Coursera Machine Learning Project

Janamejaya Chowdhary

June 25, 2017

Background

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement - a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it.

Barbell lifts were performed by six participants. Each participant attempted the barbell lift in five different ways: according to specification, i.e., correct method (Class A), throwing the elbow to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D), and throwing the hips to the front (Class E). Classes B to E are incorrect ways of performing the task. Data is generated by accelerometers on the belt, forearm, arm, and dumbbell of each participant as they perform the lifts. Each measurement was made 10 times. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har%5B1%5D (see the section on the Weight Lifting Exercise Dataset).

Objective

The goal of your project is to predict the manner in which the participants performed the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Data Ingestion (aka Acquiring and loading data)

The data is provided as part of the project from the following URLs.

```r
# Assign the URL from which the training and testing data will be downloaded
urltrain <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
urltest <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
```

First, this data is downloaded to the local directory called data and loaded into data frames

```r
# If not downloaded, download data from predefined urls urltrain and urltest
if(!file.exists("./data"))
{
 dir.create("./data")
 download.file(urltrain,destfile="./data/training.csv", method="curl")
 download.file(urltest,destfile="./data/testing.csv", method="curl")
}

# Load in the data.
# Previous experimentation shows several occurances of NA, "", and "#DIV/0!"
# These three strings are set as possible NA values
training <- read.csv("./data/training.csv", na.strings=c("NA", "", "#DIV/0!"), stringsAsFactors=FALSE)
testing <- read.csv("./data/testing.csv", na.strings=c("NA", "", "#DIV/0!"), stringsAsFactors=FALSE)
```

Data Mastication (aka Preprocessing)

Lets take a look at the size of the training and testing data

```
dim(training); dim(testing)
```

```
## [1] 19622   160
```

```
## [1]  20 160
```

The number of rows in the training dataset, 19622, is much larger than the testing dataset which has 20 rows.

Common and distinct variables in Training and Testing data

Check how many columns in the training and testing datasets have identical names.

```
sum(colnames(training)==colnames(testing))
```

```
## [1] 159
```

This number differs from the number of columns, 160 in the training and testing datasets. Lets find out which one column is unique to the training and testing datasets

```
setdiff(colnames(training), colnames(testing))
```

```
## [1] "classe"
```

```
setdiff(colnames(testing), colnames(training))
```

```
## [1] "problem_id"
```

Set classe to be a factor variable. Since this is the target variable, lets look at the distribution of values for each unique classe value.

```
training$classe <- as.factor(training$classe)
summary(training$classe)
```

```
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

Clearly no unique classe is underrepresented and additional sampling is not required.

Remove non-accelerometer data columns from Training data

In the training dataset, the first 7 variables X, user_name, raw_timestamp_part_1, raw_timestamp_part_2, cvtd_timestamp, new_window, num_window do not appear to contain any accelerometer related information and they should be ignored. Implicit in this decision is the assumption that the name of the user and the time stamps do not contain any information that could be used through a model for making predictions on the test dataset.

```
training <- training[,-c(1:7)]
testing <- testing[,-c(1:7)]
```

Remove columns with majority of missing values in Training data

Now find the fraction of NA values in each column and remove those columns for which more than 30% of the data is missing.

```
colselect <- (colSums(is.na(training))/nrow(training))<0.3
training <- training[colselect]
```

Remove columns showing near zero variation in Training data

Next, load the caret library and identify the columns that have near zero variation. Remove them.

```r
library(caret)
# Find the nearzero variance data columns in the training dataframe
tmp1 <- nearZeroVar(training)
# Remove all nearzero variance columns
if(length(tmp1)>0) { training <- training[,-tmp1] }
```

Remove columns of correlated variables in Training data

Next remove all variables from the training dataset which have absolute value of the correlation coefficient larger than a cutoff of 0.95

```r
corrmat <- cor(training[-ncol(training)])
removecols <- findCorrelation(corrmat, cutoff=0.95)
training <- training[-removecols]
```

Prune the testing dataset

Next prune the testing dataset by retaining only those columns which are common with the training dataset

```r
# Retain data for all remaining variables in training,other than classe, in the
# testing dataset
testing <- testing[,colnames(training[,-ncol(training)])]
dim(training); dim(testing)
```

```
## [1] 19622    49
```

```
## [1] 20 48
```

Now look for the number of columns with missing values in the testing data

```r
sum(colSums(is.na(training))>0); sum(colSums(is.na(testing))>0)
```

```
## [1] 0
```

```
## [1] 0
```

Data Digestion (aka Modeling the data)

Model choice

Several models such as Random Forests (RF), Gradient Boosted Model, Linear Discriminant Analysis, Support Vector Machines, and regression using LASSO were applied for the classification task (comparative analysis not presented). The most successful model was found to be Random Forests and will be utilized in the following.

Set up cluster for parallel processing

```r
library(parallel)
library(doParallel)
ncores <- detectCores()-1
cluster <- makeCluster(ncores)
registerDoParallel(cluster)
```

Test-Validation data Split

The training data is split into 70% test and 30% validation sets.

```
inTrain   <- createDataPartition(training$classe, p=0.75, list=FALSE, times=1)
trainData <- training[inTrain, ]
validData  <- training[-inTrain, ]
```

Set controls for training the model

Ten fold cross-validation will be used for model training.

```
set.seed(201706)
nfolds=10
fitControl <- trainControl(method="cv", number = nfolds, allowParallel = TRUE, seeds=NA)
```

Building and Testing the Random Forest model

Here, trainData is used to train and test the Random Forest model.

```
rfFit <- train(classe~., data=trainData, method="rf", trControl=fitControl)
```

Make predictions with the rfFit model on the training data

```
pred_rf <- predict(rfFit, trainData)
confusionMatrix(pred_rf, trainData$classe)$overall[1]
```

```
## Accuracy
##        1
```

Make predictions with the rfFit model on the validation data

```
pred_rf <- predict(rfFit, validData)
confusionMatrix(pred_rf, validData$classe)$overall[1]
```

```
##  Accuracy
## 0.9930669
```

The model appears to be very sucessful with default setting and no parameter turning is performed. The out of sample error can be calculated as

```
error <- 1.0 - confusionMatrix(pred_rf, validData$classe)$overall[1]
out_of_sample_error <- data.frame(out_of_sample_error=c(error))
out_of_sample_error
```

```
##          out_of_sample_error
## Accuracy         0.006933116
```

Predicting with the model

Now make predictions for the testing data

```
pred_rf <- predict(rfFit, testing)
pred_rf
```

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

Shutdown the cluster

```
stopCluster(cluster)
registerDoSEQ()
```

[1]:Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.