Prediction Assignment

Chao Meng, Lei

January 29, 2017

Build the Model

Step 1 - Clean NA and identity columns

```
# Load source data
pml.training <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", na.strings = c("","NA"))
# store the data
training.set <- pml.training

# count how many rows by classes
Record.count_by_classe <- training.set %>% group_by(classe) %>% summarise(n = n())

# count how many NA for each columns by classes
NA.count_by_classe <- training.set %>% group_by(classe) %>% summarise_each(funs(sum(is.na(.))))

# at the beginning, I don't know how many NA per group, or if some NA only exist on some classe but no all classe, then thos
e information may still useful. so the following code is to test if the NA data has exist over 20% per each group. then I wi
ll drop the whole column, otherwise, I will keep it.
NA.proportion = 0.2

#create a emtry dataframe
checkdata <- NA.count_by_classe[0,]

# compare the NA records per each classe with the number of rows per that classes
# for example  classe A there are 5580 rows, there are 5471 NA records from min_pitch_belt in classe A, which is large then
 20%, then it iwll return TRUE.
# if it is return TRUE for all the classe in the column "min_pitch_belt" , then I will drop it.
for (i in 1:nrow(Record.count_by_classe)) {
addrow <- NA.count_by_classe[i,] >= as.numeric(Record.count_by_classe[i,2]) * NA.proportion
checkdata <- rbind(checkdata,addrow) # join the result by each classe
}

# count how many TRUE
NA.rowcount <- checkdata %>% summarise_each(funs(sum(.)))

# list the columns name if the TRUE count equal to the number of group, that means the NA count large then 20% of the rows c
ount in all classe
NA.ColumnName <- colnames(Filter(function(x)all(x==nrow(Record.count_by_classe)), NA.rowcount))

# clean the not useful and NA columns
training.set.1   <- training.set[ , -which(names(training.set) %in% c(
                                                   "X",
                                                   "cvtd_timestamp",
                                                   "new_window",
                                                   "num_window",
                                                   "raw_timestamp_part_2",
                                                   "raw_timestamp_part_1",
                                                   "user_name",
                                                    NA.ColumnName))]
```

Step 2 - Clean the zero variance columns

```
# Then test the variable which has near zero variance, even currently no columns has been drop, but it is a machine learning
 course, in case there is other zero variance variable appear, the if statement is handle the error happen if the zerovar.Co
lumnName is emtry.
zerovar.ColumnName <- nearZeroVar(training.set.1, names = TRUE)

if (length(zerovar.ColumnName)>0)
{
training.set.2   <- training.set.1[ , -which(names(training.set.1) %in% zerovar.ColumnName)]
} else {
training.set.2   <- training.set.1
}
```

Step 3 - Split the training set to build the model and test set for cross validation

```
# we use tree to create the model, the high variance seems not too important
# I don't use PCA

# before put to the model, let's split a test set
# Split a Validation Set form the orignal training set
set.seed(888)
inTrain <- createDataPartition(y=training.set.2 $classe, p =0.75, list =FALSE)
training <- training.set.2[inTrain,]
testing <- training.set.2[-inTrain,]
```

Step 4 - Build a set of tree model using Random Forest

```
# try Random Forests - expect that this model is good at accuary,  performance and interpretability is not very importment i
n this case
mod.Fit <- randomForest(classe~. , data = training.set.2, ntree=50)
```

Step 5 - Cross validation with the testing set

```
# predict the testing set result
testing$result <- predict(mod.Fit,testing)
# cross validation - show confusion matrix
with(testing, table(result,classe))
```

```
##       classe
## result    A    B    C    D    E
##      A 1395    0    0    0    0
##      B    0  949    0    0    0
##      C    0    0  855    0    0
##      D    0    0    0  804    0
##      E    0    0    0    0  901
```

```
# expected out of sample error is
sprintf("%1.1f%%",  sum( testing$result == testing$classe) / nrow(testing) *100)
```

```
## [1] "100.0%"
```

```
# I am very surprise to get 100% accuracy.

# For Quiz   because I haven't do any data transform in this model , just drop some columns, so I can just pass the testing
 dataset to predict function
pml.testing <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv" , na.strings = c("","NA"))
predict(mod.Fit,pml.testing)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```

```
# I am very happy on the quiz result
```