Practial Machine Learning Assignment_1

Jitender Kumar

May 25, 2016

Synopsis

Objective and Goal

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, the goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants.

The objective of the project is to predict the manner in which participants did the exercise. This is the “classe” variable in the training set.

Objectives

- To create a report describing how the model is built
- How to pre process and use cross validation
- Interpretation of expected out of sample error
- Appropiate reasons of the choices made and finally
- Use the prediction model to predict 20 different test cases.

Background and Data

Background

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, your 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: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har).

Loading and Pre processing data

```
pml_train = read.csv("pml-training.csv", header = TRUE)
pml_test = read.csv("pml-testing.csv", header = TRUE)

dim(pml_train)
```

```
## [1] 19622   160
```

```
dim(pml_test)
```

```
## [1]  20 160
```

Pre processing data

We will clean and transform the data set

1. Removing the first 6 colums which would not be used for prediction

```
pml_train<-pml_train[,-c(1:6)]
```

2. Near Zero values processing

```
nzv <- nearZeroVar(pml_train, saveMetrics=TRUE)
pml_train<-pml_train[,nzv$nzv== FALSE]
```

3. NA check for columns, removing colums with more than 70% of values as NA

```
NAcols<- as.data.frame(colSums(is.na(pml_train)/nrow(pml_train)))
NAcols<- subset(NAcols, NAcols<0.7)
pml_train <- pml_train[,rownames(NAcols)]
dim(pml_train)
```

```
## [1] 19622    54
```

4. Correlation check of data set

```
Corr_cols<- findCorrelation(cor(pml_train[, -54]), cutoff=0.8)
colnames(pml_train)[Corr_cols]
```

```
##  [1] "accel_belt_z"     "roll_belt"        "accel_belt_y"
##  [4] "accel_dumbbell_z" "accel_belt_x"     "pitch_belt"
##  [7] "accel_arm_x"      "accel_dumbbell_x" "magnet_arm_y"
## [10] "gyros_dumbbell_x" "gyros_forearm_y"  "gyros_dumbbell_z"
## [13] "gyros_arm_x"
```

```
length(Corr_cols)/ncol(pml_train)
```

```
## [1] 0.2407407
```

There are 13 columns (24%) which are correlated with each other.We need to use pca for pre processing when fitting a prediciton model.

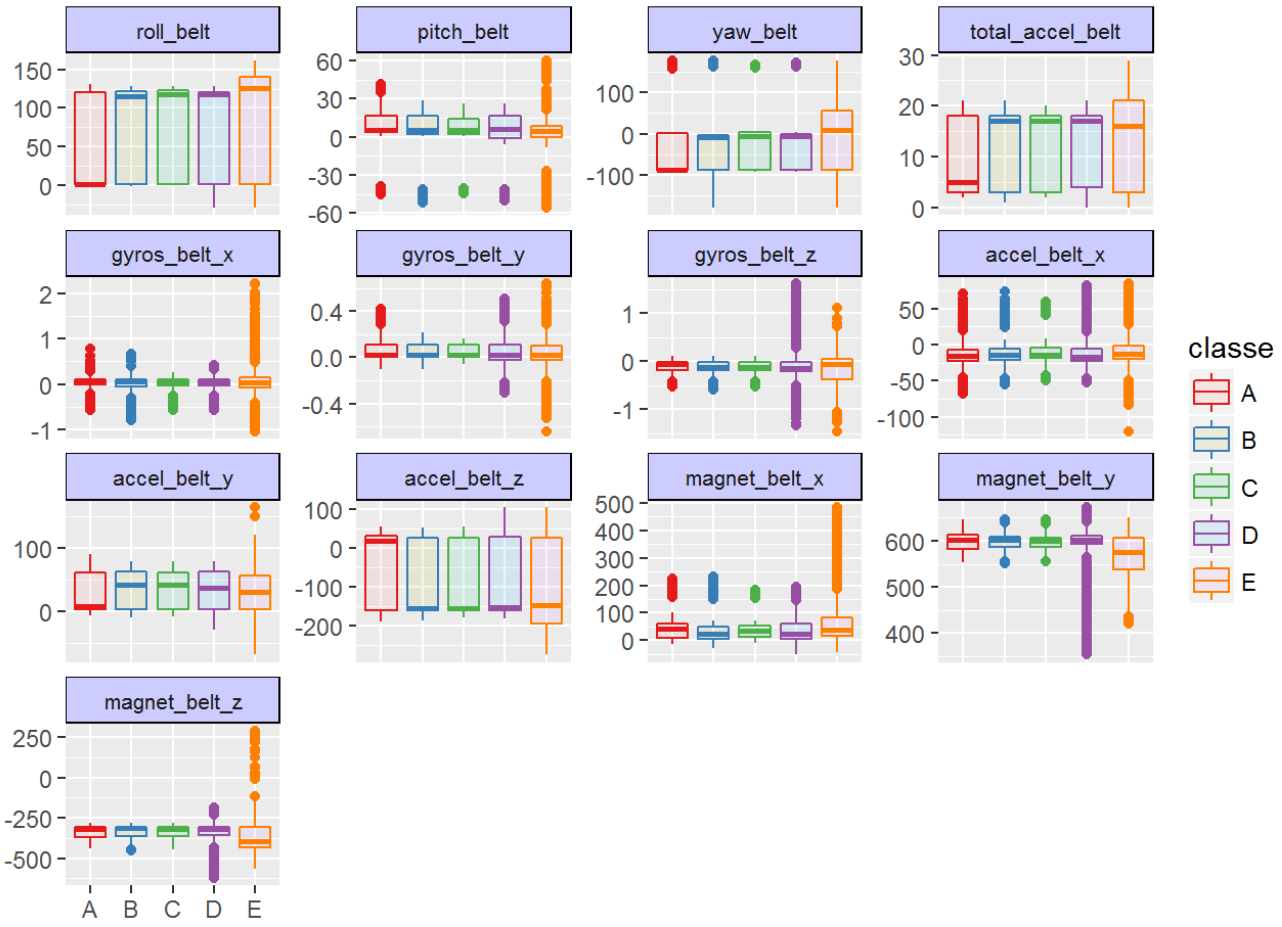
Exploring data sets

```
plot_sensor<- function(sensor){
sensor_col<- c(grep(sensor, colnames(pml_train)), ncol(pml_train))
plot_data<- melt(pml_train[,sensor_col])

ggplot(plot_data, aes(x=classe, y=value)) +
  geom_boxplot(aes(color=classe, fill=classe), alpha=1/10) +
  facet_wrap(~ variable, scale="free_y") +
  scale_color_brewer(palette="Set1") +
  labs(x="", y="") +
  theme(strip.text.x = element_text(size=8),
         strip.text.y = element_text(size=12, face="bold"),
         strip.background = element_rect(colour="black", fill="#CCCCFF") )
}
```

