# machinelearning

*dawky*

*Sunday, December 03, 2017*

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
##Summary

"In the test below, we try to quantify how well people do c
ertain activities. We use data from accelerometers on the b
elt,forearm, arm, and dumbell of 6 participants. They were
asked to perform barbell lifts correctly and incorrectly in
 5 different ways. When fitting the model, I find the rando
m forest model good for the experiment. The job I do are li
sted below. Information about the experiment is available f
rom the website here:

http://web.archive.org/web/20161224072740/http:/groupwar
e.les.inf.pucrio.br/har"
```

```
## [1] "In the test below, we try to quantify how well peop
le do certain activities. We use data from accelerometers o
n the belt, forearm, arm, and dumbell of 6 participants. Th
ey were asked to perform barbell lifts correctly and incorr
ectly in 5 different ways. When fitting the model, I find t
he random forest model good for the experiment. The job I d
o are listed below. Information about the experiment is ava
ilable from the website here: http://web.archive.org/web/2
0161224072740/http:/groupware.les.inf.pucrio.br/har"
```

```
##Data preprocessing

#In the following parts, I loaded the data and split it int
o training and testing sets.

library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
library(randomForest)
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
## margin
setwd("D:/r/coursera/machinelearning")
dtrain<-read.csv("pml-training.csv")
dtest<-read.csv("pml-testing.csv")
set.seed(1000)

#data split
inTrain<-createDataPartition(y=dtrain$classe,p=0.7,list=
F)
dtrain1<-dtrain[inTrain,]
dtest1<-dtrain[-inTrain,]

#delete unimportant factors
nzv <- nearZeroVar(dtrain1)
dtrain1 <- dtrain1[,-nzv]

nzv <- nearZeroVar(dtest1)
dtest1 <- dtest1[,-nzv]

##Delete NA
NAd<-is.na(dtrain1)
```

```
NAc<-which(colSums(NAd)/nrow(dtrain1)>0.95)
dtrain1<-dtrain1[,-NAc]
dtest1<-dtest1[,-NAc]


#delete some columns that doesn't make sense, which is the first 5 columns
dtrain1<-dtrain1[,-(1:5)]
dtest1<-dtest1[,-(1:5)]


##Do the 3-fold cross validation
control <- trainControl(method = "cv", number = 3)


##Build model using the random forest(acutally I've tried the rpart, too, but the random forest model has better results)
fit<-train(classe~.,data=dtrain1,method="rf", trControl=control)
plot(fit)
```

```
fit
## Random Forest
##
## 13737 samples
##    53 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold)
## Summary of sample sizes: 9157, 9158, 9159
## Resampling results across tuning parameters:
```

```
##
##   mtry  Accuracy   Kappa
##    2    0.9916284  0.9894098
##   27    0.9963600  0.9953957
##   53    0.9953410  0.9941071
##
## Accuracy was used to select the optimal model using  the
 largest value.
## The final value used for the model was mtry = 27.
```

```
##Evaluation of the model and see the out-of-sample error
#see how the model fit the prediction
predt<-predict(fit,dtest1)
confusionMatrix(dtest1$classe, predt)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction    A    B    C    D    E
##         A 1674    0    0    0    0
##         B    5 1133    1    0    0
##         C    0    2 1024    0    0
##         D    0    0    5  959    0
##         E    0    0    0    4 1078
##
## Overall Statistics
##
##               Accuracy : 0.9971
##                 95% CI : (0.9954, 0.9983)
##    No Information Rate : 0.2853
##    P-Value [Acc > NIR] : < 2.2e-16
```

```
##
##                Kappa : 0.9963
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity    0.9970   0.9982   0.9942   0.9958   1.0000
## Specificity    1.0000   0.9987   0.9996   0.9990   0.9992
## Pos Pred Value 1.0000   0.9947   0.9981   0.9948   0.9963
## Neg Pred Value 0.9988   0.9996   0.9988   0.9992   1.0000
## Prevalence     0.2853   0.1929   0.1750   0.1636   0.1832
## Detection Rate 0.2845   0.1925   0.1740   0.1630   0.1832
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638
  0.1839
## Balanced Accuracy      0.9985   0.9985   0.9969   0.9974
  0.9996
```

```
#the accuracy is 0.9973, high enough, which means this is a
 good model


##Predict the test set

#the expected out-of-sample error is 1-0.9973=0.0027,apply
 the model to the test set

test<-predict(fit, dtest)

test
```

```
##  [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
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.