Course 8 Project - Prediction Assignment Writeup

General Comments

My analysis below details how I received 20/20 in the output quiz.

```r
knitr::opts_chunk$set(echo = TRUE)
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.4.3
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.3
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.4.3
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.3
```

```
## 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
```

Step 01 - Load Data

After importing data into R, I removed the first column, which is an index column. Also, I converted string columns to factors for purposes of running a random forest model.

```r
##Load the data into R
train_dt <- fread("pml-training.csv")
test_dt <- fread("pml-testing.csv")

train_dt[, V1:=NULL]
test_dt[, V1:=NULL]

train_dt[, classe:=as.factor(classe)]
train_dt[, user_name:=as.factor(user_name)]
train_dt[, new_window:=as.factor(new_window)]

test_dt[, user_name:=as.factor(user_name)]
test_dt[, new_window:=as.factor(new_window)]
```

Step 02 - Data Cleaning

I remove all columns that have at least one missing value. Also, I removed the time variables because the record date is irrelevant to the type of activity one performs.

```
##Remove the columns that contain NA or null values in both train and test data
is_train_col_na <- train_dt[ , apply(train_dt, 2, function(x) !any(is.na(x),x==""))]
is_test_col_na <- test_dt[ , apply(test_dt, 2, function(x) !any(is.na(x),x==""))]


valid_train_columns <- c()

for (i in 1:length(is_train_col_na)){

  if(is_train_col_na[i] ==TRUE){
    valid_train_columns <- c(valid_train_columns,names(is_train_col_na[i]))
  }
}

train_dt<-train_dt[,valid_train_columns,with=FALSE]


valid_test_columns <- c()

for (i in 1:length(is_test_col_na)){

  if(is_train_col_na[i] ==TRUE){
    valid_test_columns <- c(valid_test_columns,names(is_test_col_na[i]))
  }
}

test_dt<-test_dt[,valid_test_columns,with=FALSE]

##Remove Time Variables
train_dt <- train_dt[,-c(2:4)]
test_dt <- test_dt[,-c(2:4)]


##Make sure both the train and test data have the same features
setdiff(names(train_dt),names(test_dt))
```

```
## [1] "classe"
```

```
##Okay

setdiff(names(test_dt),names(train_dt))
```

```
## [1] "problem_id"
```

```
test_dt[, problem_id:=NULL]
setdiff(names(test_dt),names(train_dt))
```

```
## character(0)
```

```
##Okay
```

Step 03 - Building & Running a Random Forest Model

In building and running a random forest model, I remove variables whose freqRatio and percenUnique, as defined by the nearZeroVar function in the caret package, whose values are <=5 or >=5, respectively.

```r
train_x <- train_dt[,!"classe"]
train_y <- train_dt[,classe]
test_x <- test_dt


normalization <- preProcess(train_x)
train_x <- predict(normalization,train_x)
test_x <- predict(normalization,test_x)

zer0_var <- nearZeroVar(train_x,saveMetrics = TRUE)
train_x <- train_x[,which(zer0_var$freqRatio<=5),with=FALSE]
test_x <- test_x[,which(zer0_var$freqRatio<=5),with=FALSE]

zer0_var <- nearZeroVar(train_x,saveMetrics = TRUE)
train_x <- train_x[,which(zer0_var$percentUnique>=5),with=FALSE]
test_x <- test_x[,which(zer0_var$percentUnique>=5),with=FALSE]


rf_model <- randomForest(train_x,train_y, ntree=500)

predict(rf_model,test_x)
```

```
##  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
```