Coursera Practical Machine Learning Assignment

Weight Lifting Activity Quality Prediction Model

Author: S. Wu

Project Goal

Use the “Weight Lifting Exercises Dataset” to investigate and predict the outcome variable “classe” - “how (well)” an activity was performed by the participants.

Data Source

Assignment Datasets

The datasets can be obtained from the course web site, including training data (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv) for writeup, and test data (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv) for submission. They come in the form of comma-separated-value files.

file name: “pml-training.csv”, “pml-testing.csv”

Original Data Source

The data for this project come from (http://groupware.les.inf.puc-rio.br/har)Groupware@LES (mailto:Groupware@LES) HAR Project. The paper Qualitative Activity Recognition of Weight Lifting Exercises (http://groupware.les.inf.puc-rio.br/public/papers/2013.Velloso.QAR-WLE.pdf) provides insight of the project and data features.

```
# Check and load required R packages
pkg<-c("knitr", "caret", "randomForest", "ggplot2", "gridExtra")
pkgCheck<-pkg %in% rownames(installed.packages())
for(i in 1:length(pkg)) {
    if(pkgCheck[i]==FALSE) {
        install.packages(pkg[i])
    }
    library(pkg[i],character.only = TRUE)
}
```

1. Data Loading and Cleaning

1.1 Load Data

```
if (file.exists("pml-train.csv")==FALSE) {
  url<- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
  download.file(url, "pml-train.csv")
  }
rawTrain<- read.csv("pml-train.csv", na.strings=c("NA", "", "#DIV/0!"))
```

```
#set table and figure counter
tnum<- 0L
fnum<- 0L
```

1.2 Check Data Quality and Features

```
str(rawTrain)
#Results are hidden due to long length
```

The dataset contains 160 variables(columns). It is observed that:

1. first two columns are row index and participants id, should be excluded.
2. many variables contain mostly NA values, and should be excluded.
3. columns 3~7 are time/window related, while the rest are sensor measurement related.

