Courseproject Machine Learning

Heiko Lange

23 12 2017

```
# preload all libraries and prevent unecessary output messages
library(caret)
library(randomForest)
library(survival)
library(splines)
library(parallel)
library(gbm)
```

Course Project

Problem Statement

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.

Synopsis

For this prediction task, I will compare Random Forest prediction to Stochastic Gradient Boosting, two of the better and more popular prediction algorithms. Both models will be trained with cross validation and the better of the two models is chosen based on the estimated out of sample error. To get a better estimation of the real out of sample error, we will also run the better model against an unused part of the data to get an even better estimation for out of sample error. Last but not least, I will predict the outcome of the provided test set, which is also needed to complete the prediction task.

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. 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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Analysis

Downloading data

```
full_training <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"))
full_testing <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"))
```

Preprocessing

Many variables don't add information and others contain mostly NA values and aren't helpful either. Additionally there are some variables which don't seem to have a meaningful correlation with the result, like timestamps or activity windows. These will be filtered out as part of the preprocessing.

```
nvz <- nearZeroVar(full_training)
full_training <- full_training[,-nvz]
full_training <- full_training[,colSums(is.na(full_training)) / dim(full_training)[1] <= 0.5]
drop_cols <- c("X", "raw_timestamp_part_1", "raw_timestamp_part_2", "cvtd_timestamp", "new_window", "num_window")
full_training <- full_training[,!names(full_training) %in% drop_cols]
```

Data splitting

I will split the data in 70% training set and 30% validation set. Training set will be used to train two different models with cross validation. Thanks to cross validation, we can estimate the out of sample accuracy and select the better of the two models. The validation set will be used afterwards to compare the estimated out of sample accuracy with an (so far) unused validation set of data.

```
set.seed(23456)
inTrain <- createDataPartition(full_training[,1], p = 0.7, list = FALSE)
training_set <- full_training[inTrain,]
validation_set <- full_training[-inTrain,]
```

Training different models

I will train two of the most common and successful models, Random Forest and Gradient Boosting using carets build in cross validation. We will then select the best model and compare estimated out of sample accuracy to accuracy on the validation set of the data split.

```
trainingControl <- trainControl(method = "cv", number = 10)
fit_rf <- train(classe ~ ., data = training_set, trControl = trainingControl, method = "rf")
fit_gbm <- train(classe ~ ., data = training_set, trControl = trainingControl, method = "gbm", verbose = FALSE)
```

Estimating Out of Sample error

Based on the best results for both Random Forest and Stochastic Gradient Boosting, I will calculate a 95% confidence interval for the expected accuracy.

```
best_rf <- which.max(fit_rf$results$Accuracy)
best_gbm <- which.max(fit_gbm$results$Accuracy)
df <- data.frame(model = c("Random Forest", "Stachastic Gradient Boosting"),
                 mean = c(fit_rf$results$Accuracy[best_rf], fit_gbm$results$Accuracy[
best_gbm]),
                 lowCI = c(fit_rf$results$Accuracy[best_rf] + qnorm(0.025) * fit_rf$r
esults$AccuracySD[best_rf],
                           fit_gbm$results$Accuracy[best_gbm] + qnorm(0.025) * fit_gb
m$results$AccuracySD[best_gbm]),
                 highCI = c(fit_rf$results$Accuracy[best_rf] + qnorm(0.975) * fit_rf$
results$AccuracySD[best_rf],
                            fit_gbm$results$Accuracy[best_gbm] + qnorm(0.975) * fit_g
bm$results$AccuracySD[best_gbm]))
df
```

```
##                          model      mean     lowCI    highCI
## 1                Random Forest 0.9913377 0.9880083 0.9946670
## 2 Stachastic Gradient Boosting 0.9617838 0.9557303 0.9678372
```

Choosing a model

Best model will be chosen by estimated out of sample accuracy. Also I compare estimated out of sampel accuracy to accuracy on validation data set.

```
if (fit_rf$results$Accuracy[best_rf] >= fit_gbm$results$Accuracy[best_gbm]) {
    fit_final <- fit_rf; model_final <- "Random Forest"
} else {
    fit_final <- fit_gbm; model_final <- "Stochastic Gradient Boosting"
}
cm <- confusionMatrix(predict(fit_final, newdata = validation_set), validation_set$cl
asse)
df2 <- data.frame(model = model_final, estimatedOOSerror = fit_final$results$Accuracy
[best_rf], validationOOSerror = cm$overall[1])
print(df2)
```

```
##                  model estimatedOOSerror validationOOSerror
## Accuracy Random Forest         0.9913377          0.9925221
```

Prediction on Test Set for Quiz

To complete the course project, I also need to predict on the test set provided.

```
predict(fit_final, newdata = full_testing)
```

```
##  [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
```

Appendix

Sources

The data for this project come from this source:
http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har
(http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har)

Information for reproducibility

```
sessionInfo()
```

```
## R version 3.3.2 (2016-10-31)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: macOS Sierra 10.12.6
##
## locale:
## [1] de_DE.UTF-8/de_DE.UTF-8/de_DE.UTF-8/C/de_DE.UTF-8/de_DE.UTF-8
##
## attached base packages:
## [1] parallel  splines   stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
## [1] plyr_1.8.4          gbm_2.1.3           survival_2.41-3
## [4] randomForest_4.6-12 caret_6.0-77        ggplot2_2.2.1
## [7] lattice_0.20-34
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.3   purrr_0.2.2        reshape2_1.4.2
##  [4] kernlab_0.9-25     colorspace_1.3-2   stats4_3.3.2
##  [7] htmltools_0.3.6    yaml_2.1.14        prodlim_1.6.1
## [10] rlang_0.1.4        e1071_1.6-8        ModelMetrics_1.1.0
## [13] withr_2.1.0        glue_1.2.0         foreign_0.8-67
## [16] DBI_0.5-1          foreach_1.4.3      dimRed_0.1.0
## [19] lava_1.5.1         robustbase_0.92-7  stringr_1.1.0
## [22] timeDate_3042.101  munsell_0.4.3      gtable_0.2.0
## [25] recipes_0.1.1      codetools_0.2-15   psych_1.7.3.21
## [28] evaluate_0.10      knitr_1.16         class_7.3-14
## [31] DEoptimR_1.0-8     broom_0.4.2        Rcpp_0.12.9
## [34] scales_0.4.1       backports_1.1.0    ipred_0.9-6
## [37] CVST_0.2-1         mnormt_1.5-5       digest_0.6.12
## [40] stringi_1.1.2      dplyr_0.5.0        RcppRoll_0.2.2
## [43] ddalpha_1.3.1      grid_3.3.2         rprojroot_1.2
## [46] tools_3.3.2        magrittr_1.5       lazyeval_0.2.0
## [49] tibble_1.2         tidyr_0.6.1        DRR_0.0.2
## [52] MASS_7.3-45        Matrix_1.2-7.1     lubridate_1.6.0
## [55] gower_0.1.2        assertthat_0.1     rmarkdown_1.5
## [58] iterators_1.0.8    R6_2.2.0           rpart_4.1-10
## [61] compiler_3.3.2     sfsmisc_1.1-1      nnet_7.3-12
## [64] nlme_3.1-128
```