Project Quantified Self

Jonathan Schubert

2 July 2016

Motivation and data

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. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Goal

The goal of your project is to predict the manner in which they did 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.

Read data and split into training/testing set

```
library(caret)
set.seed(5555)
dat = read.csv("/Users/jonathanschubert/Documents/DataScience/Coursera/PracticalMachineLearning/Project/
indexTrain <- createDataPartition(dat$classe, p=0.66, list=F)
train <- dat[indexTrain, ]
test <- dat[-indexTrain, ]
```

The data set was split into a training and testing set, using 2/3 for training. All empty values in the raw data are interpreted as NA values.

Clean data set and remove meaningless predictors

```
validCols <- grepl("classe|belt|arm|dumbbell", names(train))
train <- train[, validCols]
train <- train[,colSums(is.na(train)) == 0]
dim(train)
```

```
## [1] 12953    53
```

As the data set contains several columns of no further value, they have been removed. Additionally all predictors containing any NA values have been removed. By doing this in total 53 predictors of originally 160 predictors have been used in the further analysis.

Model testing and evaluation based on test set

Four different classification models have been tested. On the one hand side a k-nearest neighbor approach was applied and on the other hand different tree based models were tested (simple tree, RandomForest, GradientBoosting). All classifiers have been trained using 3-fold cross validation.

```
control <- trainControl(method="cv", 3)
model_knn   <- train(classe~., method='knn',   trControl=control, data=train)
model_tree  <- train(classe~., method='rpart', trControl=control, data=train)
model_gbm   <- train(classe~., method='gbm',   trControl=control, data=train, verbose=FALSE)
model_rf    <- train(classe~., method="rf",    trControl=control, data=train, do.trace=FALSE, ntree=100)
```

Applying those models to our out of sample test set yield the following accurancy metrics:

```
prediction_tree <- predict(model_tree, newdata=test)
prediction_knn  <- predict(model_knn,  newdata=test)
prediction_gbm  <- predict(model_gbm,  newdata=test)
prediction_rf   <- predict(model_rf,   newdata=test)

print(paste("Accurancy Tree:", confusionMatrix(prediction_tree, test$classe)$overall[1]))
```

```
## [1] "Accurancy Tree: 0.496476233318339"
```

```
print(paste("Accurancy KNN:",  confusionMatrix(prediction_knn,  test$classe)$overall[1]))
```

```
## [1] "Accurancy KNN: 0.902234218023692"
```

```
print(paste("Accurancy GBM:",  confusionMatrix(prediction_gbm,  test$classe)$overall[1]))
```

```
## [1] "Accurancy GBM: 0.960713750187434"
```

```
print(paste("Accurancy RF:",   confusionMatrix(prediction_rf,   test$classe)$overall[1]))
```

```
## [1] "Accurancy RF: 0.989953516269306"
```

In the end the RandomForest approach seems to cope best with the pretty large number of predictors and gives a very good accurancy of abou 99%.

The expected out of sample error [%] is:

```
(1-as.numeric(confusionMatrix(prediction_rf,test$classe)$overall[1]))*100
```

```
## [1] 1.004648
```

Prediction on given verification set

As the RandomForest model worked best we are going to apply it as well to the verification data set, which contains 20 data points. The results are as following:

```
dat_pred = read.csv("/Users/jonathanschubert/Documents/DataScience/Coursera/PracticalMachineLearning/Pr
prediction_rf   <- predict(model_rf,   newdata=dat_pred)
prediction_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
```