Learning outcomes

After solving these exercises, you should be able to understand the following:

1. Applying the Random Forest and Adaboost algorithms to solve classification problems.
2. Applying stacking techniques
3. Interpreting the results generated from each algorithm in R.
4. Comparison of the model performance in terms of precision, recall and accuracy

Random Forest: Hepatitis Dataset

The hepatitis dataset has 20 variables and 155 records. Use "target" as target variable. Goal is to determine whether the person will live or die.

1. Import the data into R

```
hepatitis <- read.table('hepatitis.txt', header=F, dec='.',
                        col.names=c('target','age','gender','steroid','antivirals',
                        'fatigue','malaise','anorexia','liverBig',
                        'liverFirm','spleen','spiders','ascites',
                        'varices','bili','alk','sgot','albu','protime',
                        'histology'),
                        na.strings=c('?'), sep=',')
```

2. Study dataset

```
str(hepatitis)
table(hepatitis$target)
str(hepatitis$target) # 1: Die; 2: Live
#Convert 1s and 2s into 1s and 0s
hepatitis$target= ifelse(hepatitis$target==1,0,1 ) # 1: Die(+ve); 0: Live (-ve)
```

3. Convert all categorical attributes into factors
4. Fill the missing values using knnImputation.
5. Split dataset into train and test
6. Build the classification model using randomForest
7. Build the classification model using randomForest

```
library(randomForest)
hepatitis_rf <- randomForest(target ~ ., data=trainR, keep.forest=TRUE, ntree=30)
```

8. View results and understand important attributes

```
print(hepatitis_rf)
hepatitis_rf$predicted
hepatitis_rf$importance
```

9. View results and understand important attributes

plot (directly prints the important attributes)

varImpPlot(hepatitis_rf)

10. Predict on Train and Test datasets

11. Calculate precision, recall and accuracy

Adaboost: Universal Bank Dataset

The Universal Bank dataset has 14 variables and 5000 records. Use "Personal.Loan" as target variable.

1. Import the data into R

2. Drop ID & ZIP Code
3. Seperate numerical and categorical attributes
4. Convert all categorical attributes into factors
5. Standardize numerical data using range method
6. Combine both datasets
7. Dummify "Education" attribute
8. Remove the original "Education" attribute after it is dummified.
9. Split the data into train, test and evaluation data sets
10. Build the classification model using Adaboost:

```
library(ada)
x = subset(train_data, select = -Personal.Loan)
y = as.factor(train_data$Personal.Loan)
a = subset(test_data, select = -Personal.Loan)
b = as.factor(test_data$Personal.Loan)

model=ada(x, y, iter=20, loss="logistic") #20 Iterations
```

11. Predict the values using model on test data sets.

```
pred = predict(model, a); pred
```

12. Calculate precision, recall and accuracy

result <- table(pred, b);result # 0(-ve) and 1(+ve)

