Human Activity Recognition (Assignment)

Rishikesh Pillay

6/20/2021

Overview

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. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. 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. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. 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. In this project, my goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here Steps Involve :

1. Getting and Cleaning Data
2. Subseting data
3. Exploratory Data analysis
4. Model Comparison and selection
5. Conclusion and Prediction

Data sets

The training data for this project are available The test

Getting and Preprocessing Data

I god Data from above links for training and testing . There were too many empty input which I parse to "NA" when Reading data

```
pml_training <-
        read.csv(
                "D:/Science/R_proggrame_coursera/Practical_Machine_Learning/Assignment ML/Data/pml-trai
                ,
                na.strings = c(" ", "", "NA")
        )

pml_testing <-
        read.csv(
```

```
                "D:/Science/R_proggrame_coursera/Practical_Machine_Learning/Assignment ML/Data/pml-testi
                ,
                na.strings = c(" ", "", "NA")
        )
```

while cleaning I converted all numeric Nas in zero.There are 160 variables and almost all of them are numeric.

```
library(lubridate)

pml_training$cvtd_timestamp <-
        as_datetime(pml_training$cvtd_timestamp)
pml_testing$cvtd_timestamp <-
        as_datetime(pml_testing$cvtd_timestamp)
pml_training$classe <-
        as.factor(pml_training$classe)
pml_training[is.na(pml_training)] <- 0
pml_testing[is.na(pml_testing)] <- 0
```

While Preprocessing Data I checked which variable have minimum or near zero variance , and drop them as they were no good for modal training .After this my variable drop from 160 to 59.

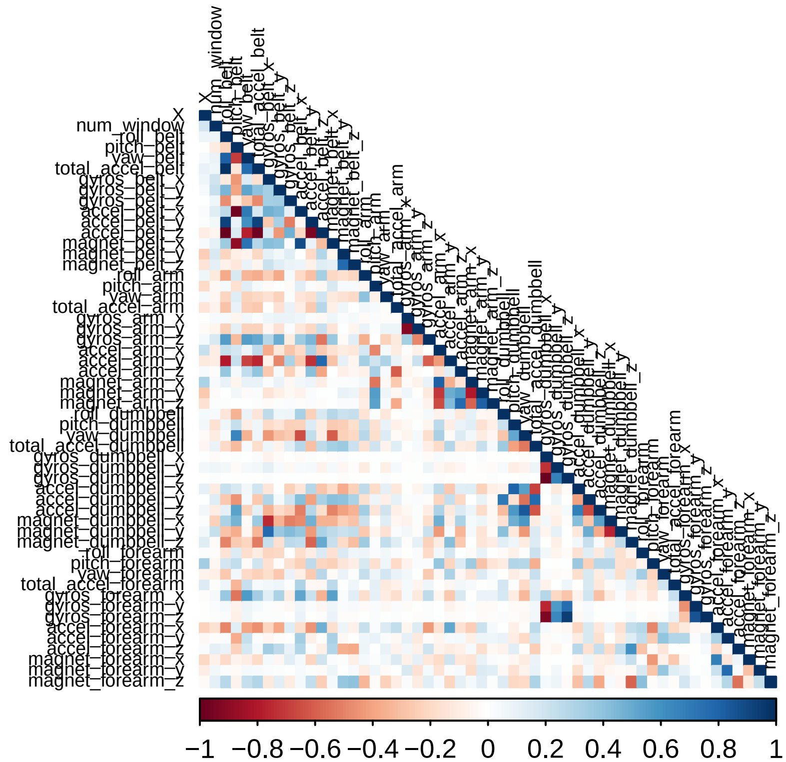
```
library(caret)
set.seed(4321)

NZV <- nearZeroVar(pml_training)

# Data division
inTrain <-
        createDataPartition(pml_training$classe, p = 0.75, list = FALSE)
train <- pml_training[inTrain, -NZV]
valid <- pml_training[-inTrain, -NZV]
```

For next step I divided training data(19622 obs) into two groups train and valid . I will use valid data to check out of sample error . And in end use Test data to answer final prediction.

