[Skip to content](https://github.com/benjamin-chan/PracticalMachineLearning/blob/master/predictionAssignment.Rmd" \l "start-of-content)

Top of Form



This repository



Bottom of Form

* [Pull requests](https://github.com/pulls)
* [Issues](https://github.com/issues)
* [**Gist**](https://gist.github.com/)
* [New repository](https://github.com/new) [New organization](https://github.com/organizations/new)

This repository

[New issue](https://github.com/benjamin-chan/PracticalMachineLearning/issues/new)

* **[@renehuang](https://github.com/renehuang)**

Signed in as **renehuang**

[Your profile](https://github.com/renehuang) [Your stars](https://github.com/stars) [Explore](https://github.com/explore) [Help](https://help.github.com/)

[Settings](https://github.com/settings/profile)

Top of Form



Sign out

Bottom of Form

Top of Form





[**Watch**](https://github.com/benjamin-chan/PracticalMachineLearning/subscription) [**2**](https://github.com/benjamin-chan/PracticalMachineLearning/watchers)

**Notifications**

**Not watching**Be notified when participating or @mentioned. Watch

**Watching**Be notified of all conversations. Unwatch

**Ignoring**Never be notified. Stop ignoring

Bottom of Form

Top of Form



Unstar [**2**](https://github.com/benjamin-chan/PracticalMachineLearning/stargazers)

Bottom of Form

Top of Form



Star [**2**](https://github.com/benjamin-chan/PracticalMachineLearning/stargazers)

Bottom of Form

Top of Form



Fork

Bottom of Form

[**28**](https://github.com/benjamin-chan/PracticalMachineLearning/network)

[**benjamin-chan**](https://github.com/benjamin-chan)**/[PracticalMachineLearning](https://github.com/benjamin-chan/PracticalMachineLearning) https://assets-cdn.github.com/images/spinners/octocat-spinner-32.gif**

* [Code [https://assets-cdn.github.com/images/spinners/octocat-spinner-32.gif](https://github.com/benjamin-chan/PracticalMachineLearning)](https://github.com/benjamin-chan/PracticalMachineLearning)
* [Issues **0** [https://assets-cdn.github.com/images/spinners/octocat-spinner-32.gif](https://github.com/benjamin-chan/PracticalMachineLearning/issues)](https://github.com/benjamin-chan/PracticalMachineLearning/issues)
* [Pull requests **0** [https://assets-cdn.github.com/images/spinners/octocat-spinner-32.gif](https://github.com/benjamin-chan/PracticalMachineLearning/pulls)](https://github.com/benjamin-chan/PracticalMachineLearning/pulls)
* [Wiki [https://assets-cdn.github.com/images/spinners/octocat-spinner-32.gif](https://github.com/benjamin-chan/PracticalMachineLearning/wiki)](https://github.com/benjamin-chan/PracticalMachineLearning/wiki)
* [Pulse [https://assets-cdn.github.com/images/spinners/octocat-spinner-32.gif](https://github.com/benjamin-chan/PracticalMachineLearning/pulse)](https://github.com/benjamin-chan/PracticalMachineLearning/pulse)
* [Graphs [https://assets-cdn.github.com/images/spinners/octocat-spinner-32.gif](https://github.com/benjamin-chan/PracticalMachineLearning/graphs)](https://github.com/benjamin-chan/PracticalMachineLearning/graphs)

**HTTPS clone URL**



**SSH clone URL**



**Subversion checkout URL**



You can clone with

Top of Form



HTTPS

Bottom of Form

,

Top of Form



SSH

Bottom of Form

, or

Top of Form



Subversion

Bottom of Form

.

[**Clone in Desktop**](https://windows.github.com/) [**Download ZIP**](https://github.com/benjamin-chan/PracticalMachineLearning/archive/master.zip)

[Permalink](https://github.com/benjamin-chan/PracticalMachineLearning/blob/f60309ea00885c32bb87d15c057bc87c0af1c101/predictionAssignment.Rmd)

***Branch:* master**

**Switch branches/tags**



* [**Branches**](https://github.com/benjamin-chan/PracticalMachineLearning/blob/master/predictionAssignment.Rmd)
* [**Tags**](https://github.com/benjamin-chan/PracticalMachineLearning/blob/master/predictionAssignment.Rmd)

