Practical Machine Learning - Course Project

farhan

January 24, 2017

Introduction

For this project, given data from device accelerometer on 6 various device research study participants. Training data consists of accelerometer data and a label identifying the quality of the activity the participant was doing. Testing data consists of accelerometer data without the identifying label. The objective is to predict the labels for the test set observations.

Below is the coding create the model machine learning which are estimating the out-of-sample error, and making predictions. To understand the step, this report will describe each step process.

Data Preparation

Load the caret package, and read in the training and testing data:

```
setwd("D:/Users/TM35460/Desktop/coursera/practical machine learning")
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
trainingDataSet<- read.csv("pml-training.csv", sep=",", header=TRUE, na.strings =
c("NA","",'#DIV/0!'))
testingDataSet<- read.csv("pml-testing.csv", sep=",", header=TRUE, na.strings = c("NA","",'#D
IV/0!'))
dim(trainingDataSet)
```

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

```
dim(testingDataSet)
```

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

On the training set , we seen data consists of 19622 values of 160 variables. On the testing set, we seen data consists of 20 values of 160 variables. For purpose precaution action, include command to remove columns with missing values

```
trainingDataSet <- trainingDataSet[,(colSums(is.na(trainingDataSet)) == 0)]
dim(trainingDataSet)
```

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

```
testingDataSet <- testingDataSet[,(colSums(is.na(testingDataSet)) == 0)]
dim(testingDataSet)
```

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

Preprocess the data

process the data

```
numericalsIdx <- which ( lapply ( trainingDataSet, class) %in% "numeric")

preprocessModel <-preProcess(trainingDataSet[,numericalsIdx],method=c('knnImpute', 'center',
'scale'))
pre_trainingDataSet <- predict(preprocessModel, trainingDataSet[,numericalsIdx])
pre_trainingDataSet$classe <- trainingDataSet$classe

pre_testingDataSet <-predict(preprocessModel,testingDataSet[,numericalsIdx])
```

Clearing the non zero variables

Clear the variables with values near zero, that means that they have not so much meaning in the predictions

```
nzv <- nearZeroVar(pre_trainingDataSet,saveMetrics=TRUE)
pre_trainingDataSet <- pre_trainingDataSet[,nzv$nzv==FALSE]

nzv <- nearZeroVar(pre_testingDataSet,saveMetrics=TRUE)
pre_testingDataSet <- pre_testingDataSet[,nzv$nzv==FALSE]
```

Model building

I decide to use 75% observation training dataset to train our model. We will then validate it on the last 70%.

```
set.seed(12031987)
idxTrain<- createDataPartition(pre_trainingDataSet$classe, p=3/4, list=FALSE)
training<- pre_trainingDataSet[idxTrain, ]
validation <- pre_trainingDataSet[-idxTrain, ]
dim(training) ; dim(validation)
```

```
## [1] 14718    28
```

```
## [1] 4904   28
```

the data for training we create as “training” and the data for validation we create as “validation”.

Build Train Modeling

The data training we train use random forest with a cross validation of 5 classes.

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
modFitrf <- train (classe ~., method="rf", data=training,
trControl=trainControl(method='cv'), number=5, allowParallel=TRUE, importance=TRUE )
modFitrf
```

```
## Random Forest
##
## 14718 samples
##    27 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 13246, 13245, 13248, 13245, 13247, 13246, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9929331  0.9910609
##   14    0.9925931  0.9906307
##   27    0.9893992  0.9865911
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

Cross Validation Testing and Out-of-Sample Error Estimate

Let’s apply our training model on our testing database, to check its accuracy.

Accuracy and Estimated out of sample error

```
predValidRF <- predict(modFitrf, validation)
confus <- confusionMatrix(validation$classe, predValidRF)
confus$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1391    2    0    0    2
##          B    3  945    1    0    0
##          C    0    2  852    1    0
##          D    0    0    3  801    0
##          E    0    0    0    3  898
```

We can notice that there are very few variables out of this model.

```
accur <- postResample(validation$classe, predValidRF)
modAccuracy <- accur[[1]]
modAccuracy
```

```
## [1] 0.9965334
```

```
out_of_sample_error <- 1 - modAccuracy
out_of_sample_error
```

```
## [1] 0.003466558
```

The summary of the model seen the estimated accuracy of the model is 99.6% and estimated out-of-sample error (means data not fitted model applied) is 0.3%