Practical Machine Learning - Final Project

Camilla Jensen

1/4/2018

Introduction

Using a large dataset from the quantified self movement research community we must in this project build a prediction model towards revealing whether a person performed barbell lifts correctly or incorrectly based on observed movement patterns using smartband type of devices (e.g. Jarbone Up, Nike FuelBand, Fitbit etc.).

The data used in the project is described and referenced on the followsing website: http://groupware.les.inf.puc-rio.br (http://groupware.les.inf.puc-rio.br). See also Velloso, Bulling, Gellerson, Ugulino and Fuks (2013).

The problem set given to us has in this report been broken down into the following tasks: 1. Preparing and preprocessing the data and setting aside a portion of the training data for cross validation; 2. Applying one or several prediction models to the data; 3. Predicting 'classe' for the 20 test cases; 4. Discussion of validity.

Each task is done section by section below.

Preparing and preprocessing the data

The testing dataset on this assignment is somewhat different from what we have seen on the course assignments and here consists only of 20 observations. The task is therefore to use a very large dataset of 19,622 observations (namely the training data) to reveal task performance in the small testing dataset.

Initial screening of the testing dataset shows that only 60 variables are complete without any NA's. This must be important for model performance. Selection of predictors must therefore take outset in what is available in the testing dataset. Also variables that are unrelated with task performance (e.g. the descriptors in the first 7 columns of the dataset) are excluded from the final testing and training datasets as well.

The remaining 52 predictor variables are then passed on to the next task which is preprocessing.'Classe' is added as an empty column in the testing dataset for sake of easy coding so we can match the predictors more easily under this task in the two datasets. It is exactly this column that we need to predict with the exercise.

(Testing has 54 variables because there is one column named 'problemid' that we keep in case we will need it later which is not in the training dataset.)

```
train_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
"
test_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

download.file(url=train_url, destfile="train.csv")
download.file(url=test_url, destfile="test.csv")

test <- read.csv("~/Desktop/test.csv", na.strings = c("NA", ""))
train <- read.csv("~/Desktop/train.csv", na.strings = c("NA", ""))

test <- test[ , ! apply( test , 2 , function(x) all(is.na(x)) ) ]

remove_vars <- c('X', 'user_name','raw_timestamp_part_1', 'raw_timestamp_part_2',
'cvtd_timestamp', 'new_window', 'num_window')

test["classe"] <- NA

testing <- test[,!(names(test) %in% remove_vars)]

train <- train[,(names(train) %in% names(testing))]

library(caret)

inTrain <-  createDataPartition(y=train$classe, p=0.90, list=FALSE)
training <- train[inTrain,]
validating <- train[-inTrain,]

dim(testing)
```

```
## [1] 20 54
```

```
dim(training)
```

```
## [1] 17662    53
```

```
dim(validating)
```

```
## [1] 1960   53
```

The training dataset uses 90% of the original training dataset, whereas 10% is set aside for cross validations. This is possible and desirable owing to the large sample size of the trainig dataset and the fact that the prediction model’s accuracy is greater for data splitting the larger is the training dataset as long as the validating dataset does not become very small. We need a very accurate model to ensure that we make 100% correct projections for the 20 test cases in the testing dataset.

Building prediction models

Here we build and compare the performance of 3 different prediction models: random forest (rf), linear discriminant analysis (lda) and generalised boosted regression models (gbm). Comparing the accuracy (where the out of sample error is 1-accuracy) we see that the rf method performs best, followed by lda and with poorest performance for gbm.

```
library(caret)
Fit1 <- train(classe~., method="rf", preProcess="pca", data=training)
Fit2 <- train(classe~., method="lda", preProcess="pca",  data=training)
Fit3 <- train(classe~., method="rpart", preProcess="pca", data=training)

confusionMatrix(validating$classe, predict(Fit1, validating))
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   A   B   C   D   E
##          A 557   0   1   0   0
##          B   2 374   3   0   0
##          C   0   3 337   2   0
##          D   0   0   9 311   1
##          E   0   1   2   0 357
## 
## Overall Statistics
## 
##                Accuracy : 0.9878          
##                  95% CI : (0.9818, 0.9921)
##     No Information Rate : 0.2852          
##     P-Value [Acc > NIR] : < 2.2e-16       
## 
##                   Kappa : 0.9845          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9964   0.9894   0.9574   0.9936   0.9972
## Specificity            0.9993   0.9968   0.9969   0.9939   0.9981
## Pos Pred Value         0.9982   0.9868   0.9854   0.9688   0.9917
## Neg Pred Value         0.9986   0.9975   0.9907   0.9988   0.9994
## Prevalence             0.2852   0.1929   0.1796   0.1597   0.1827
## Detection Rate         0.2842   0.1908   0.1719   0.1587   0.1821
## Detection Prevalence   0.2847   0.1934   0.1745   0.1638   0.1837
## Balanced Accuracy      0.9979   0.9931   0.9771   0.9938   0.9977
```

```
confusionMatrix(validating$classe, predict(Fit2, validating))
```