Exploratory Data anlysis

My train data has dimension 14718 observation over 59 variables Validation data has 4904 observation Now check correlation of data over other variables

```
library(corrplot)
mat <- cor(train[-c(2, 3, 4, 5, 59)])
corrplot(
        mat,
        type = "lower",
        method = "color",
        tl.cex = 0.6,
        tl.col = rgb(0, 0, 0)
)
```



Darker colour indicates higher correlation .I did heatmap and summary on data but since its large data it was not much help .You can find them in my github repo. There are 5 type of Classe “A”,“B”,“C”,“D”, “E”

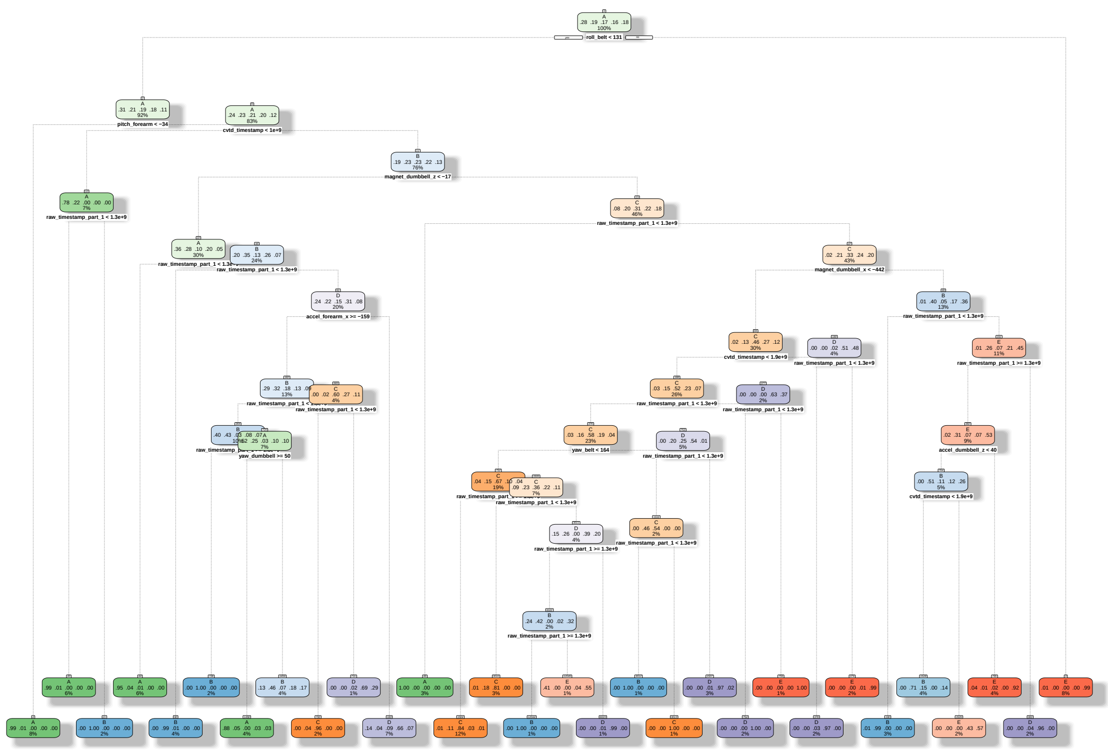
Model selection

I am using or rather checking accuracy against valid data and choose the best one.

