Practical Machine Learning Project

Erasemus

March 20, 2015

This document presents the project for the Coursera class “Practical Machine Learning” sponsored by the Bloomberg School af Health at the Johns Hopkins University and taught by Roger Peng, Brian Caffe and Jeffrey Leek.

This project analyzes data collected as decribed in:
Wearable Computing: Accelerometers’ Data Classification of Body Postures and Movements Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers’ Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6.

Read more: http://groupware.les.inf.puc-rio.br/work.jsf?p1=10335#ixzz3SLM8kNln (http://groupware.les.inf.puc-rio.br/work.jsf?p1=10335#ixzz3SLM8kNln)
This assignment challenges the student to develop a predictive model for the data which was collected during the performance of specific body postures and movements.
The approach used to develop a predictive model for predicting the exercise and to perform cross validation included the following steps:
1. load the training data
2. inspect and clean the data, eliminating fields with near zero variance and those with out correlation to the activity (identity of subject, time of measurement , etc. and those with low percentage of values that are not NA)
3. create a partition to cross validate our model for predicting the dependent variable “classe”
4. use random forest as the method for deriving a model, based on the results reported in the above cited work
5. determine the accuracy of the model via a confusionMatrix on the test data partition created in step 3
6. assess the out of sample error by using the model to predict a subset of data not subjected to step 2
Results: The random forest prediction model demonstrated accuracy in excess of 99% as determined by a confusion matrix generated against the cross validation data partition. The out of sample error is .3% . In addition the kappa value greater than .99 indicates a very high correlation of observed results to expected results. A plot of the random forest prediction model shows that the range of accuracy for the set of predictors chosen by the above methods is between 99% an 99.5% accuracy.

```
## Loading required package: lattice
## Loading required package: ggplot2
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
## Loading required package: bitops
## Loading required package: foreach
## Loading required package: iterators
## Loading required package: parallel
```

```
#load the training data
myCsv <- getURL("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")
temporaryFile <- tempfile()
con <- file(temporaryFile, open = "w")
cat(myCsv, file = con)
close(con)
df<-read.csv(temporaryFile)
```

```
#
# remove variables with little or no variability
#
df <- df[,-nearZeroVar(df)]
#
# Remove the columns that, by inspection, add no predictive value
# because they have identity data for tracking the subject and data collection
#
df <- subset(df, select=-c(X,user_name,raw_timestamp_part_1,raw_timestamp_part_2,cvtd_timestamp))
```

create the partitions for training and cross-validation

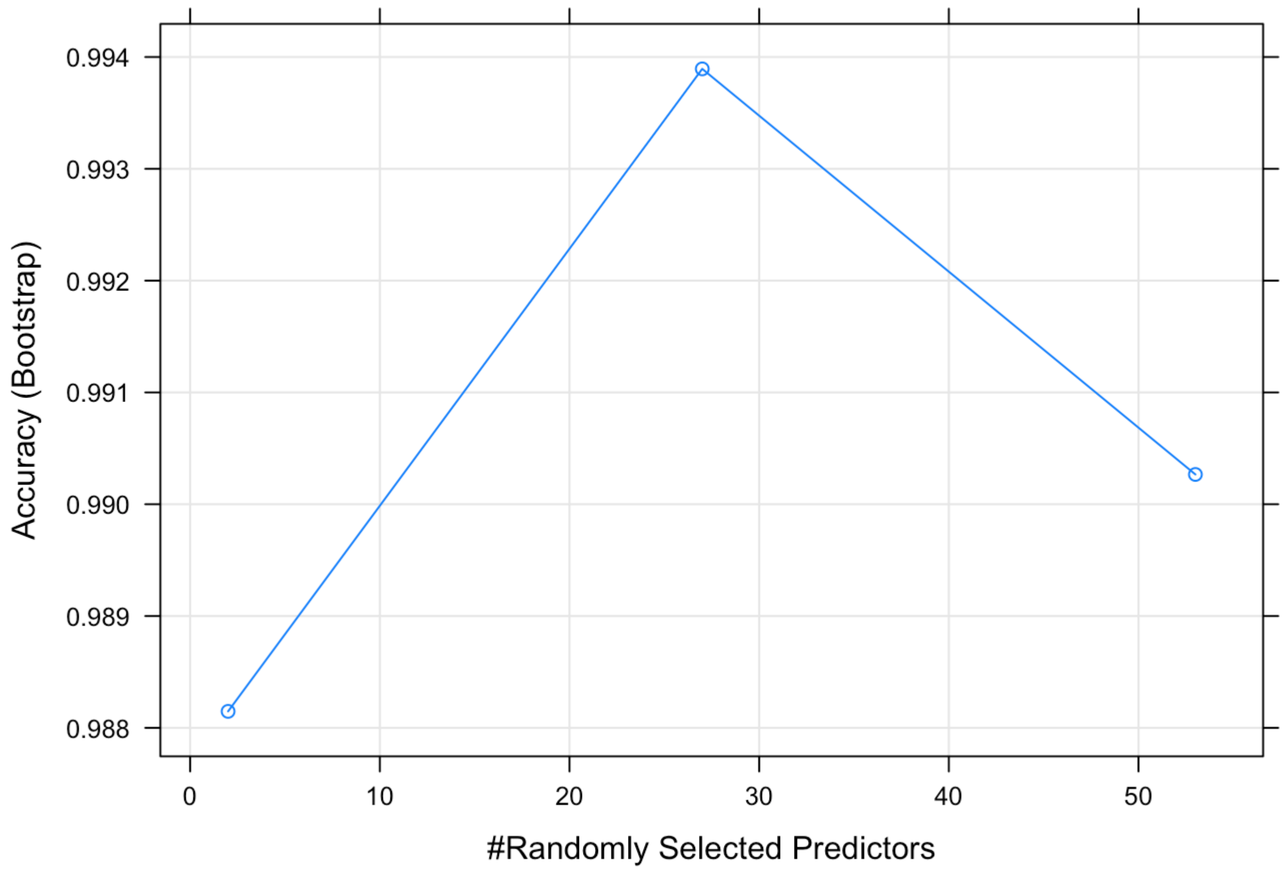
```
trainingRows <- createDataPartition(df$classe, p = 0.6, list = FALSE)
training <- df[trainingRows,]
CV <- df[-trainingRows,]
```

```
# remove variables with low density of information, those which are mostly NAs
# function for determining density of variables
den <- function(x) {
    n <- length(x)
    na.count <- sum(is.na(x))
    return((n - na.count)/n)
}

# density of input variables based on training subset
colDens <- apply(training, 2, den)
# remove the low density variables
fdf <- training[, colDens > .75]
```

```
# Using the training set above, train a model using RandomForest
modelFit<-train(classe ~ ., data=fdf, method="rf")
```

```
plot.train(modelFit)
```



```
preds<-predict(modelFit,CV)
# use confusion matrix with the cross validation partition to assess the accuracy and
# the sample error
rfCM<-confusionMatrix(preds,CV$classe)
sp <- varImp(modelFit,scale=FALSE)
#Look at the accuracy and sample error
rfCM$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.9969411      0.9961308      0.9954520      0.9980392      0.2844762
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

```
# and the out of sample errors
    sum(preds != CV$classe)/length(CV$classe)
```

```
## [1] 0.003058884
```