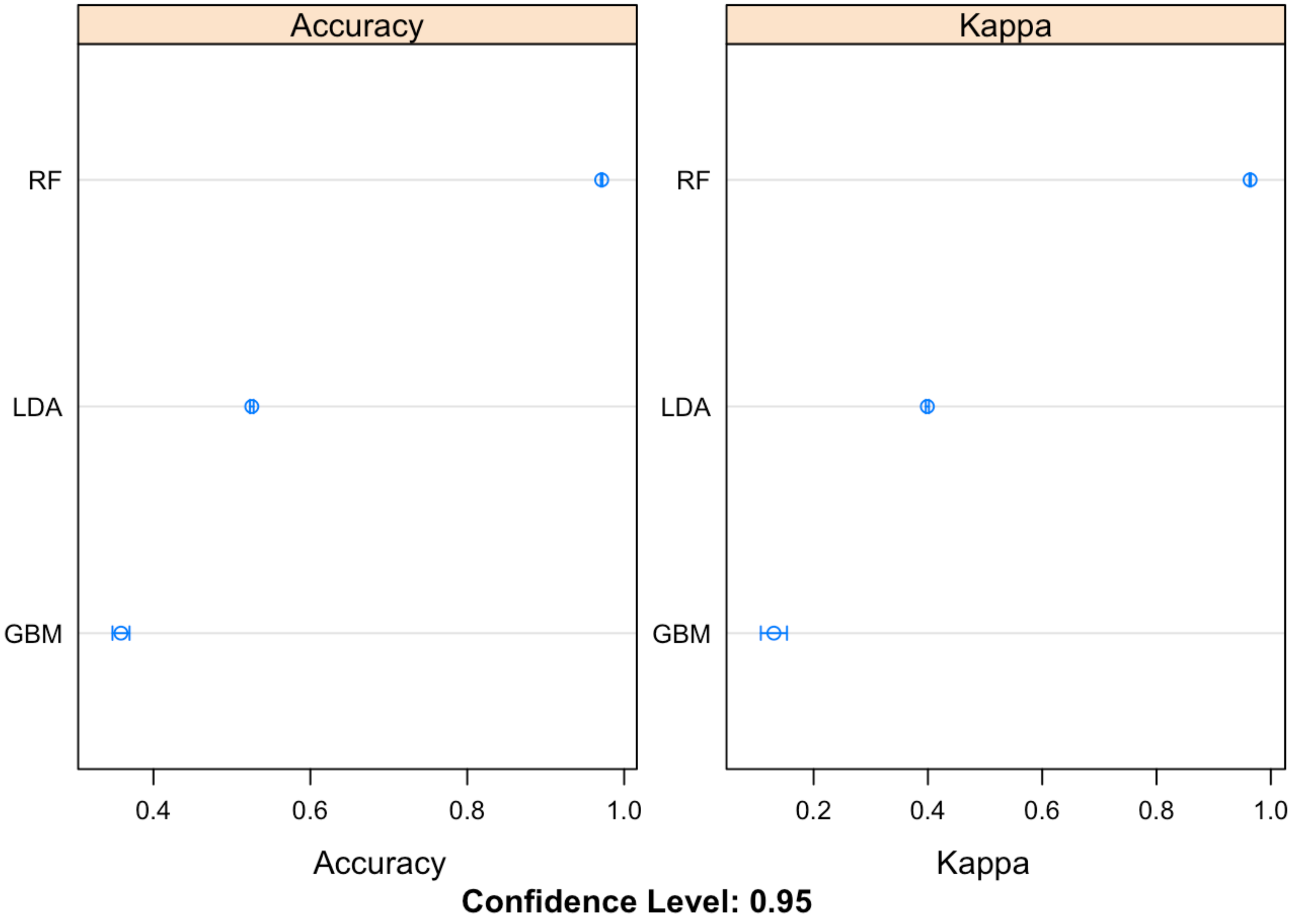
```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   A   B   C   D   E
##          A 368  54  46  85   5
##          B  95 158  68  40  18
##          C  98  34 166  35   9
##          D  22  58  39 169  33
##          E  31  68  49  53 159
## 
## Overall Statistics
##                                           
##                Accuracy : 0.5204          
##                  95% CI : (0.498, 0.5427)
##     No Information Rate : 0.3133          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.3917          
##  Mcnemar's Test P-Value : < 2.2e-16       
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.5993  0.42473  0.45109  0.44241  0.70982
## Specificity            0.8588  0.86083  0.88945  0.90368  0.88422
## Pos Pred Value         0.6595  0.41689  0.48538  0.52648  0.44167
## Neg Pred Value         0.8245  0.86464  0.87515  0.87004  0.95937
## Prevalence             0.3133  0.18980  0.18776  0.19490  0.11429
## Detection Rate         0.1878  0.08061  0.08469  0.08622  0.08112
## Detection Prevalence   0.2847  0.19337  0.17449  0.16378  0.18367
## Balanced Accuracy      0.7291  0.64278  0.67027  0.67304  0.79702
```

```
confusionMatrix(validating$classe, predict(Fit3, validating))
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   A   B   C   D   E
##          A 551   0   0   0   7
##          B 321   0   0   0  58
##          C 340   0   0   0   2
##          D 296   0   0   0  25
##          E 251   0   0   0 109
## 
## Overall Statistics
## 
##                Accuracy : 0.3367          
##                  95% CI : (0.3158, 0.3581)
##     No Information Rate : 0.8974          
##     P-Value [Acc > NIR] : 1               
## 
##                   Kappa : 0.086           
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.3132       NA       NA       NA  0.54229
## Specificity            0.9652   0.8066   0.8255   0.8362  0.85731
## Pos Pred Value         0.9875       NA       NA       NA  0.30278
## Neg Pred Value         0.1384       NA       NA       NA  0.94250
## Prevalence             0.8974   0.0000   0.0000   0.0000  0.10255
## Detection Rate         0.2811   0.0000   0.0000   0.0000  0.05561
## Detection Prevalence   0.2847   0.1934   0.1745   0.1638  0.18367
## Balanced Accuracy      0.6392       NA       NA       NA  0.69980
```

