Prediction Assignment Writeup

Viktor Z

March 20, 2016

Executive Summary

Random Forest model provides extremely high accuracy when solving classification problem how well person performs sport activity.

Data

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

Loading data

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

```
training <- read.csv("pml-training.csv", stringsAsFactors = FALSE)
testing <- read.csv("pml-testing.csv", stringsAsFactors = FALSE)
```

Our model shouldn't be dependent on time and user attributes, so lets skip them. Also set classe attribute to be factor, and all other variables as numeric.

```
training <- subset(training, select = -c(X, user_name, raw_timestamp_part_1, raw_t
imestamp_part_2, cvtd_timestamp, new_window, num_window))
training$classe <- factor(training$classe)
for(i in 1:152) training[,i] <- as.numeric(training[,i])

testing <- subset(testing, select = -c(X, user_name, raw_timestamp_part_1, raw_tim
estamp_part_2, cvtd_timestamp, new_window, num_window))
for(i in 1:152) testing[,i] <- as.numeric(testing[,i])
```

Data partitioning

Because there is no *classe* in testing data, I split training data to three training and testing.

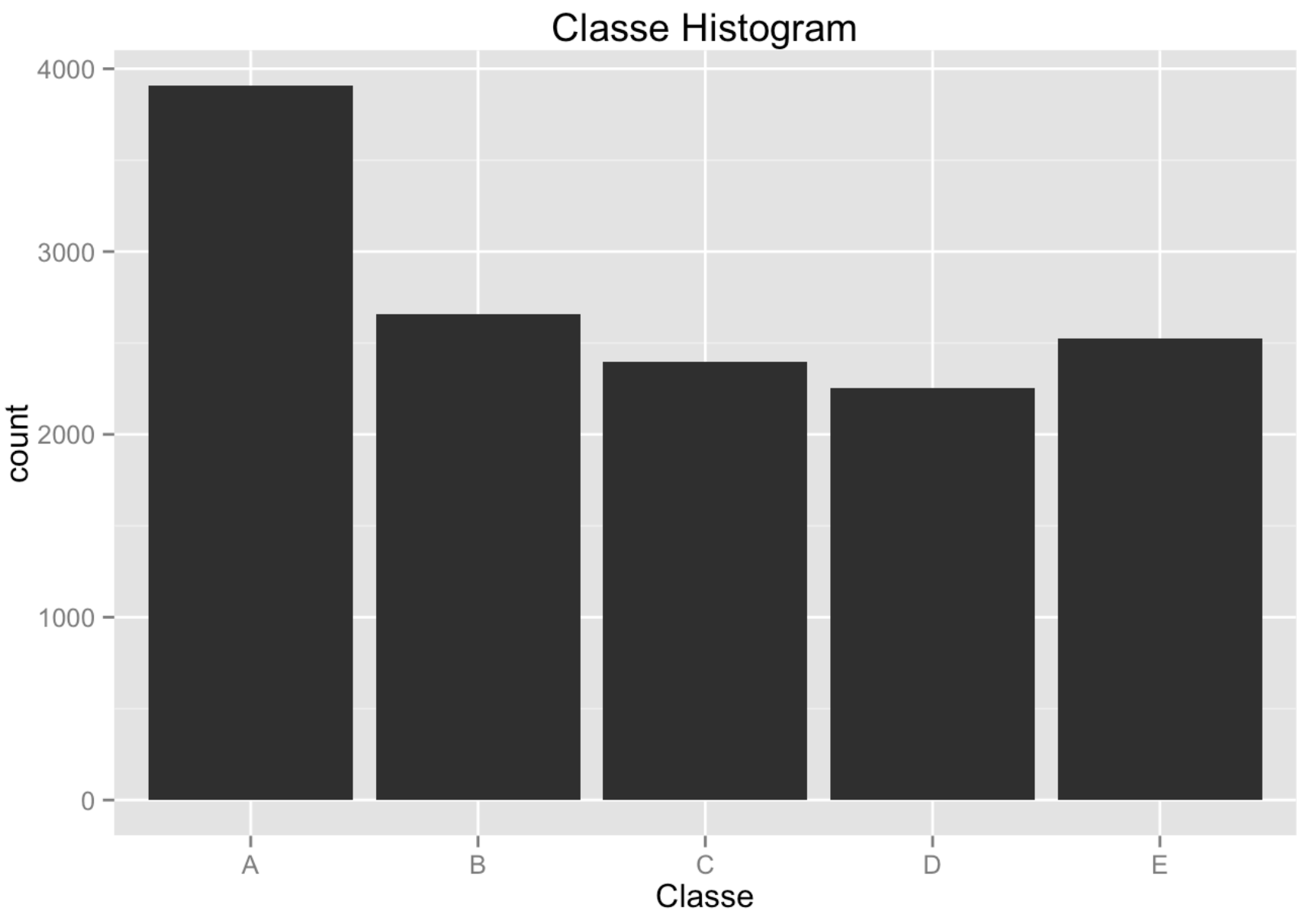
```
indexes <- createDataPartition(training$classe, p = 0.7, list = F)
subsetTesting <- training[-indexes, ]
subsetTraining <- training[indexes, ]
data.frame(training = dim(subsetTraining), testing = dim(subsetTesting))
```

```
##   training testing
## 1    13737    5885
## 2      153     153
```

