Predicting exercise specification based on a series of variables

Rodrigo Lassance

Introduction

The data provided consists of measurements taken by people while doing exercises in different specifications:

- A - exactly according to the specification;
- B - throwing the elbows to the front;
- C - lifting the dumbbell only halfway;
- D - lowering the dumbbell only halfway;
- E - throwing the hips to the front).

The objective of this study was to identify at what specification the person was based on those measurements.

Treating the data

Before applying models to the data at hand, it is pertinent to check if all collected variables are useful. It was observed that many of them had more than 19000 missing values (in a dataset of 19622), so they were removed. In the test dataset, there are also variables with missing values, which would make predictions impossible due to lack of information for the models, so they were removed as well.

Now, even with the variables with missing data removed, there are other variables which could hinder the models. Specifically, the first variable (which only informs of the observation number) and the variables related to time (since, even if time was relevant in the train data, it could represent the experiment structure, when in reality we are interested in predicting the position of the person during exercise on any circunstance). Due to those reasons, they were also removed. Then, only 56 variables remain in the data.

```
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",'Train.csv')
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",'Test.csv')

Train<-read.csv('Train.csv')
Test<-read.csv('Test.csv')

NAcols<-c(which(apply(Train,2,function(x) sum(is.na(x)))>0),
          which(apply(Test,2,function(x) sum(is.na(x)))>0))
names(NAcols)<-NULL

Train<-Train[,-c(1,3,4,5,NAcols)]
Test<-Test[,-c(1,3,4,5,NAcols)]
```

Model building

In this section, the interest resides in choosing the model that seems to best explain the data at hand. Since the test dataset does not contain the response variable, it cannot be used for that purpose. Then, the decision

taken was to partition the training data in 2 parts, the first (75% of the data) being used to train the models and the second (25% of the data) to be checked for model accuracy.

```
library(caret)
```

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

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

```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
trainIndex = createDataPartition(Train[,1], p = 3/4,list=FALSE)
Train1<-Train[trainIndex,]
Train2<-Train[-trainIndex,]
```

The models used in this context were models already presented in other activities (quizzes of the course). The methods chosen were gbm, lda, rpart and rf. For estimation, a 10-fold cross validation procedure was used. The code used for these models are not shown here due to the fact that their output is extense and could hinder reading.

Once all of those four models were estimated, a combination of all of them was proposed, by collecting their predictions to the training data itself and using them as covariates in another random forest model, which was named as COMB.

```
library(caret)
pGBM<-predict(GBM,Train1)
pLDA<-predict(LDA,Train1)
pPART<-predict(RPART,Train1)
pRF<-predict(RF,Train1)
COMB.DF<-data.frame(classe=Train1$classe,pGBM,pLDA,pPART,pRF)
COMB<-train(classe~.,data=COMB.DF,method='rf',trControl=TC)
```

After all models were estimated, the interest resides in checking the prediction accuracy for the second part of the dataset, named as Train2.

```
library(caret)
p2GBM<-predict(GBM,Train2)
p2LDA<-predict(LDA,Train2)
p2PART<-predict(RPART,Train2)
p2RF<-predict(RF,Train2)
COMB.DF2<-data.frame(pGBM=p2GBM,pLDA=p2LDA,pPART=p2PART,pRF=p2RF)
p2COMB<-predict(COMB,COMB.DF2)
```

```
##                 GBM                 LDA                PART
##           0.9875612           0.7442904           0.4942904
##                  RF Combination of models
##           0.9975530           0.9975530
```

From this information, it seems that the random forest model and the combination of models provide similar accuracy. Then, to avoid unnecessary model complexity (which could demand a greater computational time), the RF model was chosen for prediction. This means that the expected out of sample error of the final model is less than 1% in this case.

Results

Now, with the model chosen and estimated, the last thing that remains is to provide the predictions for the test data. Unfortunately, since it does not contain the true values, we are unable to use it to evaluate the accuracy of the model for this case specifically.

```
data.frame(Prediction=predict(RF,Test))
```

```
##    Prediction
## 1           B
## 2           A
## 3           B
## 4           A
## 5           A
## 6           E
## 7           D
## 8           B
## 9           A
## 10          A
## 11          B
## 12          C
## 13          B
## 14          A
## 15          E
## 16          E
## 17          A
## 18          B
## 19          B
## 20          B
```