Predict form of barbell lifts

Code

Introduction

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. The goal of this analysis is to predict how well people perform barbell lifts based on data collected from multiple accelerometers during the exericise.

Data

The data consists of measurements taken from multiple accelerometers placed on 6 subjects while performing barbell lifts with different forms. The training data consists of ~20.000 samples of 160 variables.

Exploratory analysis

While exploring the dataset one of the most prominent observations is that 109 variables are empty. These variables are discarded from the dataset to create a tidy dataset to improve model training performance.

The code for loading and preprocessing the data can be found in appendix A.

Predictor selection

The next stap after creating a nice tidy dataset is to determine what variables to use for the statistical model. The following steps are performed to select the best variables:

- Create a smaller random dataset of 3.000 samples for speed purposes
- Traing “random forest” model with 5 fold cross validation
- Determine the importance of all the predictor variables
- Train “random forest” model with only the most important predictor
- Train another model based on the two most important predictors
- Repeat training a couple of models by adding the next most important predictor to every next model
- Determine if the accuracy of the models significantly increases by adding another important variable by using a t-test
- Choose the model with the most predictors which is still significantly better than it’s smaller predecessor
- Determine which variables are used for the best model

By following this process the dataset is reduced to a total of 5 variables which seem the most significant.

The code for selecting the most significant variables can be found in appendix B.

Training the final model

After selecting the best predictors the final “random forest” model is created using all training samples while performing 5 fold cross validation. The calculated accuracy while performing cross validation is approximately equal to the outer sample accuracy.

The outer sample error for the created model is equal to ~3%.

The code for training the final model and determine the error can be found in appendix C.

Appendix A: Load and preprocess data

Hide

```
#########################################################
##                   LOAD LIBRARIES                    ##
#########################################################

library(ggplot2)
library(caret)
library(readr)
library(dplyr)

#########################################################
##                   FETCH DATA                        ##
#########################################################

#fetch data
pml_training <- read_csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv",na = "NA")
pml_testing <- read_csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-t
esting.csv",na = "NA")

#inspect class distribution
barplot(table(pml_training$classe),xlab="classes",ylab="frequency",main="class dis
tribution training set")

#########################################################
##                   PREPROCES DATA                    ##
#########################################################


#### remove NA data columns ####

#get an overview of the emptyness of columns in training set
nas<-sapply(pml_training,function(x) sum(is.na(x)))
table(nas) # this table shows that most of the columns are filled or empty, no dif
ficult grey area which needs imputing

#remove empty columns from training and test
relevantColumns<-names(nas[nas<=1])
pml_training<-pml_training[, names(pml_training)[names(pml_training) %in% relevant
Columns] ]

#### remove empty string columns ####
```

```
#get an overview of the emptyness of columns in training set
nas<-sapply(pml_training,function(x) sum(x==""))
table(nas) # this table shows that most of the columns are filled or empty, no dif
ficult grey area which needs imputing

#remove empty string columns
relevantColumns<-names(nas[nas==0])
pml_training<-pml_training[, names(pml_training)[names(pml_training) %in% relevant
Columns] ]

#### remove other non relevant columns ####
pml_training<-pml_training[ , !(names(pml_training) %in% c('cvtd_timestamp','raw_t
imestamp_part_1','raw_timestamp_part_2','X1','new_window','num_window'))]

#### convert string columns to factor ####
pml_training %>% mutate_if(is.character, as.factor) -> pml_training
```