[Velloso](https://github.com/benjamin-chan/PracticalMachineLearning/blob/Velloso/predictionAssignment.Rmd) [gh-pages](https://github.com/benjamin-chan/PracticalMachineLearning/blob/gh-pages/predictionAssignment.Rmd) [master](https://github.com/benjamin-chan/PracticalMachineLearning/blob/master/predictionAssignment.Rmd) [test](https://github.com/benjamin-chan/PracticalMachineLearning/blob/test/predictionAssignment.Rmd)

Nothing to show

Nothing to show

[PracticalMachineLearning](https://github.com/benjamin-chan/PracticalMachineLearning)/**predictionAssignment.Rmd**

@benjamin-chan[benjamin-chan](https://github.com/benjamin-chan) on Oct 23 2014

[Add some notes](https://github.com/benjamin-chan/PracticalMachineLearning/commit/ffd0eff87333ea5fd475bcc06d7c0a9d4ab20b41)

[**1** contributor](https://github.com/benjamin-chan/PracticalMachineLearning/blob/master/predictionAssignment.Rmd#blob_contributors_box)

Users who have contributed to this file

* **@benjamin-chan**[**benjamin-chan**](https://github.com/benjamin-chan)

[**Raw**](https://github.com/benjamin-chan/PracticalMachineLearning/raw/master/predictionAssignment.Rmd) [**Blame**](https://github.com/benjamin-chan/PracticalMachineLearning/blame/master/predictionAssignment.Rmd) [**History**](https://github.com/benjamin-chan/PracticalMachineLearning/commits/master/predictionAssignment.Rmd)

Top of Form



Bottom of Form

Top of Form



Bottom of Form

250 lines (183 sloc) 8 KB

| **title** | **author** | **output** |
| --- | --- | --- |
| Coursera: Practical Machine Learning Prediction Assignment | Benjamin Chan [GitHub](<https://github.com/benjamin-chan>) | | **html\_document** | | --- | | | **keep\_md** | **toc** | | --- | --- | | true | true | | |

message(sprintf("Run time: %s\nR version: %s", Sys.time(), R.Version()$version.string))

**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://groupware.les.inf.puc-rio.br/har> (see the section on the Weight Lifting Exercise Dataset).

\**Data \**

The training data for this project are available here:

<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv>

The test data are available here:

<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv>

The data for this project come from this source: <http://groupware.les.inf.puc-rio.br/har>. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

**What you should submit**

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.

1. Your submission should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-).
2. You should also apply your machine learning algorithm to the 20 test cases available in the test data above. Please submit your predictions in appropriate format to the programming assignment for automated grading. See the programming assignment for additional details.

\**Reproducibility \**

Due to security concerns with the exchange of R code, your code will not be run during the evaluation by your classmates. Please be sure that if they download the repo, they will be able to view the compiled HTML version of your analysis.

**Prepare the datasets**

Read the training data into a data table.

require(data.table)

setInternet2(TRUE)

url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"

D <- fread(url)

Read the testing data into a data table.

url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

DTest <- fread(url)

Which variables in the test dataset have zero NAs? Use this tip: [finding columns with all missing values in r](http://stackoverflow.com/a/11330265).

Belt, arm, dumbbell, and forearm variables that do not have any missing values in the test dataset will be **predictor candidates**.

isAnyMissing <- sapply(DTest, function (x) any(is.na(x) | x == ""))

isPredictor <- !isAnyMissing & grepl("belt|[^(fore)]arm|dumbbell|forearm", names(isAnyMissing))

predCandidates <- names(isAnyMissing)[isPredictor]

predCandidates

Subset the primary dataset to include only the **predictor candidates** and the outcome variable, classe.

varToInclude <- c("classe", predCandidates)

D <- D[, varToInclude, with=FALSE]

dim(D)

names(D)

Make classe into a factor.

D <- D[, classe := factor(D[, classe])]

D[, .N, classe]

Split the dataset into a 60% training and 40% probing dataset.

require(caret)

seed <- as.numeric(as.Date("2014-10-26"))

set.seed(seed)

inTrain <- createDataPartition(D$classe, p=0.6)

DTrain <- D[inTrain[[1]]]

DProbe <- D[-inTrain[[1]]]

Preprocess the prediction variables by centering and scaling.

X <- DTrain[, predCandidates, with=FALSE]

preProc <- preProcess(X)

preProc

XCS <- predict(preProc, X)

DTrainCS <- data.table(data.frame(classe = DTrain[, classe], XCS))

Apply the centering and scaling to the probing dataset.