Exploration

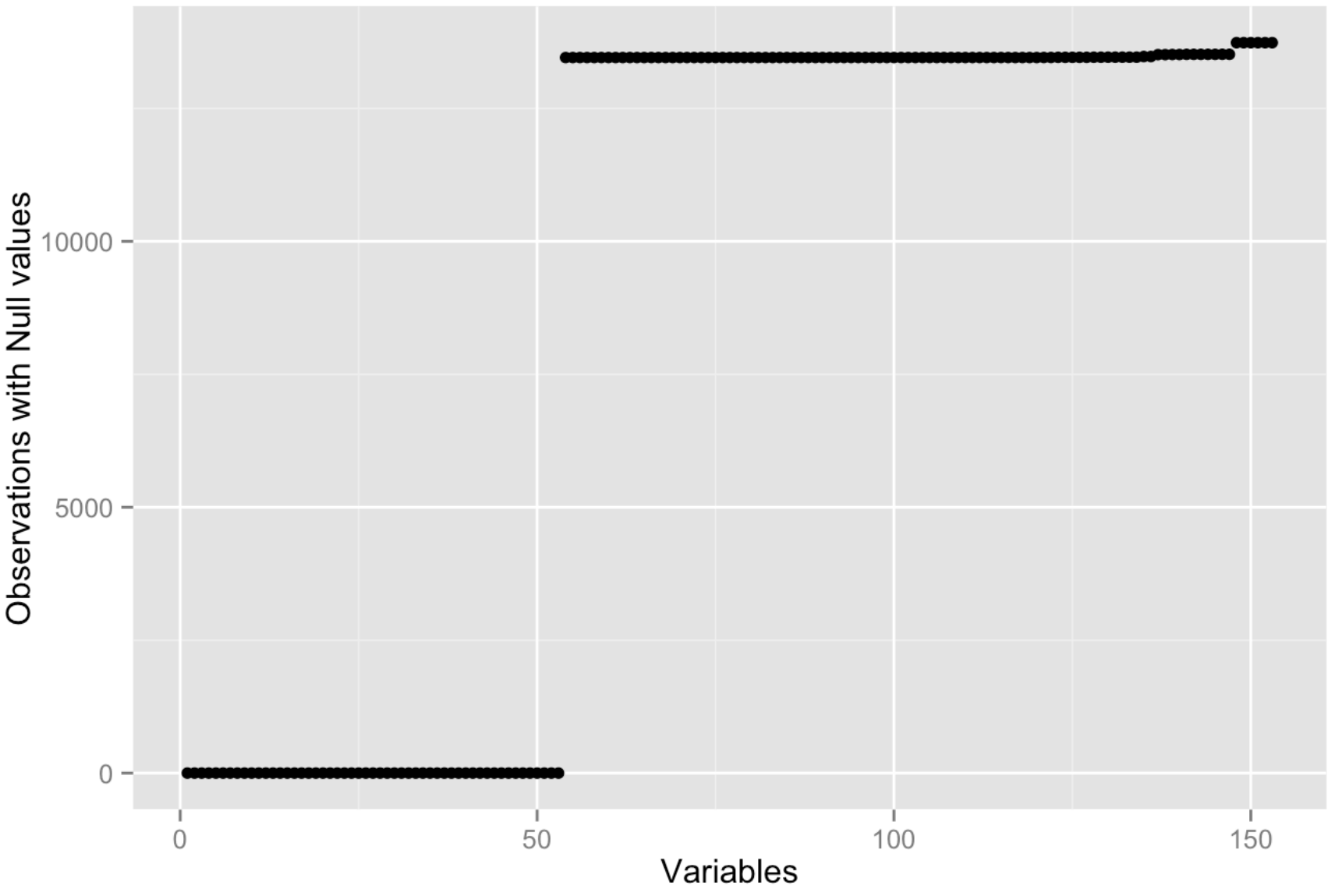
All classes have enough observations, I can use simple accuracy metric.

```
qplot(subsetTraining$classe, xlab = "Classe", main = "Classe Histogram")
```