The results from the different methods are resampled and plotted for easier comparison. Linear methods strongly underperform the non-linear random forest method in this prediction model. This is perhaps not surprising as the dependent variable (or the variable we want to predict) has five levels and those levels cannot be easily translated to a cardinal or even ordinal scale.

```
results <- resamples(list(RF=Fit1, LDA=Fit2, GBM=Fit3))
scales <- list(x=list(relation="free"), y=list(relation="free"))
dotplot(results, scales=scales)
```



Prediction results

We now use the results from the rf prediction model (Fit1) to make prediction for the 20 test cases in the testing dataset. The results are shown below.

```
pred1 <- predict(Fit1, testing)
print(pred1)
```

```
##  [1] B A A A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```

Validity

Model accuracy was alone estimated using data splitting. This is deemed sufficient in view to the large sample size. Better results could be obtained by using more refined methods towards optimising model accuracy such as k-fold.

Different methods were applied to the training and validating parts of the split dataset. Those methods were linear as well as non-linear including random forest (rf), linear discriminant analysis (lda) and generalised boosted regression model (gbm).

Since there is little theory available to support model building and also expectations towards predictions it is hard to know beforehand what the optimal method would be. Here was used a reasonable range of different methods.

Overall the results are therefore held to be valid in predicting the behaviour for the test cases.

References

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.