1. Decision Tree
2. Random Forest
3. Generalized Boosted Model

Decision Tree

```
set.seed(534)
library(rpart)
library(rattle)
modDT <- rpart(classe ~ ., data  = train[-1], method = 'class')
fancyRpartPlot(modDT)
```



Rattle 2021−Jun−21 09:35:47 Nilesh.Pillay

```r
predDT <- predict(modDT, valid, type = "class")
conmat1 <- confusionMatrix(predDT, valid$classe)
acc1 <- conmat1$overall["Accuracy"]
```

Accuracy of decision tree `0.8809135` on valid data ## Random forest

```r
set.seed(553)
library(randomForest)
modRF <- randomForest(classe ~ . , data = train[-1])

#prediction
predRF <- predict(modRF, valid, type = "class")
conmat2 <- confusionMatrix(predRF, valid$classe)
acc2 <- conmat2$overall["Accuracy"]
acc2
```

```
##  Accuracy
## 0.9977569
```

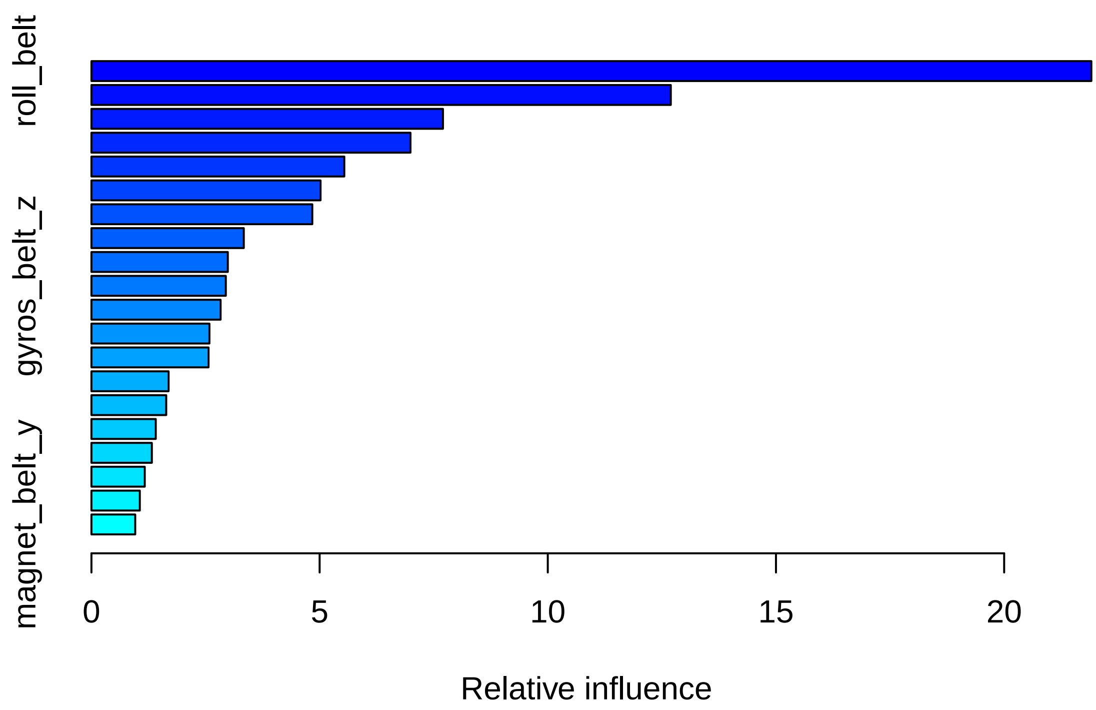
Accuracy of Random forest `0.9977569` on valid data

