Practical Machine Learning Project

Minh Vo

10/27/2020

Synopsis

In this project, I use data collected from accelerometers on the belt, forearm, arm, and dumbell of 6 participants, along with machine learning techniques to predict the manner in which people do exercise.

The data

The training and testing data for this project, respectively, are downloaded from the provided links :

```
url_trn = 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
url_tst = 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'
download.file(url = url_trn, destfile = 'pml_training.csv', method = 'curl')
download.file(url = url_tst, destfile = 'pml_testing.csv', method = 'curl')
```

Loading and Cleaning Data

I first load the data and do some cleaning. Since there are a lot of missing data in the form of blank, I first fill the blank with NA and remove variables with more than 50% NAs.

```
trn = read.csv('pml_training.csv')
tst = read.csv('pml_testing.csv')

# Replace empty cells in the training and testing data with NA
trn[trn==''] = NA
tst[tst==''] = NA

# Remove variables with majority NAs
rmNA_trn = apply(trn,2, function(x){mean(is.na(x))>.50})
training = trn[, !rmNA_trn]

rmNA_tst = apply(tst,2, function(x){mean(is.na(x))>.50})
testing = tst[,!rmNA_tst]

dim(training)
```

```
## [1] 19622    60
```

```
dim(testing)
```

```
## [1] 20 60
```

After the process, 60 variables remain: 1 outcome and 59 predictors. I further remove the first seven variables because they are related to identification and time stamp and thus have no forecasting power. The remaining data consist of 52 predictors and 1 outcome. I then convert the outcome *classe* to a factor variable.

```
trndat <- training %>% select(-c(1:7))%>%
        mutate(classe = factor(classe, levels = c('A', 'B', 'C', 'D', 'E')))
tstdat <- testing %>% select(-c(1:7)) %>% rename(classe = problem_id)
```

Model Selection

To select the best model for the data, I use the (cleaned) training data set to fit and evaluate 3 models: Random Forest (RF), Generalized Boosted Model (GBM) and Linear Discriminant Analysis (LDA). I then use the best one in terms of forecast accuracy to fit the entire training data and then apply it to the testing data set. To that end, I partition the training data set *trndat* into the *selTraining* to train the three models, and *selTesting* to evaluate them.

To reduce the number of predictors, I use 'pca' preProcess option in the train function. Furthermore, I use 5-fold cross validation when I train the models.

```
# Partition trndat into training and testing for model selection
set.seed(456)
inTrain <- createDataPartition(trndat$classe, p=0.7, list=FALSE)
selTraining <- trndat[inTrain, ]
selTesting <- trndat[-inTrain, ]

# Fitting 3 models: GBM, RF and LDA
ctrl = trainControl(method = 'cv', number = 5)
sel_gbm = train(classe ~ ., data = selTraining, method='gbm', trControl = ctrl, verbose = F, preProcess=
sel_rf = train(classe ~ ., data = selTraining, method='rf', trControl = ctrl, verbose = F, preProcess='
sel_lda = train(classe ~ ., data = selTraining, method='lda', trControl = ctrl, verbose = F, preProcess=

# Predicting
pred_gbm = predict(sel_gbm, selTesting)
pred_rf = predict(sel_rf, selTesting)
pred_lda = predict(sel_lda, selTesting)

# Evaluating their accuracy
confusionMatrix(pred_rf, selTesting$classe)$overall[1] # RF accuracy
```

```
##  Accuracy
## 0.9789295
```

```
confusionMatrix(pred_gbm, selTesting$classe)$overall[1] # GBM accuracy
```

```
##  Accuracy
## 0.8258284
```

```
confusionMatrix(pred_lda, selTesting$classe)$overall[1] #LDA accuracy
```

```
## Accuracy
## 0.528972
```

Forecasting

Since the RF model yields the best accuracy, it is chosen to fit the training data and forecast using the testing data.

```
set.seed(1234)
rf = train(classe ~ ., data = trndat, method='rf', verbose = F, preProcess='pca')
```

```
predict(rf, tstdat)
```

```
##  [1] B A A A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```

Based on the quiz result, the accuracy of the model forecast is 95%.