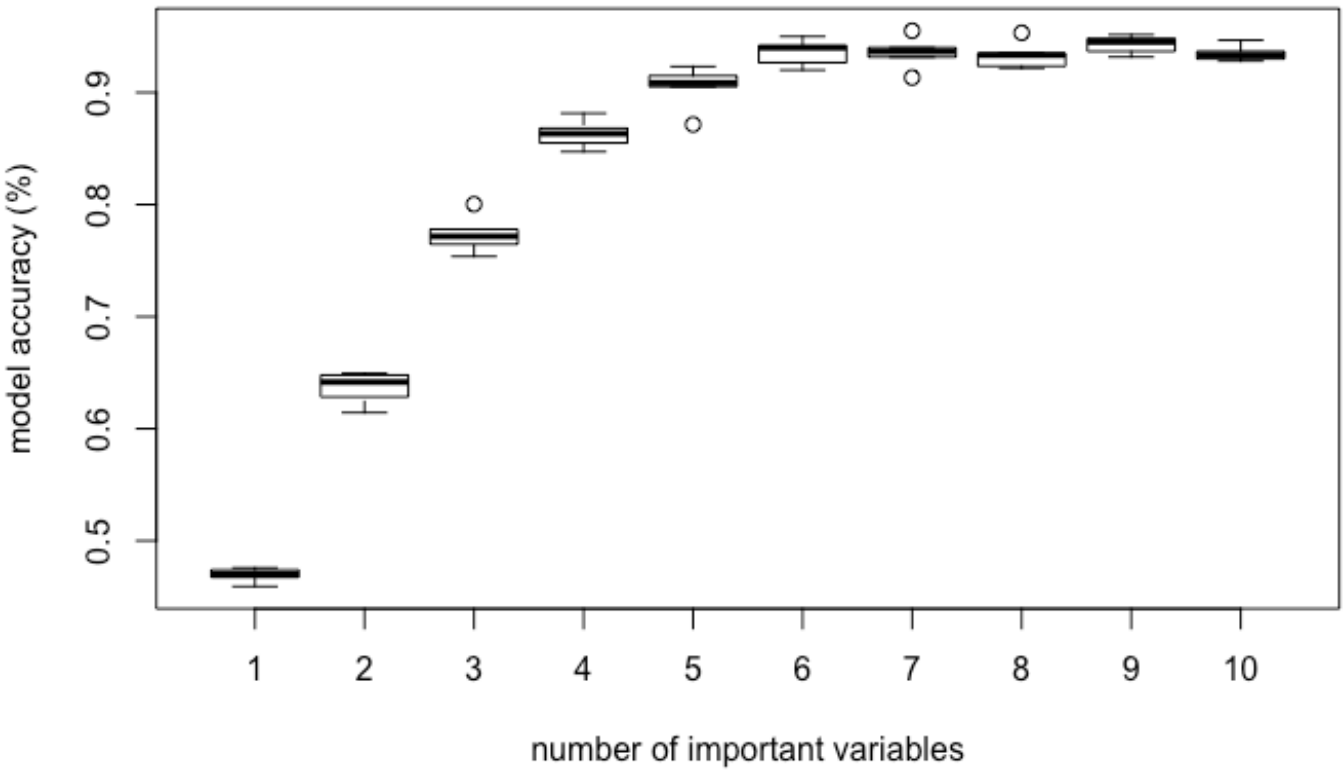
Appendix B: Select best predictors

Hide

```
# init training parameters
train_control <- trainControl(method="cv", number=5)

# construct smaller training data set (lesser rows) for speed purposes
set.seed(1234)
training<-pml_training[sample(1:dim(pml_training)[1])[1:3000],]

#train random forest model
mod_rf<-train(classe~.,data=training,method='rf',trControl=train_control,preProces
s = c("center","scale"))

#determine importance of variables
importance<-varImp(mod_rf,scale=TRUE)

#get overview of importance of variables
importance

#train multiple models
mod1<-train(classe~roll_belt,data=training,method='rf',trControl=train_control,pre
Process = c("center","scale"))
mod2<-train(classe~roll_belt+pitch_forearm,data=training,method='rf',trControl=tra
in_control,preProcess = c("center","scale"))
mod3<-train(classe~roll_belt+pitch_forearm+pitch_belt,data=training,method='rf',tr
Control=train_control,preProcess = c("center","scale"))
mod4<-train(classe~roll_belt+pitch_forearm+pitch_belt+yaw_belt ,data=training,meth
od='rf',trControl=train_control,preProcess = c("center","scale"))
mod5<-train(classe~roll_belt+pitch_forearm+pitch_belt+yaw_belt+magnet_dumbbell_y,d
ata=training ,method='rf',trControl=train_control,preProcess = c("center","scale")
)
mod6<-train(classe~roll_belt+pitch_forearm+pitch_belt+yaw_belt+magnet_dumbbell_y+r
oll_forearm ,data=training ,method='rf',trControl=train_control,preProcess = c("ce
nter","scale"))
mod7<-train(classe~roll_belt+pitch_forearm+pitch_belt+yaw_belt+magnet_dumbbell_y+r
oll_forearm+roll_dumbbell ,data=training ,method='rf',trControl=train_control,preP
rocess = c("center","scale"))
mod8<-train(classe~roll_belt+pitch_forearm+pitch_belt+yaw_belt+magnet_dumbbell_y+r
oll_forearm+roll_dumbbell+gyros_dumbbell_y  ,data=training ,method='rf',trControl=
train_control,preProcess = c("center","scale"))
mod9<-train(classe~roll_belt+pitch_forearm+pitch_belt+yaw_belt+magnet_dumbbell_y+r
oll_forearm+roll_dumbbell+gyros_dumbbell_y+magnet_dumbbell_x   ,data=training ,met
hod='rf',trControl=train_control,preProcess = c("center","scale"))
mod10<-train(classe~roll_belt+pitch_forearm+pitch_belt+yaw_belt+magnet_dumbbell_y+
roll_forearm+roll_dumbbell+gyros_dumbbell_y+magnet_dumbbell_x+accel_dumbbell_y   ,
data=training ,method='rf',trControl=train_control,preProcess = c("center","scale"
))
```