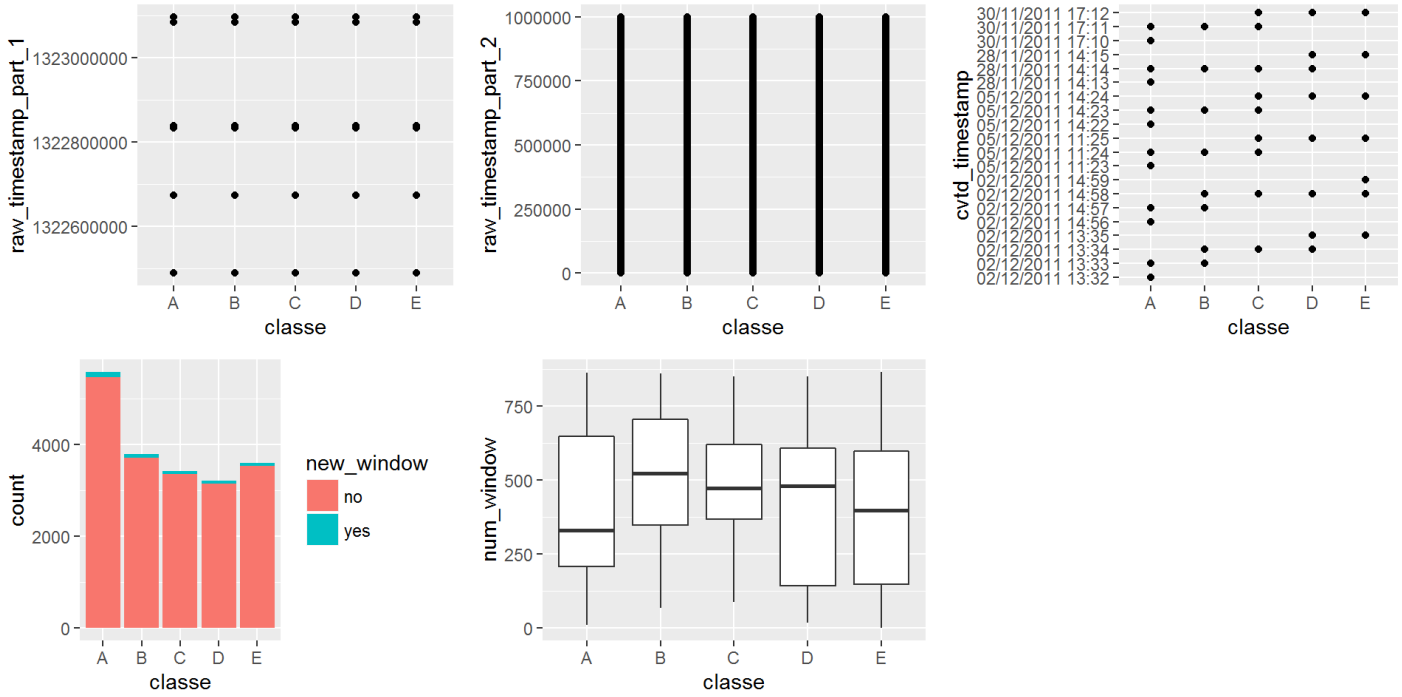

Figure 1 Outcome “classe” by Time Related Variables

Figure 1 suggests that time variables and the binary “new_window” may not be useful predictors for outcome “classe”.

```
# remove variables and produce dataset "dat" for prediction model fitting
NAs<- which(colSums(is.na(rawTrain)) > 0)
dat<- rawTrain[, -c(1:6, NAs)]
```

A clean dataset “dat” is produced with 54 potential predictors.

2. Data Partition

Split the cleaned training data(dataset “dat”) into 70% “train set” and 30% “test set”.

Note: The assignment comes with “training data”(pml-train.csv) for writeup and “test data”(pml-test.csv) for submission. The “training data” is cleaned and then split internally into a train set for model fitting and a test set for validation.

To avoid confusion, in this report, “train set” and “test set” refer to the internal datasets split from the training data for machine learning, while “submission test cases” refers to the 20 test cases from pml-test.csv for the submission assignment.

```
set.seed(12345)
inTrain<- createDataPartition(y=dat$classe, p=0.7, list=FALSE)
trainSet<- dat[inTrain,]
testSet<- dat[-inTrain,]
rbind(trainSet=dim(trainSet), testSet=dim(testSet))
```

```
##          [,1] [,2]
## trainSet 13737   54
## testSet   5885   54
```

The train set has 13737 samples, whereas the test set has 5885 samples.

3. Building Prediction Model

Try random forest and decision tree methods to predict the categorical outcome “classe”.

3.1 Random Forest

```
set.seed(12345)
# cross validation with out-of-bag (oob)
fitRF<- randomForest(classe ~. , data=trainSet, importance=TRUE)
fitRF
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = trainSet, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.32%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    2    0    0    0 0.0005120328
## B    4 2651    3    0    0 0.0026335591
## C    0   10 2384    2    0 0.0050083472
## D    0    0   15 2235    2 0.0075488455
## E    0    0    0    6 2519 0.0023762376
```

3.2 Classification and Regression Trees - rpart

```
set.seed(12345)
# set cross validation = 10-folds cross validation
trCtrl<- trainControl(method = "cv", number=10)
fitRpart<- train(classe ~., data=trainSet, method="rpart", trControl=trCtrl)
fitRpart
```

```
## CART
##
## 13737 samples
##    53 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12361, 12364, 12363, 12364, 12364, 12365, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa       Accuracy SD  Kappa SD
##   0.03794121  0.5685390  0.45143177  0.02635675   0.03973272
##   0.05538602  0.4208606  0.21558761  0.06803729   0.11263014
##   0.11585800  0.3157822  0.04813822  0.04049296   0.06217433
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was cp = 0.03794121.
```

3.3 Classification and Regression Trees - ctree

```
set.seed(12345)
# set cross validation = 10-folds cross validation
trCtrl<- trainControl(method = "cv", number=10)
fitCtree<- train(classe ~., data=trainSet, method="ctree", trControl=trCtrl)
fitCtree
```

```
## Conditional Inference Tree
##
## 13737 samples
##    53 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12361, 12364, 12363, 12364, 12364, 12365, ...
## Resampling results across tuning parameters:
##
##   mincriterion  Accuracy   Kappa      Accuracy SD  Kappa SD
##   0.01          0.8921878  0.8635379  0.01007613   0.01279215
##   0.50          0.8914596  0.8626211  0.01009080   0.01279879
##   0.99          0.8831603  0.8521459  0.01222897   0.01549620
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mincriterion = 0.01.
```

