Prediction Assignment Writeup

Peter Willmott

November 12, 2018

Executive Summary

This project aims to use data from accelerometers, placed on the arm, forearm, belt, and dumbell, to quantify and predict 'how well' a particular activity is performed. In this case 10 repetitions of the Unilateral Dumbell Biceps Curl. 6 male participants (20 - 28 years) lifted a dumbell (1.25kg) in 5 different fashions:

- **Class A:** Exactly according to specification
- **Class B:** Throwing the elbows to the front
- **Class C:** Lifting the dumbell only halfway
- **Class D:** Lowering the dumbell only halfway
- **Class E:** Throwing the hips to the front

The data set is credited to Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Excercises.

The final model was able to predict the manner in which they did excercise to an accuracy of 99.3% (Based on a Validation Data set) and an out of bag (OOB) error of 0.72%. The best model was Random Forest (RF), which produced greater accuracy than both Gradient Boosting Machine (GBM) and Linear discriminant analysis (LDA). Cross validation was used to estimate models accuracy.

The Data

The data was 2 csv files (pml-training.csv and pml-testing.csv), each consisting of 160 variables. The pml-training data contained 19622 observations and the pml-testing contained 20.

Loading required libraries, reading in the data and setting the seed.

```
library(caret)
library(ggplot2)
library(gbm)

set.seed(3141)
pml_training <- read.csv("pml-training.csv")
testing <- read.csv("pml-testing.csv")
```

Exploratory Analysis

Brief look at the size, missing values and near zero variance (NZV) of the data.

```
dim(testing)
```

```
## [1]  20 160
```

```
dim(pml_training)
```

```
## [1] 19622   160
```

```
sum(is.na(pml_training))
```

```
## [1] 1287472
```

```
sum(is.na(testing))
```

```
## [1] 2000
```

```
sum(nearZeroVar(pml_training))
```

```
## [1] 5201
```

```
sum(nearZeroVar(testing))
```

```
## [1] 8211
```

Looking at the data there are alot of observations in the pml_training data set, enough to allow for the creation ofa validation test set. However the data has a significant amount of NA values as well as near zero variance variables, these will have to be removed. Several columns are inconsequential to the model and will be removed (e.g. user_name).

Cleaning Data

```
pml_training1 <- pml_training[-c(1:7)] # Remove insignificant columns
nearZV <- nearZeroVar(pml_training1) # Index of NZV columns
pml_training2 <- pml_training1[, -nearZV] # Remove NZV columns
pml_training3 <- pml_training2[, colMeans(is.na(pml_training2)) <= .5] # Remove column if more than 50% is N
A

cTransform <- colnames(pml_training3[, -53]) # classe removed
testingF <- testing[cTransform] # cleaned testing set
```

Validation Data Set

creating a separate validation set from the cleaned pml_training data that can be used to perform cross validation and get the expected out of sample error.

```
inTrain <- createDataPartition(y = pml_training3$classe, p = 0.7, list = FALSE)
training <- pml_training3[inTrain,]
validation <- pml_training3[-inTrain,]
```

Training Models

A Random Forest, Gradient Boosting Machine and Linear Discriminant Analysis algorithm were performed separately on the training data set. Each method was trained using a k-fold cross validation of 5 subsets, this was done using the trControl variable introduced in the caret package. The models were then used to predict on the validation data set.

Random Forest

This method was chosen because of its very succesful use in countless research papers, and is well suited to this data set because of the non-bionomial outcome and large sample size. Random forest is an example of a bagging algorithm

Gradient Boosting Machine (GBM)

This model was chosen because it is one of the most powerful techniques for building predictive models. GBM does have a tendancy to overfit but does reduce bias and variance. GBM was also used as it is an example of a boosting algorithm, contrast to Random Forest.

Linear Discriminant Analysis (LDA)

This model was chosen because there are more than 2 classes in this classification problem. It is a linear classification technique, it is a simple model in both preparation and application and the linear approach is different to the above algorithms.

```
train.control <- trainControl(method = "cv", number = 5) # Setting the cross validation parameters

rf_model <- train(classe ~., method = "rf", data = training, trControl = train.control) # Training the rando
m forest model fit
gbm_model <- train(classe ~., method = "gbm", data = training, trControl = train.control) # Training the Gra
dient Boosting Machine
lda_model <- train(classe ~., method = "lda", data = training, trControl = train.control) # Training the Lin
ear Discriminant Analysis

# Predicting on the Validation data set
rf_pred <- predict(rf_model, validation)
gbm_pred <- predict(gbm_model, validation)
lda_pred <- predict(lda_model, validation)
```