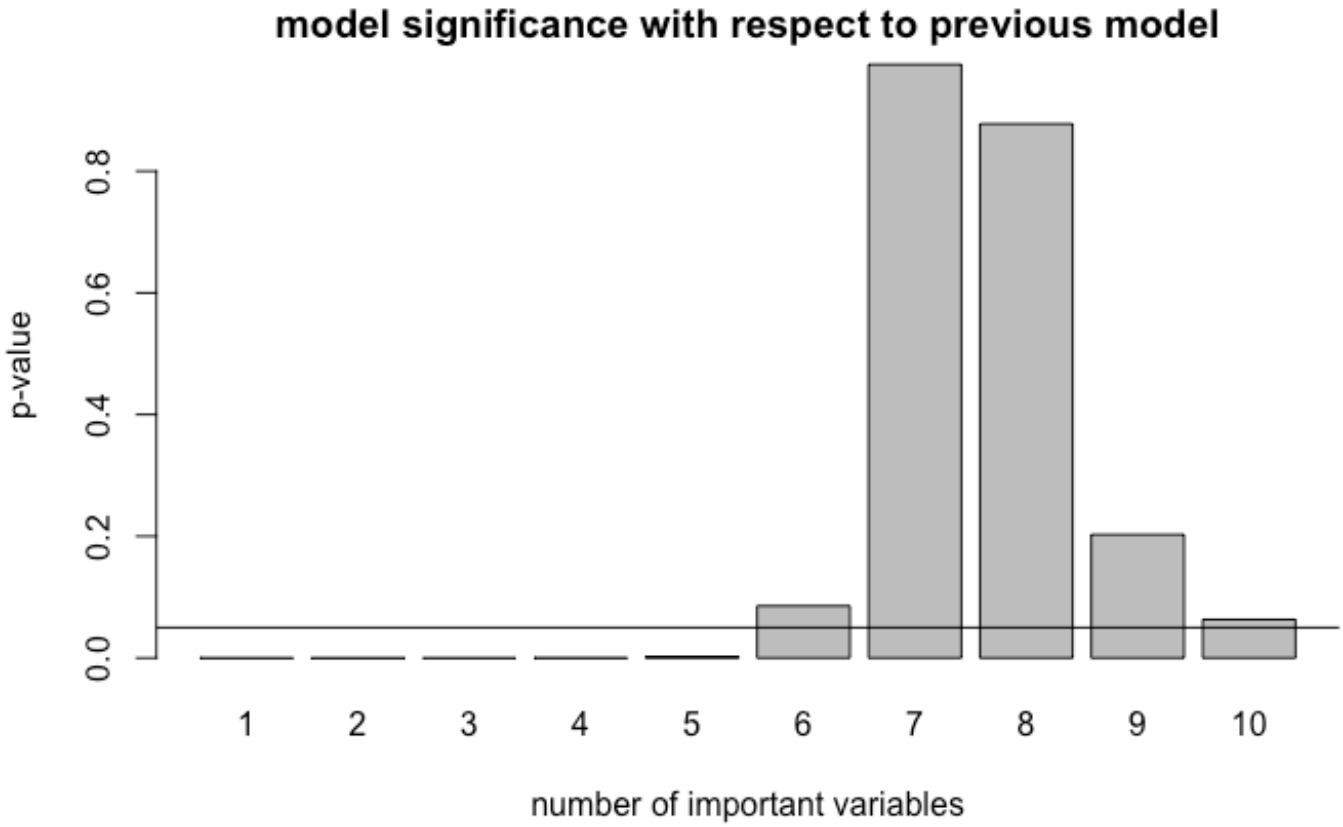
Hide

```
#plot accuracy of all models
boxplot( data.frame(mod1$resample$Accuracy,mod2$resample$Accuracy,mod3$resample$Ac
curacy,mod4$resample$Accuracy,mod5$resample$Accuracy,mod6$resample$Accuracy,mod7$r
esample$Accuracy,mod8$resample$Accuracy,mod9$resample$Accuracy,mod10$resample$Accu
racy),names=1:10,ylab="model accuracy (%)",xlab="number of important variables")
```



Hide

```
#plot significance of models compared to the previous (less predictors) model
barplot(
c(
t.test(mod1$resample$Accuracy)$p.value
,t.test(mod2$resample$Accuracy-mod1$resample$Accuracy)$p.value
,t.test(mod3$resample$Accuracy-mod2$resample$Accuracy)$p.value
,t.test(mod4$resample$Accuracy-mod3$resample$Accuracy)$p.value
,t.test(mod5$resample$Accuracy-mod4$resample$Accuracy)$p.value
,t.test(mod6$resample$Accuracy-mod5$resample$Accuracy)$p.value
,t.test(mod7$resample$Accuracy-mod6$resample$Accuracy)$p.value
,t.test(mod8$resample$Accuracy-mod7$resample$Accuracy)$p.value
,t.test(mod9$resample$Accuracy-mod8$resample$Accuracy)$p.value
,t.test(mod10$resample$Accuracy-mod9$resample$Accuracy)$p.value
)
,names.arg=1:10,main='model significance with respect to previous model',ylab='p-v
alue',xlab='number of important variables')
abline(h=0.05)
```



Appendix C: Final model

Hide

```
# train final model
mod_final<-train(classe~roll_belt+pitch_forearm+pitch_belt+yaw_belt+magnet_dumbbel
l_y,data=pml_training ,method='rf',trControl=train_control,preProcess = c("center"
,"scale"))
```

Hide

```
#get error of final model
1-mean(mod_final$resample$Accuracy)
```