Introduction

What a wonderful, modern world we live in today where technology has permeated so many aspects of our lives including health and fitness. We have many options to choose from to collect data about personal activities. But, we can do better than just saving and viewing this information. We can create a prediction model which learns from our data and foresees the outcomes of interesting questions we think of. This paper will describe how I implemented this process using weight lifting exercise data. I try to predict how well each subject performed the exercise and which typical mistake was performed out of four mistakes.

Six participants were asked to perform dumbbell lifts five different ways as described in the following table.

Accelerometers were placed on their belt, forearm, arm and dumbbell to collect the activity data.

The Data Set

Data Processing

The original data set contains variables that the researchers needed to compute their own features using sliding windows and a feature selection algorithm based on correlation and backtracking. This resulted in missing values, Excel divide by zero errors, and sparse columns. I handled these values using R’s read.csv function and the stringsAsFactors and na.strings arguments.

To handle the sparse columns, I created an R function that computes the number of missing values for each variable. Using the missing value counts, I created new data frames with a subset of columns that do not have any missing values and the rest of the observations in the training and testing sets.

This reduced the number of variables from 160 to 60. I also discarded 7 of the 60 variables because they were not helpful for prediction since they are the observation number, user name, time stamps, and sliding window data. The final set of 53 variables comprised of the same 13 variables (roll, pitch, yaw, total acceleration, and accelerometer, gyroscope and magnetometer readings on the x, y, and z axes) collected at the 4 locations (belt, arm, forearm, and dumbbell) and the outcome variable classe.

Data Exploration

Using only the training set, I created summaries of these 52 features to get a quick look at the feature names and range of values. They all appear to be random real numbers positive, negative and zero. I created histograms for each and noticed skewed data for magnet\_belt\_x, y, and z, magnet\_arm\_z, roll and pitch and accelerometer and gyroscope x, y, and z \_dumbbell, magnetometer dumbbell x, y, and z, and gyroscope forearm x,y, and z.

I also used caret’s featurePlot function to see interactions between some groups of features collected at each location.

I created density plots for the 52 variables colored by the classe outcome variable.

I also created a table of the counts of each exercise in the training set and it shows most of the observations fall in class A, according to specification, while the rest are approximately evenly distributed amongst the typical mistake classes B to E.

Choosing a Prediction Model

As far as I could tell, the exploratory data analysis did not show any imbalance between the outcome and predictors, outliers, or data that is unexplained by predictors. Transforming the skewed variables is more useful for regression and not classification which is the purpose of this project. So, I chose to create a model using all 52 features to predict the classe outcome.

Choosing a Classification Algorithm

There are many classification algorithms to consider such as linear regression, logistic regression, linear discriminant analysis, classification trees, bagging, random forests, boosting, and support vector machines.

This problem involves five classes and contains an adequate sample size and number of predictors. Linear regression and logistic regression are fine for two class problems, while early classification algorithms like linear discriminant analysis and classification trees are not as accurate as the ensemble methods of bagging, random forests, and boosting. Support vector machines may not be as interpretable as the others. I chose random forests because it reduces the variance in the data by averaging the observations in the data set. Additionally, it decorrelates the decision trees by randomly selecting predictors used to decide a split in the tree. Random forests is a state of the art ensemble classification algorithm and has been successful in many Kaggle competitions.

Training the Prediction Model

I used the caret R package to split the data into training and testing sets with a 75%/25% split. I chose K fold cross validation where K = 10 to balance the bias/variance trade-off of the sampled data. I implemented this with caret’s trainControl function. I used the parallel and doParallel packages to take advantage of the 4 cores in the i7-4700MQ processor in my Windows 10 HP Envy 15 laptop with 16GB of memory. Training time was about 10 minutes.

The random forest algorithm uses the parameter mtry or the number of variables randomly sampled as candidates at each split. The final model used mtry = 27 with an accuracy of 0.9917782.

Expected Out of Sample Error

I made predictions using this model on the 4,904 observations in the hold out testing set. Caret’s confusionMatrix function produces the expected out of sample error to help us see the performance of my classification model.

The model has an accuracy of 0.9929 with a 95% Confidence Interval of (0.9901, 0.995) and P-Value of 2.2e-16.

Conclusion

The hold out testing set counts of each exercise is shown below.

My model predicted 1,394 A, 939 B, 849 C, 791 D, and 896 E and misclassified 1 A, 10 B, 6 C, 13 D, and 5 E.

A very good colleague that we all know recommends five attributes of the âbestâ machine learning method; interpretable, simple, accurate, fast, and scalable.

-Interpretable -

-Simple - used weight lifting exercise data of five exercises and common, effective random forest algorithm to group observations into five categories

-Accurate - 99% accuracy on testing set

-Fast - about 9 minutes to run on a fairly powerful modern laptop

-Scalable - uses R code which can easily be migrated to computer cluster environment