Results

Random Forest performed the best out of the 3 chosen algorithms

Originally the models were combined into a stacked model but the model performed equal to the random forest model and likely introduced

less over fitting.

```
# Accuracy
confusionMatrix(rf_pred, validation$classe) # Random Forest
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    3    0    0    0
##          B    0 1136    8    0    0
##          C    0    0 1018   19    0
##          D    0    0    0  944    1
##          E    0    0    0    1 1081
##
## Overall Statistics
##
##                Accuracy : 0.9946
##                  95% CI : (0.9923, 0.9963)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9931
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9974   0.9922   0.9793   0.9991
## Specificity            0.9993   0.9983   0.9961   0.9998   0.9998
## Pos Pred Value         0.9982   0.9930   0.9817   0.9989   0.9991
## Neg Pred Value         1.0000   0.9994   0.9983   0.9960   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1930   0.1730   0.1604   0.1837
## Detection Prevalence   0.2850   0.1944   0.1762   0.1606   0.1839
## Balanced Accuracy      0.9996   0.9978   0.9941   0.9895   0.9994
```

```
confusionMatrix(gbm_pred, validation$classe)$overall[1:2] # Gradient Boosting Machine
```

```
##  Accuracy     Kappa
## 0.9626168 0.9526970
```

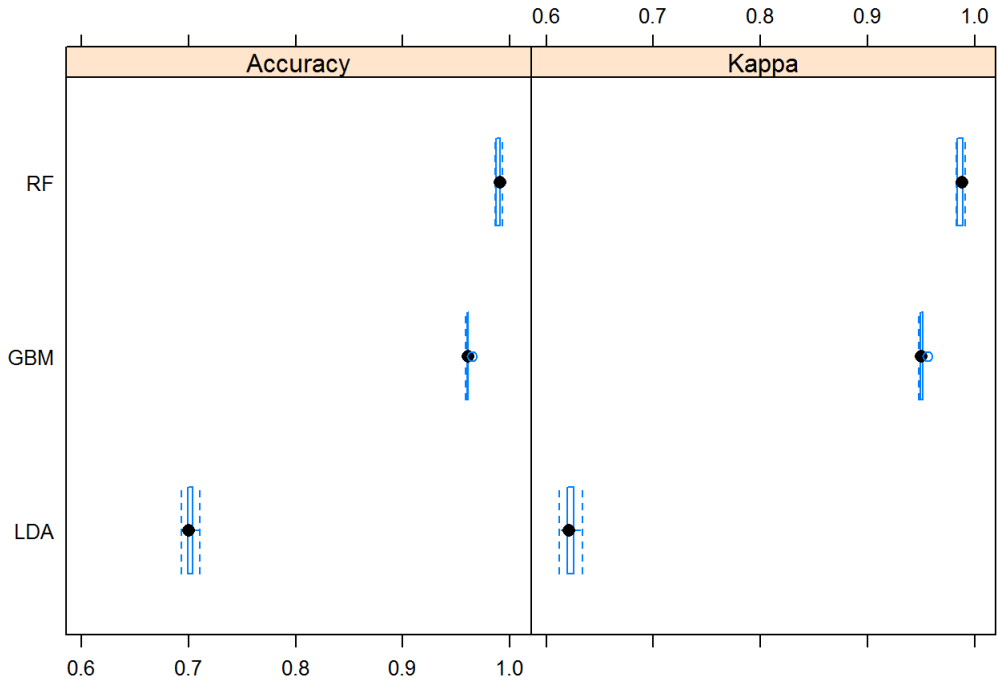
```
confusionMatrix(lda_pred, validation$classe)$overall[1:2] # Linear Discriminant Analysis
```

```
##  Accuracy     Kappa
## 0.7109601 0.6342000
```

```
results <- resamples(list(RF = rf_model, GBM = gbm_model, LDA = lda_model))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: RF, GBM, LDA
## Number of resamples: 5
##
## Accuracy
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## RF  0.9865308 0.9868996 0.9908992 0.9897358 0.9912664 0.9930834    0
## GBM 0.9588642 0.9599563 0.9606987 0.9611997 0.9614265 0.9650528    0
## LDA 0.6937068 0.6996724 0.7004004 0.7017551 0.7042972 0.7106987    0
##
## Kappa
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## RF  0.9829581 0.9834257 0.9884850 0.9870140 0.9889518 0.9912492    0
## GBM 0.9479604 0.9493461 0.9502527 0.9509121 0.9512094 0.9557918    0
## LDA 0.6123104 0.6195877 0.6209716 0.6225362 0.6257606 0.6340508    0
```

```
bwplot(results)
```




```
rf_model$finalModel # OOB error for random forest
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.72%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3903    2    0    0    1 0.0007680492
## B   17 2633    8    0    0 0.0094055681
## C    0   20 2373    3    0 0.0095993322
## D    0    0   40 2210    2 0.0186500888
## E    0    0    0    6 2519 0.0023762376
```

Since Random Forest performed the best out of all the models it was used to predict the independant test set

```
rf_predTest <- predict(rf_model, testingF)
rf_predTest
```

```
##  [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
```