13. Experiment with different number of iterations and find the best.

Stacking: Universal Dataset

The Universal Bank dataset has 14 variables and 5000 records. Use "Personal.Loan" as target variable.

```
# Remove unwanted attributes (ID, Zipcode)
data <- subset(data, select = -c(1,5))
#  Convert certain identified IV into factors
        #Education, Personal Loan, Securities.Account,CD.Account,Online,
        #Credit Card, Family are all categorical and need to be changed to factors
# Separating numerical and categorical attributes
        str(data)
        data_cat <- subset(data, select = -c(1,2,3,5,7))
        str(data_cat)
        data_num <- subset(data, select = c(1,2,3,5,7))
        str(data_num)
# Converting the attributes of categorical subset into factors
        data_cat <- apply(data_cat, 2, function(x) {as.factor(x)})
        data_cat <- data.frame(data_cat) #need to convert DF as changing to as.factor outputs a list)
        str(data_cat)
# Descritize numerical attributes
        library(infotheo)
        data_num <- apply(data_num, 2, function(x) {discretize(x, disc = "equalfreq", nbins = 4)})
        data_num <- data.frame(data_num)
        names(data_num)[1:5] <- c("Age", "Exp", "Income", "CCAvg", "Mortgage")
        # or colnames(data_num) = c("age", "Exp", "Income", "CCAvg", "Mortgage")
# Combining categorical and numerical datasets
        data_combine <- cbind(data_cat, data_num)
        str(data_combine)
```

```
# Convert to factors
data_combine <- apply(data_combine, 2, function(x) {as.factor(x)})
data_combine <- data.frame(data_combine)
str(data_combine)

# Split into Train and Test
rows <- seq(1, nrow(data_combine),1)
set.seed(2020)
trainrows <- sample(rows, nrow(data_combine)*.65)
trainR <- data_combine[trainrows,] #all rows in trainrows & all columns of parent dataset
testR <- data_combine[-trainrows,]
str(trainR)

# APPLYING SEVERAL MACHINE LEARNING CLASSFICATION TECHNIQUES
#(1)  Build rpart model on the training dataset
        library(rpart)
        cart_obj <- rpart(Personal.Loan ~ ., trainR, method = "class")
        summary(cart_obj)

# predicting on train dataset
cart_pred <- predict(cart_obj, newdata = trainR, type="vector")
table(cart_pred)

# if we choose type=vector, then we will have to use the following ifelse
cart_pred <- ifelse(test = cart_pred == 1, 0, 1) #if 1 replace with 0, # else repl with 1
table(cart_pred)

# prediction on test dataset
        cart_test <- predict(cart_obj, newdata = testR, type="vector")
        cart_test <- ifelse(test = cart_test == 1, 0, 1)
        table(cart_test)
        check1 <- table(testR$Personal.Loan, cart_test)
```

```
        sum(diag(check1))/sum(check1)

# (2) Build C5.0 model on the training dataset
        library(C50)
        dtC50_obj <- C5.0(Personal.Loan ~., trainR, rules = T)
        summary(dtC50_obj)

# predicting with the train dataset
        dtC50_pred <- predict(dtC50_obj, trainR, type = "class")
        dtC50_pred <- as.vector(dtC50_pred)
        table(dtC50_pred)

# prediction on test dataset
        dtC50_test <- predict(dtC50_obj, newdata = testR, type = "class")
        dtC50_test <- as.vector(dtC50_test)
        table(dtC50_test)
        check2 <- table(testR$Personal.Loan, dtC50_test)
        sum(diag(check2))/sum(check2)

# (3) Build Logistic regression on the training dataset
        glm_obj <- glm(formula = Personal.Loan ~ .,data= trainR, family = binomial())
        summary(glm_obj)

# predicting on train dataset
        glm_pred <- predict(glm_obj, trainR, type = "response")
        glm_pred <- ifelse(test = glm_pred > 0.5, 1, 0) #it gives probabilities, so we
        #need to convert to 1's and 0's; if >0.5 show as 1 or else show as 0.
        table(glm_pred)

# prediction on test dataset
        glm_test <- predict(glm_obj, testR, type = "response")
        glm_test <- ifelse(test = glm_test > .5, 1, 0)
```

```
table(glm_test)
check3 <- table(testR$Personal.Loan, glm_test)
sum(diag(check3))/sum(check3)

# (4) Combining training predictions of CART, C5.0 & Log Regression together
train_pred_all_models <- cbind(cart_pred, dtC50_pred, glm_pred)
train_pred_all_models <- data.frame(apply(train_pred_all_models, 2, function(x) {as.factor(x)}))
# or first use "apply" then type data_ensemble <- data.frame(data_ensemble)
str(train_pred_all_models)
summary(train_pred_all_models)
rm(cart_pred, glm_pred, dtC50_pred)

# (5) Viewing the predictions of each model
table(train_pred_all_models$cart_pred) #CART
table(train_pred_all_models$dtC50_pred) #C5.0
table(train_pred_all_models$glm_pred) #Logistic Regression
table(trainR$Personal.Loan) #Original Dataset DV

# (6) Adding the original DV to the dataframe
train_pred_all_models <- cbind(train_pred_all_models, trainR$Personal.Loan)
names(train_pred_all_models)[4] = "target"

# (7) Ensemble Model with GLM as Meta Learner
str(train_pred_all_models)
head(train_pred_all_models)

glm_ensemble <- glm(target ~ ., train_pred_all_models, family = binomial())
summary(glm_ensemble)
```

```
# (8) Check the "glm_ensemble model" on the train data
        pred <- predict(object = glm_ensemble, train_pred_all_models, type = "response")
        pred <- ifelse(test = pred > 0.5, 1, 0)
        table(pred)

        Tab <- table(train_pred_all_models$target,pred)
        accuracy_train = sum(diag(Tab))/sum(Tab)
        accuracy_train
#################################################################
# (9) Combining test predictions of CART, C5.0 & Log Regression together
        test_pred_all_models <- cbind(cart_test, dtC50_test, glm_test)
        test_data_ensemble <- data.frame(test_pred_all_models)
        str(test_pred_all_models)
        head(test_pred_all_models)

# (10) Change column names
        colnames(test_pred_all_models)[1:3] <- c("cart_pred", "dtC50_pred", "glm_pred")
        test_pred_all_models <-as.data.frame(test_pred_all_models)

# (11) Check the "glm_ensemble model" on the test data
        final_pred <- predict(glm_ensemble, test_pred_all_models, type = "response")

        final_pred <- ifelse(test = final_pred > 0.5, 1, 0)
        table(final_pred)
        tab_test <- table(testR$Personal.Loan, final_pred);tab_test
        test_accuracy <- sum(diag(tab_test))/sum(tab_test);test_accuracy
#################################################################################
```