machine learning course project

Farren

05/09/2018

The goal of this project is to predict how well people exercise.

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

```
## Warning in as.POSIXlt.POSIXct(Sys.time()): unknown timezone 'zone/tz/2018e.
## 1.0/zoneinfo/Asia/Singapore'
```

```
## Warning: package 'data.table' was built under R version 3.3.2
```

Preprocessing the data 1. removing the unneccessary variable V1 2. changing the “” to NA, and converting the data to numeric where possible 3. Removing username and date since not useful

```
#training
training <- training %>% select(-V1)
training[training ==""] <- NA
training[training =="#DIV/0!"] <- NA
training <- data.frame(training)
training[,c(6:158)] <- data.frame(apply(training[c(6:158)], 2, as.numeric))
training$new_window <- as.factor(training$new_window)
training$classe <- as.factor(training$classe)
training <- training %>% select(-user_name,-cvtd_timestamp)
#testing
testing <- testing %>% select(-V1)
testing[testing ==""] <- NA
testing[testing =="#DIV/0!"] <- NA
testing <- data.frame(testing)
testing[,c(6:158)] <- data.frame(apply(testing[c(6:158)], 2, as.numeric))
testing$new_window <- as.factor(testing$new_window)
```

Creating a validation set from the training set

```
set.seed(12345)
val_ind <- createDataPartition(y=training$classe, p =0.7, list =FALSE)
validation <- training[-val_ind,]
training <- training[val_ind,]
```

Feature selection 1. Looking at the percentage of NAs. Remove columns where % of NAs >90% 2. Creating dummy variable for the factor var(window) 3. Scale the variable

```r
#% of NAs, and choose the same var for testing set
training <- training %>% select(which(colMeans(is.na(.)) < 0.9))
# Creating dummy var (both test and train)
dummies <- dummyVars(classe~new_window,data=training)
dummyvar <- as.data.frame(predict(dummies,newdata = training))
```

```
## Warning in model.frame.default(Terms, newdata, na.action = na.action, xlev
## = object$lvls): variable 'classe' is not a factor
```

```r
training <- cbind(training, dummyvar)
training <- training %>% select(-new_window)
training_names_woclasse <- training %>% select(-classe) %>% names()
training_names_wclasse <- training %>% names()
#validation
dummies <- dummyVars(factor(classe)~new_window,data=validation)
dummyvar <- as.data.frame(predict(dummies,newdata = validation))
```

```
## Warning in model.frame.default(Terms, newdata, na.action = na.action, xlev
## = object$lvls): variable 'classe' is not a factor
```

```r
validation <- cbind(validation, dummyvar)
validation <- validation %>% select(-new_window)
validation_classe <- validation %>% select(classe)
validation <- validation %>% select(training_names_woclasse)

#testing
testing <- testing %>% mutate(new_window.no = ifelse(new_window=="no",1,0),
                              new_window.yes = ifelse(new_window=="yes",1,0))
testing <- testing %>% select(-new_window)
testing <- testing %>% select(training_names_woclasse)

#scaling the data set for continuous variable and apply to test dataset
preobj <- preProcess(training[,c(-58,-57,-56)], method=c("center","scale"))
#standardized var
pre_proc_train <- predict(preobj,training[,c(-58,-57,-56)])
new_train <- training[,c(56:58)]
new_train <- cbind(new_train, pre_proc_train)
#test set, they will be centered using train mean and train sd
pre_pro_test <- predict(preobj,newdata = testing[,c(-56,-57)])
new_test <- testing[,c(56:57)]
new_test <- cbind(new_test, pre_pro_test)
#validation
pre_pro_val <- predict(preobj,newdata = validation[,c(-56,-57)])
new_val <- validation[,c(56:57)]
new_val <- cbind(new_val, pre_pro_val,validation_classe)
```

Do a random forest to predict

```
#fit a model with the PC components instead of the original train data
modfit <- train(classe~., method ="rf", data = new_train,
              trControl = trainControl(method ="cv"), number =3)
```

Estimated error, using validation set

```
#use random forest to predict and heres the confusion matrix for the validation set
#so we try to use validation
pred_val <- predict(modfit, new_val)
table(pred_val, factor(new_val$classe))
```

```
##
## pred_val    A    B    C    D    E
##        A 1674    0    0    0    0
##        B    0 1139    1    0    0
##        C    0    0 1025    0    0
##        D    0    0    0  964    2
##        E    0    0    0    0 1080
```

```
summary(modfit$finalModel)
```

```
##                 Length Class      Mode
## call                5  -none-     call
## type                1  -none-     character
## predicted       13737  factor     numeric
## err.rate         3000  -none-     numeric
## confusion          30  -none-     numeric
## votes           68685  matrix     numeric
## oob.times       13737  -none-     numeric
## classes             5  -none-     character
## importance         57  -none-     numeric
## importanceSD        0  -none-     NULL
## localImportance     0  -none-     NULL
## proximity           0  -none-     NULL
## ntree               1  -none-     numeric
## mtry                1  -none-     numeric
## forest             14  -none-     list
## y               13737  factor     numeric
## test                0  -none-     NULL
## inbag               0  -none-     NULL
## xNames             57  -none-     character
## problemType         1  -none-     character
## tuneValue           1  data.frame list
## obsLevels           5  -none-     character
## param               1  -none-     list
```

predict on testing data

```
pred_test <- predict(modfit, new_test)
```