Learning outcomes:

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

1. Applying gbm, xgboost and Adaboost algorithms to solve classification problems.
2. Introduction to h2o.
3. Interpreting the results generated from each algorithm in R.
4. Comparison of the model performance in terms of precision, recall and accuracy

Pre-processing: Universal Bank Dataset:

The Universal Bank dataset has 14 variables and 5,000 records. Use "Personal.Loan" as target variable.

1. Import the data into R
2. Drop the features based on the data understanding.
3. Convert the features into appropriate data type.
4. Convert all categorical attributes into factors.

GBM: Universal Bank Dataset

1. After applying the pre-processing Step1-4
2. Split the data into train and evaluation data sets.

 # Buliding GBM in R using caret package

3. Building the GBM model in R

```
# Bulding gbm using caret package in R
fitControl <- trainControl(method = "repeatedcv", number = 4, repeats = 4)
gbmFit1 <- train(loan ~ ., data = train, method = "gbm", trControl =
fitControl,verbose = FALSE)
```

4. Predicting on the train and test data

```
# Predict on the test data
trainpedgbm <- predict(gbmFit1, data = train)
testgbm <- predict(gbmFit1, test)
```

5. Evaluate the performance of the model on test data

```
# Error metrics on the test data
confusionMatrix(test$loan,testgbm,positive = "1")
```

Buliding GBM in R-H2o using caret

6. Initialize the h2o

```
library(h2o)
# Start H2O on the local machine using all available cores and with 4 gigabytes of memory
h2o.init(nthreads = -1, max_mem_size = "2g")
```

7. Convert the train and test data to h2o objects.

```
# Import a local R train data frame to the H2O cloud
train.hex <- as.h2o(x = train, destination_frame = "train.hex")
# Import a local R test data frame to the H2O cloud
test.hex <- as.h2o(x = test, destination_frame = "test.hex")
```

8. Building the GBM model with different hyper tuning parameters by doing a grid search.
9. Identify the best model and hyper parameters based on AUC.
10. Predicting on the test data using the best model.

```
# Predict on same test data set
predict.hex = h2o.predict(best_GBM_model,
                          newdata = test.hex[,setdiff(names(test.hex), "loan")])
data_GBM = h2o.cbind(test.hex["loan"], predict.hex)
```

11. Convert the h2o object into R data frame

```
# Copy predictions from H2O to R
pred_GBM = as.data.frame(data_GBM)
```

12. Evaluate the performance of the model on the test data.

XGBoost: Universal Bank Dataset

1. After applying the pre-processing Step1-4
2. Convert all the categorical features into numeric data.
3. Split the data into train and test data.

4. Standardize the data.
5. Convert the data into DMatrix form.

```
# Constructing the Dense matrix on the train and test data
dtrain = xgb.DMatrix(data = as.matrix(train_Data[,ind_Attr]),
                     label = train_Data$loan)
dtest = xgb.DMatrix(data = as.matrix(test_Data[,ind_Attr]),
                    label = test_Data$loan)
```

6. Creating a watchlist to evaluate the model performance on the test data

```
#Use watchlist parameter. It is a list of xgb.DMatrix, each of them tagged with
a name.
watchlist = list(train=dtrain, test=dtest)
```

7. Build the xgboost model.

```
# Building the xgboost model
model = xgb.train(data=dtrain, max.depth=4,
                  eta=0.3, nthread = 2, nround=20,
                  watchlist=watchlist,
                  eval.metric = "error",
                  objective = "binary:logistic", verbose = 1)
```

8. Identify the important features.

```
importance <- xgb.importance(feature_names = setdiff(names(final_Data),
"loan"), model = model)
print(importance)
xgb.plot.importance(importance_matrix = importance)
```

9. Predict on the test data.

```
# prediction on test data
pred <- predict(model, as.matrix(test_Data[,ind_Attr]))
prediction <- as.numeric(pred > 0.5,1,0)
```

prediction <- as.factor(as.character(prediction))

10. Evaluate the performance of the model

```
confusionMatrix(testR$loan, prediction,positive = "1")
```

Adaboost: Universal Bank Dataset

1. After applying the pre-processing Step1-4
2. Split the data into train and evaluation data sets.
3. Build the classification model using Adaboost:

```
# build the classification model using Adaboost
library(ada)
model = ada(loan ~ ., iter = 20,data = trainR, loss="logistic")
```

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

```
# predict the values using model on test data sets.
pred = predict(model, testR);
pred
```

5. Evaluate the performance of the model

```
confusionMatrix(testR$loan,pred,positive = "1")
```

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