3.4 Model Selection

```
tnum=tnum+1
accuracy<- c(1-0.0032, max(fitRpart$results$Accuracy), max(fitCtree$results$Accuracy))
ise<- 1-accuracy
kable(data.frame(Model=c("random forest","rpart","ctree"),
                         accuracy=paste0(format(accuracy*100, digit=4),"%"),
                         ise=paste0(format(ise*100, digit=4),"%")),
                         col.names=c("Model","Accuracy", "In Sample Error"), row.names=F)
```

Model	Accuracy	In Sample Error
random forest	99.68%	0.32%
rpart	56.85%	43.15%
ctree	89.22%	10.78%

Table 1. Prediction Model Accuracy

The ***random forest*** algorithm, with out-of-bag (oob) cross-validation, produces the highest accuracy, thus is chosen as the final prediction model.

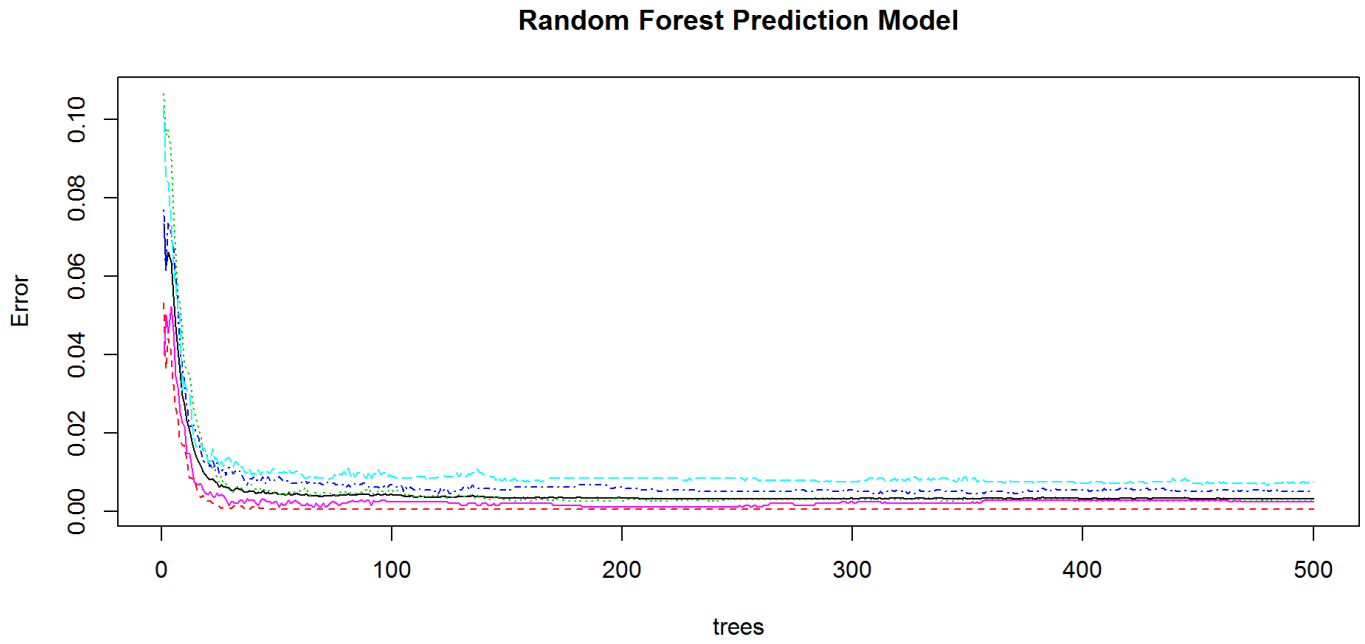

Figure 2 Random Forest Prediction Model

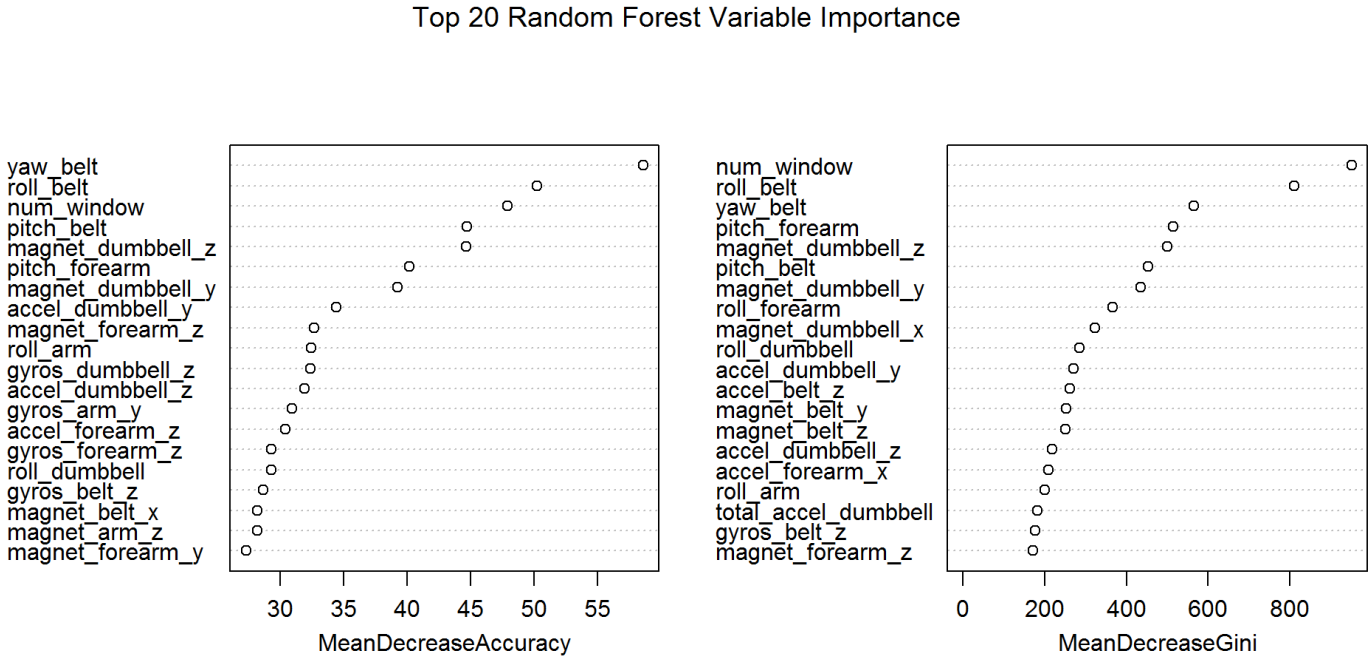

Figure 3 Random Forest Prediction Model Top 20 Variable Importance

4. Test Set Prediction and Expected Out of Sample Error

```
val<- confusionMatrix(testSet$classe, predict(fitRF, newdata=testSet))
val
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    7 1131    1    0    0