Gradiant Boosting Model

```r
library(gbm)
set.seed(332)
modGbm <- gbm(classe ~ ., data = train[-c(1, 2, 3, 4, 5)])
```

```
## Distribution not specified, assuming multinomial ...
```

```r
summary(modGbm, cBars = 20)
```




```
##                                    var     rel.inf
## roll_belt                    roll_belt 21.91092722
## pitch_forearm            pitch_forearm 12.69549691
## num_window                  num_window  7.70301402
## magnet_dumbbell_y    magnet_dumbbell_y  6.99033817
## yaw_belt                      yaw_belt  5.54070655
## roll_forearm              roll_forearm  5.01975836
## magnet_dumbbell_z    magnet_dumbbell_z  4.83958962
## pitch_belt                  pitch_belt  3.33855254
## gyros_dumbbell_y      gyros_dumbbell_y  2.98808575
## gyros_belt_z              gyros_belt_z  2.94447881
## magnet_belt_z            magnet_belt_z  2.83016808
## roll_dumbbell            roll_dumbbell  2.58525921
## accel_forearm_x        accel_forearm_x  2.56602970
## accel_dumbbell_y      accel_dumbbell_y  1.69070897
## magnet_arm_x              magnet_arm_x  1.63851133
```

```
## accel_dumbbell_x         accel_dumbbell_x  1.40963185
## magnet_arm_z                 magnet_arm_z  1.32326944
## magnet_forearm_z         magnet_forearm_z  1.16819965
## magnet_dumbbell_x       magnet_dumbbell_x  1.05985728
## magnet_belt_y               magnet_belt_y  0.95932835
## yaw_arm                           yaw_arm  0.91706574
## total_accel_dumbbell total_accel_dumbbell  0.66972791
## magnet_arm_y                 magnet_arm_y  0.65073076
## gyros_belt_y                 gyros_belt_y  0.64110463
## accel_belt_z                 accel_belt_z  0.62458444
## roll_arm                         roll_arm  0.59313420
## gyros_arm_y                   gyros_arm_y  0.58485408
## pitch_dumbbell             pitch_dumbbell  0.57432895
## magnet_forearm_x         magnet_forearm_x  0.52885856
## accel_dumbbell_z         accel_dumbbell_z  0.48604353
## gyros_dumbbell_x         gyros_dumbbell_x  0.42566562
## magnet_forearm_y         magnet_forearm_y  0.32554843
## total_accel_forearm   total_accel_forearm  0.32060437
## gyros_dumbbell_z         gyros_dumbbell_z  0.29009032
## gyros_arm_x                   gyros_arm_x  0.25590463
## gyros_forearm_z           gyros_forearm_z  0.19606663
## accel_arm_z                   accel_arm_z  0.19519221
## gyros_forearm_y           gyros_forearm_y  0.16838295
## magnet_belt_x               magnet_belt_x  0.15672422
## accel_arm_x                   accel_arm_x  0.09191966
## accel_belt_x                 accel_belt_x  0.05765927
## accel_forearm_y           accel_forearm_y  0.04389712
## total_accel_belt         total_accel_belt  0.00000000
## gyros_belt_x                 gyros_belt_x  0.00000000
## accel_belt_y                 accel_belt_y  0.00000000
## pitch_arm                       pitch_arm  0.00000000
## total_accel_arm           total_accel_arm  0.00000000
## gyros_arm_z                   gyros_arm_z  0.00000000
## accel_arm_y                   accel_arm_y  0.00000000
## yaw_dumbbell                 yaw_dumbbell  0.00000000
## yaw_forearm                   yaw_forearm  0.00000000
## gyros_forearm_x           gyros_forearm_x  0.00000000
## accel_forearm_z           accel_forearm_z  0.00000000
```

```r
predGBM <- predict.gbm(modGbm, valid, type = "response")
labels = colnames(predGBM)[apply(predGBM, 1, which.max)]
conmat3 <- confusionMatrix(as.factor(labels), valid$classe)
acc3 <-  conmat3$overall['Accuracy']
acc3
```

```
##  Accuracy
## 0.8150489
```

The predicted result is not easy-readable data so we'll get class names with the highest prediction value. Accuracy of Gradient Boosting Model `0.8150489` on valid data

Conclusion

accuracy = weights false positives/negatives equal

```
##   model       acc
## 1    DT 0.8809135
## 2    RF 0.9977569
## 3   GBM 0.8150489
```

- As Random forest have large accuracy I will select that model for prediction on test data set
- Gbm model is good for inference for which features are good for modeling (These do not refer to the variance.)

Prediction

```
library(randomForest)
prediction <- predict(modRF, pml_testing, type = "class")
prediction
```

```
##  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 could have used stacking of prediction but I didn't as Random forest gives good prediction.
- You have notice I used randomforest and gbm packages instead of caret ,because train function was taking too much time and computing power for large training dataset
- You can find other plots like heatmap in outputs folder in github repo I didn't included it here since it was not giving relevant information