Practical Machine Learning Project

Hitesh Tulsani

Monday, November 17, 2014

Predicting Quality of exercise activity

Research on activity recognition has traditionally focused on discriminating between different activities, i.e. to predict which" activity was performed at a specific point in time. The quality of executing an activity, the "how (well)", has only received little attention so far, even though it potentially provides useful information for a large variety of applicationsIn this project we will use supervised learning techniques to predict how well an activity is being performed, using the training and testing datasets of the Practical Machine Learning course, which come from the HAR project by Groupware@LES.

Getting the data The training dataset is available here and the testing dataset is available here.

```
set.seed(230583) ##required for reproducibility
library(caret) ##required for modelling
```

```
## Warning: package 'caret' was built under R version 3.1.2
```

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

```
##if files exists - don't download
if(!file.exists("pml-training.csv")){
    download.file("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", "pml-training.
}


if(!file.exists("pml-testing.csv")){
    download.file("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", "pml-testing.csv
}

## read the training data into data
data <- read.csv("pml-training.csv", na.strings = c("NA", "#DIV/0!"))
```

Preliminary data analysis The dataset consists in 19622 observations of 160 features. The goal is to predict the `classe` feature, which is a factor identifying the quality of the activity performance, with following meanings:

- Class A: exactly according to the specification
- Class B: throwing the elbows to the front
- Class C: lifting the dumbbell only halfway
- Class D: lowering the dumbbell only halfway and
- Class E: throwing the hips to the front

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. There are few variables in the dataset, which will lead to overfitting as these are not related to the quality of the performance. There are other measures which are just statistics such as avg, variance, sd etc, which must be removed:

```
data <- data[, -c(1:7)]
data <- data[, -grep("^avg|^var|^stddev|^amplitude|^min|^max|^skewness|^kurtosis", attr(data, "names"))]
```

The data is then divided into training and test sets. Post this, the correlated variables are removed from training set:

```
inTrain <- createDataPartition(y = data$classe, p = 0.7, list=FALSE)
training <- data[inTrain,]
testing <- data[-inTrain,]

##remove correlated variables
training <- training[,-findCorrelation(cor(training[,-53]))]
```

Predicting the error rate for random forests I chose Random Forests because with so many variables and the associated noise, random forest method with (OOB) cross-validation is more adequate.

```
## 5 out-of-bag resamples
tr <- trainControl(method="oob", number=5)

##Build a model with 50 trees
modelrf <- train(classe~., data=training, method="rf", trControl = tr, ntree=50)
```

```
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
modelrf$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, ntree = 50, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 23
##
##         OOB estimate of  error rate: 0.79%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3897    4    1    2    2 0.002304147
## B   21 2624   11    0    2 0.012791573
## C    0   20 2369    4    3 0.011268781
## D    0    0   21 2228    3 0.010657194
## E    0    2    7    5 2511 0.005544554
```

The estimated error rate with 50 trees and 5 OOB resamples is: 0.79%

Final model Final Model is a Random Forest with 1000 trees and 15 OOB resamples:

```
tr <- trainControl(method="oob", number=15)
model <- train(classe~., data=training, method="rf", trControl = tr, ntree=1000)

model
```

```
## Random Forest
##
## 13737 samples
##    45 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9918468  0.9896852
##   23    0.9932300  0.9914360
##   45    0.9844216  0.9802923
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 23.
```

```
model$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, ntree = 1000, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 23
##
##         OOB estimate of  error rate: 0.7%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3900    3    1    0    2 0.001536098
## B   20 2629    9    0    0 0.010910459
## C    0   18 2370    8    0 0.010851419
## D    0    0   21 2228    3 0.010657194
## E    0    0    3    8 2514 0.004356436
```

The estimated error rate is 0.7%

```
confusionMatrix(predict(model, testing), testing$classe)
```

Testing the Model

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1670   11    0    0    0
##          B    4 1124    7    0    0
##          C    0    3 1017    7    1
##          D    0    1    2  957    4
##          E    0    0    0    0 1077
##
## Overall Statistics
##
##                Accuracy : 0.9932
##                  95% CI : (0.9908, 0.9951)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9914
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9976   0.9868   0.9912   0.9927   0.9954
## Specificity            0.9974   0.9977   0.9977   0.9986   1.0000
## Pos Pred Value         0.9935   0.9903   0.9893   0.9927   1.0000
## Neg Pred Value         0.9990   0.9968   0.9981   0.9986   0.9990
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2838   0.1910   0.1728   0.1626   0.1830
## Detection Prevalence   0.2856   0.1929   0.1747   0.1638   0.1830
## Balanced Accuracy      0.9975   0.9923   0.9945   0.9957   0.9977
```

The error rate for the test dataset is approx 0.713%, being close to the model.

```
quizdata<-read.csv("pml-testing.csv",na.strings = c("NA", "#DIV/0!"))

predict(model, quizdata)
```

Applying the model to predict 20 test cases

```
##  [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
```

References

1. Quality Activity Recognition of Weigh Lifting Exercise
2. Original Paper