X <- DProbe[, predCandidates, with=FALSE]

XCS <- predict(preProc, X)

DProbeCS <- data.table(data.frame(classe = DProbe[, classe], XCS))

Check for near zero variance.

nzv <- nearZeroVar(DTrainCS, saveMetrics=TRUE)

if (any(nzv$nzv)) nzv else message("No variables with near zero variance")

Examine groups of prediction variables.

histGroup <- function (data, regex) {

col <- grep(regex, names(data))

col <- c(col, which(names(data) == "classe"))

require(reshape2)

n <- nrow(data)

DMelted <- melt(data[, col, with=FALSE][, rownum := seq(1, n)], id.vars=c("rownum", "classe"))

require(ggplot2)

ggplot(DMelted, aes(x=classe, y=value)) +

geom\_violin(aes(color=classe, fill=classe), alpha=1/2) +

# geom\_jitter(aes(color=classe, fill=classe), alpha=1/10) +

# geom\_smooth(aes(group=1), method="gam", color="black", alpha=1/2, size=2) +

facet\_wrap(~ variable, scale="free\_y") +

scale\_color\_brewer(palette="Spectral") +

scale\_fill\_brewer(palette="Spectral") +

labs(x="", y="") +

theme(legend.position="none")

}

histGroup(DTrainCS, "belt")

histGroup(DTrainCS, "[^(fore)]arm")

histGroup(DTrainCS, "dumbbell")

histGroup(DTrainCS, "forearm")

**Train a prediction model**

Using random forest, the out of sample error should be small. The error will be estimated using the 40% probing sample. I would be quite happy with an error estimate of 3% or less.

Set up the parallel clusters.

require(parallel)

require(doParallel)

cl <- makeCluster(detectCores() - 1)

registerDoParallel(cl)

Set the control parameters.

ctrl <- trainControl(classProbs=TRUE,

savePredictions=TRUE,

allowParallel=TRUE)

Fit model over the tuning parameters.

method <- "rf"

system.time(trainingModel <- train(classe ~ ., data=DTrainCS, method=method))

Stop the clusters.

stopCluster(cl)

**Evaluate the model on the training dataset**

trainingModel

hat <- predict(trainingModel, DTrainCS)

confusionMatrix(hat, DTrain[, classe])

**Evaluate the model on the probing dataset**

hat <- predict(trainingModel, DProbeCS)

confusionMatrix(hat, DProbeCS[, classe])

**Display the final model**

varImp(trainingModel)

trainingModel$finalModel

**The estimated error rate is less than 1%.**

Save training model object for later.

save(trainingModel, file="trainingModel.RData")

**Predict on the test data**

Load the training model.

load(file="trainingModel.RData", verbose=TRUE)

Get predictions and evaluate.

DTestCS <- predict(preProc, DTest[, predCandidates, with=FALSE])

hat <- predict(trainingModel, DTestCS)

DTest <- cbind(hat , DTest)

subset(DTest, select=names(DTest)[grep("belt|[^(fore)]arm|dumbbell|forearm", names(DTest), invert=TRUE)])

**Submission to Coursera**

Write submission files to predictionAssignment\_files/answers.

pml\_write\_files = function(x){

n = length(x)

path <- "predictionAssignment\_files/answers"

for(i in 1:n){

filename = paste0("problem\_id\_",i,".txt")

write.table(x[i],file=file.path(path, filename),quote=FALSE,row.names=FALSE,col.names=FALSE)

}

}

pml\_write\_files(hat)

[Jump to Line](https://github.com/benjamin-chan/PracticalMachineLearning/blob/master/predictionAssignment.Rmd#jump-to-line)

Top of Form



Go

Bottom of Form

* [Status](https://status.github.com/)
* [API](https://developer.github.com/)
* [Training](https://training.github.com/)
* [Shop](https://shop.github.com/)
* [Blog](https://github.com/blog)
* [About](https://github.com/about)
* [Pricing](https://github.com/pricing)
* © 2015 GitHub, Inc.
* [Terms](https://github.com/site/terms)
* [Privacy](https://github.com/site/privacy)
* [Security](https://github.com/security)
* [Contact](https://github.com/contact)
* [Help](https://help.github.com/)

Something went wrong with that request. Please try again.

You signed in with another tab or window. Reload to refresh your session. You signed out in another tab or window. Reload to refresh your session.