Machine Learning Prediction Assignment

Cheng Seng

August 9, 2018

Goal of Project

The goal of this project was to predict the manner in which participants completed barbell lift exercise. Data was collected from accelerometers on the belt, forearm, arm, and dumbell of 6 participants who were asked to perform barbell lifts correctly and incorrectly in 5 different ways. The data for this assignment has been kindly provided from this source

Loading and preprocessing the data

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

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

```
## corrplot 0.84 loaded
```

First we load up the data and remove variables with NAs & empty strings and informational variables e.g. dates, name, etc that will not affect the prediction.

```
# load data
dat = read.csv("./data/pml-training.csv", na.strings=c("", "NA"))
# remove NA columns
dat <- dat[, colSums(is.na(dat)) == 0]
# remove unimportant columns in 1st seven columns
dat = dat[8:length(dat)]
```

The dataset now contains 53 variables, one of which is `classe` which we convert into a factor.

```
dat$classe<-as.factor(dat$classe)
```

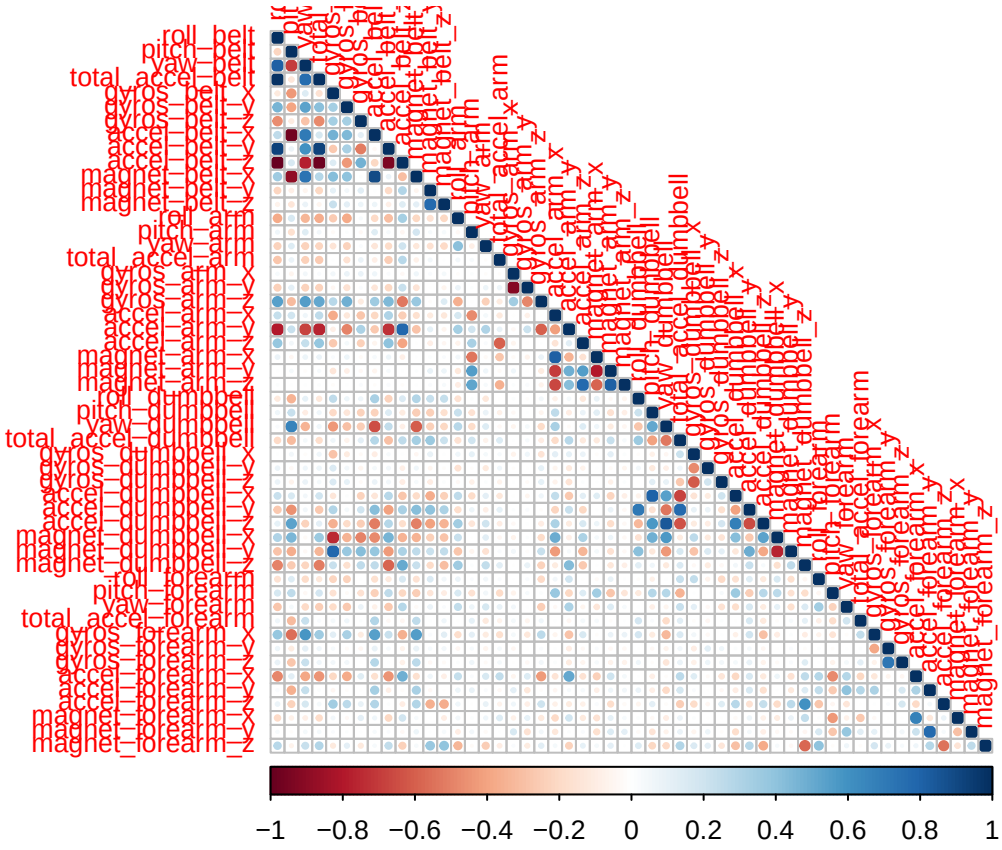
Training

We next split the data into training and validation sets.

```
# Create training and validation data
set.seed(41239) # Set seed for reproducibility
inTrain <- createDataPartition(dat$classe, p=0.7, list=FALSE)
training <- dat[inTrain,]
validation <- dat[-inTrain,]
```

A correlation matrix was plotted with the training data to visualize any highly correlated variables. The circles indicate the level of correlation between different variables.

```
correlMatrix <- cor(training[, -length(names(training)) ] )
corrplot(correlMatrix, type = "lower", tl.cex = 0.8)
```



We choose to use the `Random Forest` algorithm to fit a model to select the vital varaibles and avoid overfitting. We will apply a 5-fold cross validation in the algorithm.

```
rf <- train(classe ~ ., data = training, method = "rf", trControl = trainControl(method = "cv", 5), ntr
rf
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10990, 10987, 10991, 10991, 10989
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9900994  0.9874750
##   27    0.9898812  0.9871983
##   52    0.9831106  0.9786332
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

Validation

The validation data is then used with the fitted model. The confusion matrix estimates the performance of the model on new data.

```
pred = predict(rf,validation)
#Get an estimate of how well the model has been trained;
confusionMatrix(validation$classe, pred)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    2    0    0    0
##          B    9 1129    1    0    0
##          C    0    5 1021    0    0
##          D    0    0   11  952    1
##          E    0    0    0    1 1081
##
## Overall Statistics
##
##                Accuracy : 0.9949
##                  95% CI : (0.9927, 0.9966)
##     No Information Rate : 0.2856
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9936
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9946   0.9938   0.9884   0.9990   0.9991
## Specificity            0.9995   0.9979   0.9990   0.9976   0.9998
## Pos Pred Value         0.9988   0.9912   0.9951   0.9876   0.9991
## Neg Pred Value         0.9979   0.9985   0.9975   0.9998   0.9998
## Prevalence             0.2856   0.1930   0.1755   0.1619   0.1839
## Detection Rate         0.2841   0.1918   0.1735   0.1618   0.1837
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9971   0.9959   0.9937   0.9983   0.9994
```

```
# print out of sample error
acc <- postResample(pred, validation$classe)[1]
print(round(acc, digits = 4))
```

```
## Accuracy
##   0.9949
```

```
ose <- (1 - acc) * 100
names(ose) <- c("Out of Sample Error")
print(round(ose, digits = 2))
```

```
## Out of Sample Error
##                0.51
```

The performance results indicate that random forest may be a good model for predicting this data as it has an accuracy of `99.5%` with an out of sample error of `0.51%`

Testing

We finally test the fitted model with the testing data. The same cleaning steps performed for the training data are repeated for the testing applied data. The validated model is applied to predict the classes for each of the 20 rows.

```
# load & clean test data
testing = read.csv("./data/pml-testing.csv", na.strings=c("", "NA"))
testing <- testing[, colSums(is.na(testing)) == 0]
# remove unimportant columns in 1st seven columns
testing = testing[8:length(testing)]

# Fit the model to the testing data
predict(rf,testing)
```

```
##  [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
```