Brief Description: With wearable devices, it is now possible to collect large amounts of data to quantify particular exercises. Not much research has been done to show how correctly someone does a particular exercise. The goal of the Practical Machine Learning project is to predict if someone is doing the exercise correctly or incorrectly. Data was taken from the accelerometers on the belt, forearm, arm, and dumbell from 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 (see the section on the Weight Lifting Exercise Dataset).

For more information on the Weight Lifting Exercises Dataset, please refer to the researh paper at http://groupware.les.inf.puc-rio.br/work.jsf?p1=11201. The authors of the paper are cited below:

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.

Analysis:

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

First load the training and test data into R with read.csv("", header=T). Use View() to view your dataset into R. The "V" in View() is capitalized.

Second, with such a large data set you would want to clean your data to remove missing data and unnecessary columns that might decrease the accuracy of your prediction model. I choose to remove the any columns associated with the timestamps, subject names, window numbers, and if a new window or not. The reason why you would want to remove subject names is to avoid producing a model for each individual subject. If you keep the window numbers then the classifier can use it to pick the outcome of the model. If you keep the timestamps in your model then the classifier will use vectors with nearby times. I used subset(, select= c(:)) to filter my data.

Third, you would want to measure the importance of each column in your model. Choose the most important columns to fit your model with to improve the accuracy of your model and to prevent overfitting your model. The prediction models are in the caret package. Download the caret package with, install.packages(“caret”), and load the caret library with, library(caret). Train your model on the “classe” variable with, train(classe~ ., data= , method=“rf”, importance=TRUE). Random forest (method=“rf”) uses cross validation when you run your model to split your data.

Warning: Model might take hours to run depending on your computer. After running your model, you might want to save it to your computer with, saveRDS(, “name.Rds”). Saving your file will help prevent you from having to run your model every time you shutdown and restart R. Use, readRDS(“name.Rds”) to load your model when you restart R.

Next, use varImp() to list the variables by maximum importance across the classes. I choose to use the top 10 variables for my model. Take the list of most important variables and filter your training data set with subset().

Fourth, predict on your model with predict(,). View the accuracy of your model with, confusionMatrix(, $classe). Below, is my model with 100% Accuracy and Kappa. 100% Kappa means that the classifier is in total agreement with a random classifier.

Confusion Matrix and Statistics

Overall Statistics

Accuracy : 1
95% CI : (0.9998, 1) No Information Rate : 0.2844
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 1
Mcnemar''s Test P-Value : NA

Use print($finalModel) to view the out of bag (OOB) error rate.

Call: randomForest(x = x, y = y, mtry = param$mtry, importance = TRUE) Type of random forest: classification Number of trees: 500 No. of variables tried at each split: 2

OOB estimate of error rate: 4.35%

Fifth, Clean your testing data similar to how you cleaned your training data. Run your prediction algorithm on your filtered testing data with, predict(,). Compare the OOB error rate when running your prediction model on your testing data.

My first prediction model had an OOB error rate of 14%. On my first attempt I made the mistake by filtering on too much data. I went back and refit my model and on the variables with maximum importance across the classes. I filtered on my new variables and retrained my model.

My second prediction model had an OOB error rate of 4% and I was able to predict on 19/20 test cases correctly.