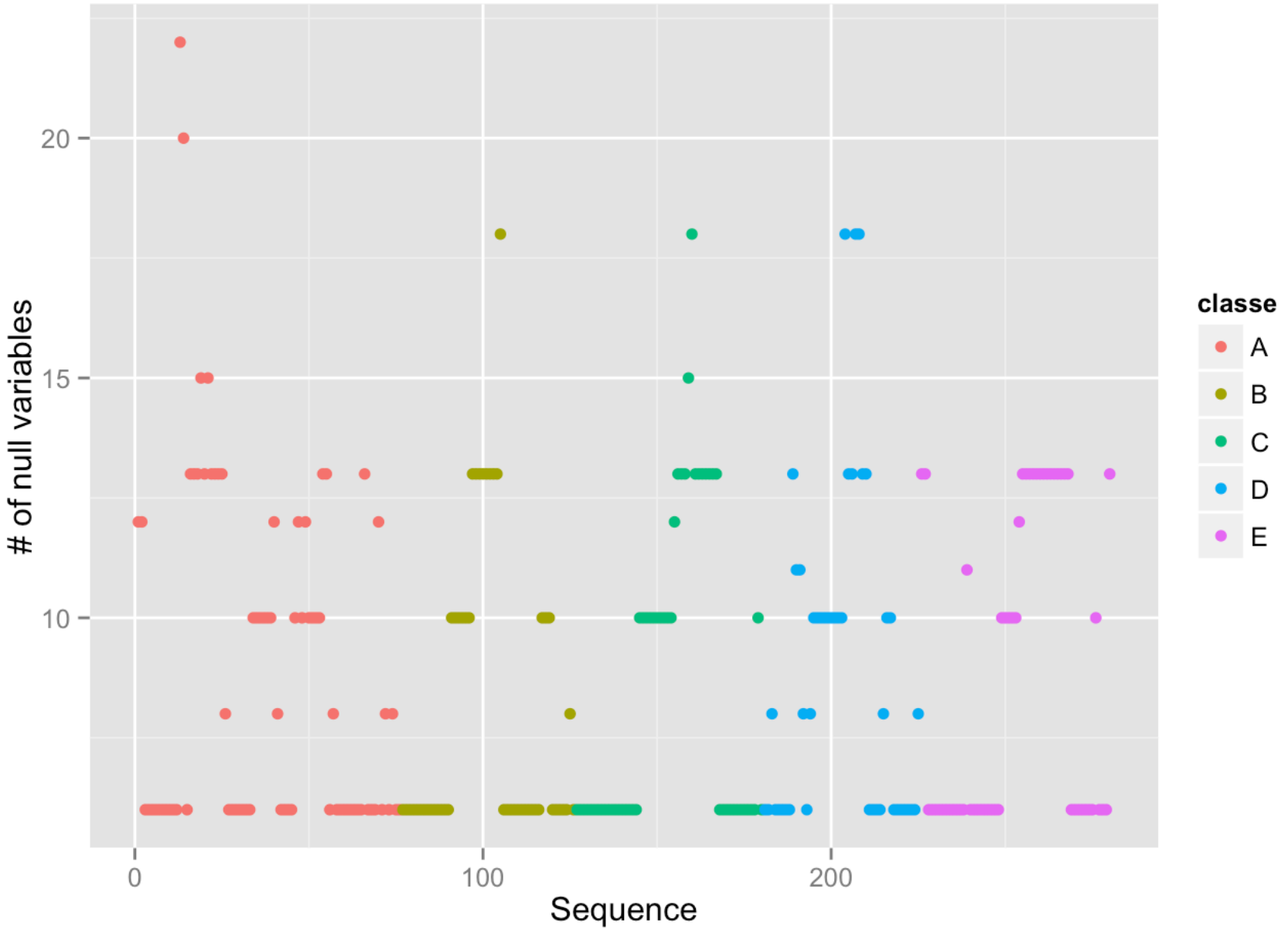
During date loading I saw a lot of N/A values, lets take a closer look on them.

```
qplot(x, y, data = data.frame(y = sort(apply(is.na(subsetTraining), 2, sum)), x =
seq_along(colnames(subsetTraining))), xlab = "Variables", ylab = "Observations wit
h Null values", main = "Unused variables")
```



