Notebook 3: Ensemble Techniques

Code

Group 10 (Umar, Carolline)

October 22, 2022

Introduction

This notebook will demonstrate the use of various ensemble models. For this demonstration we'll use decision trees as a baseline, and use other ensemble techniques such as the following:

- XGboost
- Random Forest
- Support Vector Machine(SVM)

About this Set

Cardiovascular Disease (https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset?select=cardio_train.csv) was downloaded from Kaggle.com. The dataset consists of 70 000 records of patients data, 11 features + target.

Getting Started

Hide

```
# Importing Libraries
library(caret)
library(tidyverse)

# Importing data sets
dataset = read.csv('cardio_train.csv')

# Running a few data exploration functions.
glimpse(dataset)

summary(dataset)

# Converting the target variable into factor levels
dataset$CARDIO_DISEASE = as.factor(dataset$CARDIO_DISEASE)
```

Splitting into train and test

Hide

```
split = sample.split(dataset$CARDIO_DISEASE, SplitRatio = 0.8)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)
```

Creating Baseline Decission Tree

Hide

```
# specifying the CV technique which will be passed into the train() function later and number para
meter is the "k" in K-fold cross validation
train_control = trainControl(method = "cv", number = 5, search = "grid")

## Customsing the tuning grid (ridge regression has alpha = 0)
classification_Tree_Grid =  expand.grid(maxdepth = c(1,3,5,7,9))

set.seed(50)

# training a Regression model while tuning parameters (Method = "rpart")
model = train(CARDIO_DISEASE~., data = training_set, method = "rpart2", trControl = train_control,
tuneGrid = classification_Tree_Grid)

# summarising the results
print(model)
```

Making Predictions on Baseline Model

Hide

```
# Using baseline model to make predictions on test set
pred_y = predict(model, test_set)

# Confusion Matrix
confusionMatrix(dataset = pred_y, test_set$CARDIO_DISEASE)
```

XGboost

Hide

```
install.packages('xgboost')
library(xgboost)

train_label <- ifelse(training_set$CARDIO_DISEASE==1, 1, 0)
train_matrix <- dataset.matrix(training_set[, -31])
model <- xgboost(dataset = train_matrix, label = train_label, nrounds = 100, objective='binary:log
istic')
```

Hide

```
test_label <- ifelse(test_set$CARDIO_DISEASE==1, 1, 0)
test_matrix <- dataset.matrix(test_set[, -31])

probs <- predict(model, test_matrix)
pred <- ifelse(probs > 0.5, 1, 0)

acc_xg <- mean(pred==test_label)
cc_xg <- mcc(pred, test_label)

print(paste("Accuracy: ", acc_xg))
print(paste("Correlation Coefficient: ", cc_xg))
```

Random Forest

Hide

```
library(randomForest)
library(caret)
library(e1071)

# Define the control
trControl <- trainControl(method = "cv",
    number = 10,
    search = "grid")

set.seed(1234)

# Run the model
rf_default <- train(CARDIO_DISEASE~.,
    data = training_set,
    method = "rf",
    metric = "Accuracy",
    trControl = trControl)

# Print the results
print(rf_default)

# Testing 20 values
set.seed(1234)

tuneGrid <- expand.grid(.mtry = c(1: 10))
rf_mtry <- train(CARDIO_DISEASE~.,
    data = training_set,
    method = "rf",
    metric = "Accuracy",
    tuneGrid = tuneGrid,
    trControl = trControl,
    importance = TRUE,
    nodesize = 14,
    ntree = 300)
print(rf_mtry)
```

Support Vector Machine (SVM Classification)

Hide

```
install.packages('e1071')
library(e1071)

svm_c <- svm(CARDIO_DISEASE~., data=training_set, kernel="linear", cost=10, scale=TRUE)
summary(svm_c)
```

Evaluating and Plotting Results

In this line of code veiwers can use the following R code to make evaluation based on their

Hide

```
pred <- predict(svm_c, newdata = test_set)
table(pred, test_set$CARDIO_DISEASE)
mean(pred==test_set$CARDIO_DISEASE)
```

following the evaluation users can visualize the output by plotting the support vectors.

Hide

```
plot(svm_c, test_set, WEIGHT - CHOLESTEROL, slice - list(CARDIO_DISEASE = 1, CARDIO_DISEASE = 0))
```