##          C    0    6 1020    0    0
##          D    0    0   12  952    0
##          E    0    0    0    3 1079
##
## Overall Statistics
##
##                Accuracy : 0.9951
##                  95% CI : (0.9929, 0.9967)
##     No Information Rate : 0.2856
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9938
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9958   0.9947   0.9874   0.9969   1.0000
## Specificity            1.0000   0.9983   0.9988   0.9976   0.9994
## Pos Pred Value         1.0000   0.9930   0.9942   0.9876   0.9972
## Neg Pred Value         0.9983   0.9987   0.9973   0.9994   1.0000
## Prevalence             0.2856   0.1932   0.1755   0.1623   0.1833
## Detection Rate         0.2845   0.1922   0.1733   0.1618   0.1833
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9979   0.9965   0.9931   0.9972   0.9997
```

Predicted with the **random forest** model with out-of-bag cross-validation, the **estimated out of sample error is 0.49%** (= 1 - Accuracy 99.51%).

5. Submission Test Cases Prediction

```
if (file.exists("pml-test.csv")==FALSE) {
  url<- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
  download.file(url, "pml-test.csv")
  }
testCases<- read.csv("pml-test.csv", na.strings=c("NA", "", "#DIV/0!"))
submissionPred<- predict(fitRF, newdata=testCases)
submissionPred
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```