Significant part of observations have only 52 variables. Lets check classes in observations (280) with non null variables to see any patterns.

```
df <- data.frame(y = apply(is.na(subsetTraining[apply(is.na(subsetTraining), 1, su
m) < 53, ]), 1, sum), classe = subsetTraining[apply(is.na(subsetTraining), 1, sum)
< 53, 153], x = seq_along(subsetTraining[apply(is.na(subsetTraining), 1, sum) < 53
, 153]))
qplot(x, y, data = df, colour = classe, xlab = "Sequence", ylab = "# of null varia
bles")
```



I dont’t see any pattern in class - number of null variables, so lets skip those predictors for now, latter we can use them to build separate models for different cases and combine them.

```
nullVars <- apply(is.na(subsetTraining), 2, any)
```

Building model

I use Random Forest as one of most accurate models for classification problem ## Training

```
modelRf <- train(classe ~ ., data = subsetTraining[, !nullVars], method = "rf")
```

```
## Loading required package: randomForest
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

Testing

```
predictions <- predict(modelRf, newdata = subsetTesting[,!nullVars])
confusionMatrix(predictions, subsetTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    7    0    0    0
##          B    1 1130    8    0    0
##          C    0    2 1016   20    0
##          D    0    0    2  944    4
##          E    1    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.9924
##                  95% CI : (0.9898, 0.9944)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9903
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9921   0.9903   0.9793   0.9963
## Specificity            0.9983   0.9981   0.9955   0.9988   0.9998
## Pos Pred Value         0.9958   0.9921   0.9788   0.9937   0.9991
## Neg Pred Value         0.9995   0.9981   0.9979   0.9959   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2841   0.1920   0.1726   0.1604   0.1832
## Detection Prevalence   0.2853   0.1935   0.1764   0.1614   0.1833
## Balanced Accuracy      0.9986   0.9951   0.9929   0.9890   0.9980
```

Prediction

```
predict(modelRf, 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
```