_i)$, and classify $\hat{y}_i = 1$ if $p(y_i = 1|x_i) > \delta$ and otherwise $\hat{y}_i = -1$. The threshold δ starts from 0.1 to 0.9 increased by 0.01. You also need to find the optimal threshold from the validation set. If you have a tie of CER, first select the smaller value of `interaction.depth`, and then take an average of thresholds that have the same smallest CER. Finally, apply the boosting model with the optimal number of `interaction.depth` and the optimal value of threshold to the test set and compute the averaged test CER. Fix the number of trees as 100 (i.e., `n.trees=100` in the `gbm` function).

5. Repeat Q4, replacing the `gbm` function by the `adaboost` function in an R package `JOUSBoost`. Refer the manuals of `adaboost` functions. Similar to Q4, you also need to find the optimal `tree_depth` among 1, 2, 3 and 4, and the optimal value of the threshold from the validation set in the same way. Do not change the other arguments in the function, including `n_rounds`.

6. Fit a lasso with the following `lambda` values.

```
> lambda <- seq(0.00007, 0.26, length.out=200)
```

   Apply a `glmnet` function to the training set using `family="binomial"`. For each `lambda` value, compute the prediction probability of the i-th validation observation, $p(y_i = 1|x_i)$, where $\hat{y}_i = 1$ if $p(y_i = 1|x_i) > \delta$ and otherwise $\hat{y}_i = -1$. The threshold δ starts from 0.1 to 0.9 increased by 0.01. Find the optimal `lambda` and threshold that minimize the CER of validation set. If there exist multiple `lambda` and thresholds that has the same smallest CER, first pick up a smaller `lambda` and then take an average of the thresholds that have the smallest CER with the smaller `lambda`. Finally, apply the lasso with the optimal `lambda` and the optimal value of threshold to the test set and compute the averaged test CER.

7. From Q1 to Q6, you have applied 9 statistical methods such as LR, LDA, QDA, NB, tree, RF (random forest), gbm, adaboost, and lasso. For each method, you have 100 CERs of 100 different test sets. Draw a side-by-side boxplot where 9 methods are on the x-axis and CERs are on the y-axis. Which method has the smallest mean/variance of the test CER? Pick up your best method.