Practical Machine Learning Course Project

The goal of this project is to predict how well someone performed an exercise based on data from fitness trackers worn by the subjects. Using a large dataset consisting of quantitative measurements of movement during exercise, models are to be built in order to predict if the exercise was done correctly or incorrectly.

The training set was first split into 2 parts, 80% going to training and 20% to validating. This was done using the createDataPartition function and so this was a random division based on the “classe” variable, teh variable that denotes whether or not the exercise was performed correctly. Any data columns that had NA had 15000+ NA’s out of under 16000 entries were omitted.

```
library("caret")
```

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

```
library("randomForest")
```

```
## Warning: package 'randomForest' was built under R version 3.2.1
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
setwd("~/GitHub/Practical-Machine-Learning-Course-Project")
df <- read.csv("pml-training.csv", na.strings=c("#DIV/0!", "NA"))
datatesting <- read.csv("pml-testing.csv", na.strings=c("#DIV/0!", "NA"))

index <- createDataPartition(df$classe,p=0.8,list=FALSE)
training <- df[index,]
testing <- df[-index,]
```

Of the remaining columns, only the “forearm” predictors were used to build a resultant RF model. This was done to reduce modeling time and to get an idea of accuracy.

```
rf1 <- randomForest(classe ~ gyros_forearm_x + gyros_forearm_y + gyros_forearm_z + acce
l_forearm_x + accel_forearm_y + accel_forearm_z + magnet_forearm_x + magnet_forearm_y + m
agnet_forearm_z, method="rf", data=training, importance=TRUE)
confusionMatrix(testing$classe, predict(rf1, testing))$overall['Accuracy']
```

```
##  Accuracy
## 0.8236044
```

This model had approximately 82% accuracy when validated against the previously mentioned validation subset. Next, a model using the 53 columns that have no NAs was produced.

```
rf2 <- randomForest(classe ~  gyros_forearm_x + gyros_forearm_y + gyros_forearm_z + acce
l_forearm_x + accel_forearm_y + accel_forearm_z + magnet_forearm_x + magnet_forearm_y + m
agnet_forearm_z + num_window + roll_belt + pitch_belt + yaw_belt + total_accel_belt + gyr
os_belt_x + gyros_belt_y + gyros_belt_z + accel_belt_x + accel_belt_y + accel_belt_z + ma
gnet_belt_x + magnet_belt_y + magnet_belt_z + roll_arm + pitch_arm + yaw_arm + total_acce
l_arm + gyros_arm_x + gyros_arm_y + gyros_arm_z + accel_arm_x + accel_arm_y + accel_arm_z
+ magnet_arm_x + magnet_arm_y + magnet_arm_z + roll_dumbbell + pitch_dumbbell + yaw_dumbb
ell + gyros_dumbbell_x + gyros_dumbbell_y + gyros_dumbbell_z + accel_dumbbell_x + total_a
ccel_dumbbell + accel_dumbbell_y + accel_dumbbell_z + magnet_dumbbell_x + magnet_dumbbel
l_y + magnet_dumbbell_z + roll_forearm + pitch_forearm + yaw_forearm +total_accel_forear
m, method="rf", data=training, importance=TRUE)
confusionMatrix(testing$classe, predict(rf2, testing))$overall['Accuracy']
```

```
##  Accuracy
## 0.9982157
```

This model reached 99% accuracy. Assuming possible overfitting, the MeanDecreasedGini was calcualted to find out which variables that the model was most sensitive to change. These top 10 were used to fit another model.

```
names(sort(importance(rf2)[,7], decreasing=TRUE))[1:10]
```

```
##  [1] "num_window"        "roll_belt"         "yaw_belt"
##  [4] "pitch_forearm"     "magnet_dumbbell_z" "pitch_belt"
##  [7] "magnet_dumbbell_y" "roll_forearm"      "magnet_dumbbell_x"
## [10] "accel_dumbbell_y"
```

```
rf3 <- randomForest(classe ~  num_window+roll_belt+yaw_belt+pitch_forearm+magnet_dumbbel
l_z+pitch_belt+magnet_dumbbell_y, method="rf", data=training, importance=TRUE)
confusionMatrix(testing$classe, predict(rf3, testing))$overall['Accuracy']
```

```
##  Accuracy
## 0.9984706
```

This model also received 99% accuracy. In an effort to reduce possible over fitting, the lower 5 variables were deselected.

```
rf4 <- randomForest(classe ~  num_window+roll_belt+yaw_belt+pitch_forearm+magnet_dumbbel
l_z+pitch_belt+magnet_dumbbell_y+roll_forearm+magnet_dumbbell_x+roll_dumbbell, method="r
f", data=training, importance=TRUE)
confusionMatrix(testing$classe, predict(rf4, testing))$overall['Accuracy']
```

```
##  Accuracy
## 0.9989804
```

This model received 99% accuracy.

Going further to reduce the number of predictors to depend on for the most accurate results, it was found using only the “num_window” variable was all that was needed to get 99% accuracy.

```
rf5<- randomForest(classe ~  num_window+roll_belt+yaw_belt+pitch_forearm+magnet_dumbbel
l_z, method="rf", data=training, importance=TRUE)
confusionMatrix(testing$classe, predict(rf5, testing))$overall['Accuracy']
```

```
##  Accuracy
## 0.9984706
```

```
rf6<- randomForest(classe ~  num_window, method="rf", data=training, importance=TRUE)
confusionMatrix(testing$classe, predict(rf6, testing))$overall['Accuracy']
```

```
##  Accuracy
## 0.9994902
```

An examination into how well a predictive model could be built without the num_window variable was performed.

```
rf7 <- randomForest(classe ~ roll_belt+yaw_belt+pitch_forearm+magnet_dumbbell_z+pitch_bel
t+magnet_dumbbell_y+roll_forearm+magnet_dumbbell_x+roll_dumbbell, method="rf", data=train
ing, importance=TRUE)
confusionMatrix(testing$classe, predict(rf7, testing))$overall['Accuracy']
```

```
##  Accuracy
## 0.9877645
```

```
rf8<- randomForest(classe ~  roll_belt+yaw_belt+pitch_forearm+magnet_dumbbell_z, metho
d="rf", data=training, importance=TRUE)
confusionMatrix(testing$classe, predict(rf8, testing))$overall['Accuracy']
```

```
##  Accuracy
## 0.9467244
```

```
rf9<- randomForest(classe ~ yaw_belt, method="rf", data=training, importance=TRUE)
confusionMatrix(testing$classe, predict(rf9, testing))$overall['Accuracy']
```

```
## Accuracy
## 0.564364
```

This shows that without the num_window variable, accuracy begins to drop off for the model's predictive power.

All models were to make predictions from the official test set. In the end, all models, except the first and last, perfectly pass against the official test dataset. Evidentally, only the num_window was needed to make accurate predictions.

```
as.character(predict(rf1,datatesting))
```

```
##  [1] "B" "A" "C" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "E" "D" "B"
```

```
as.character(predict(rf2,datatesting))
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
as.character(predict(rf3,datatesting))
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
as.character(predict(rf4,datatesting))
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
as.character(predict(rf5,datatesting))
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
as.character(predict(rf6,datatesting))
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
as.character(predict(rf7,datatesting))
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
as.character(predict(rf8,datatesting))
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
as.character(predict(rf9,datatesting))
```

```
##  [1] "B" "B" "B" "C" "E" "A" "D" "B" "A" "A" "B" "A" "E" "A" "D" "E" "C"
## [18] "B" "D" "A"
```