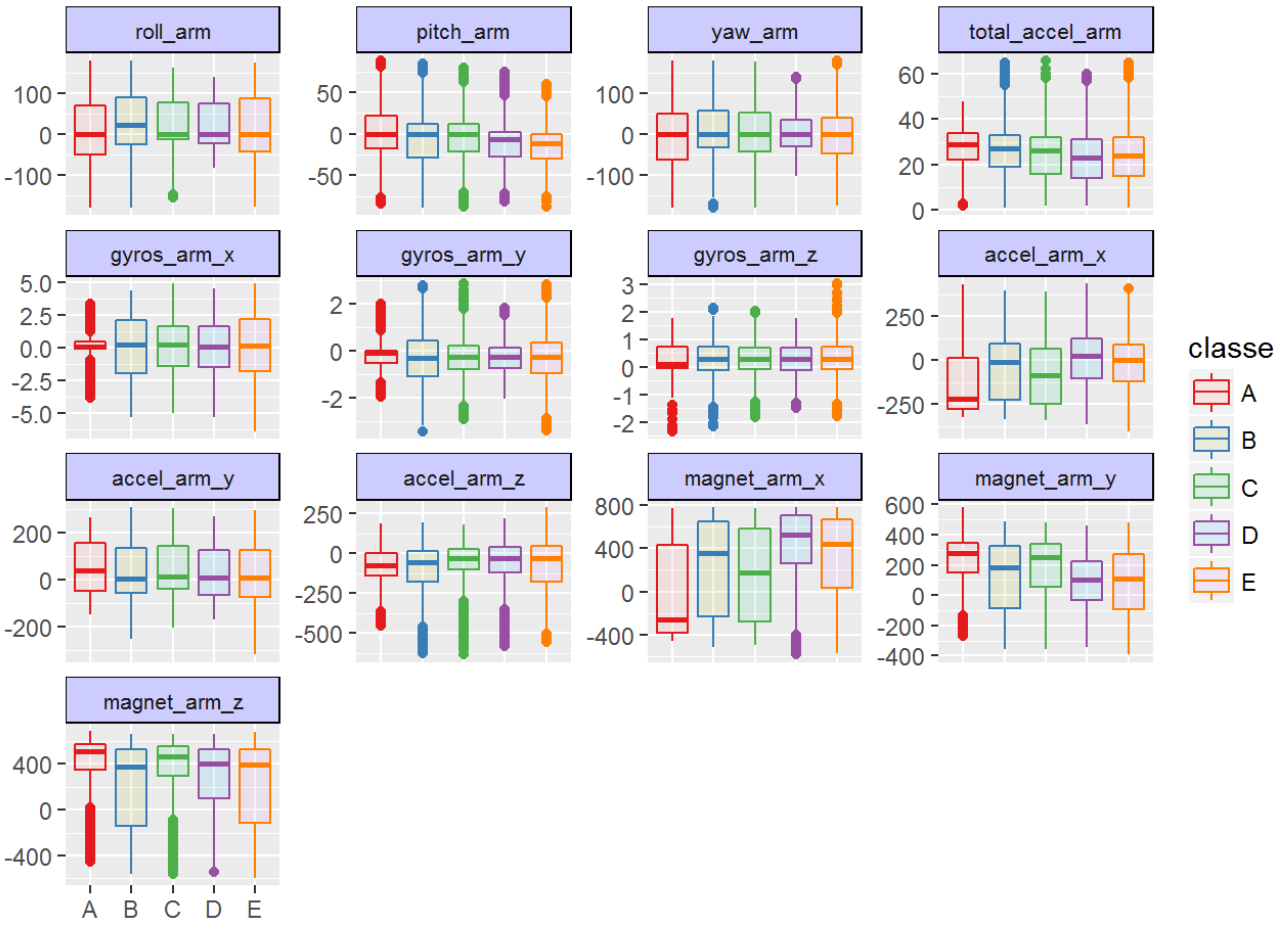
```
plot_sensor("belt")
```

```
## Using classe as id variables
```



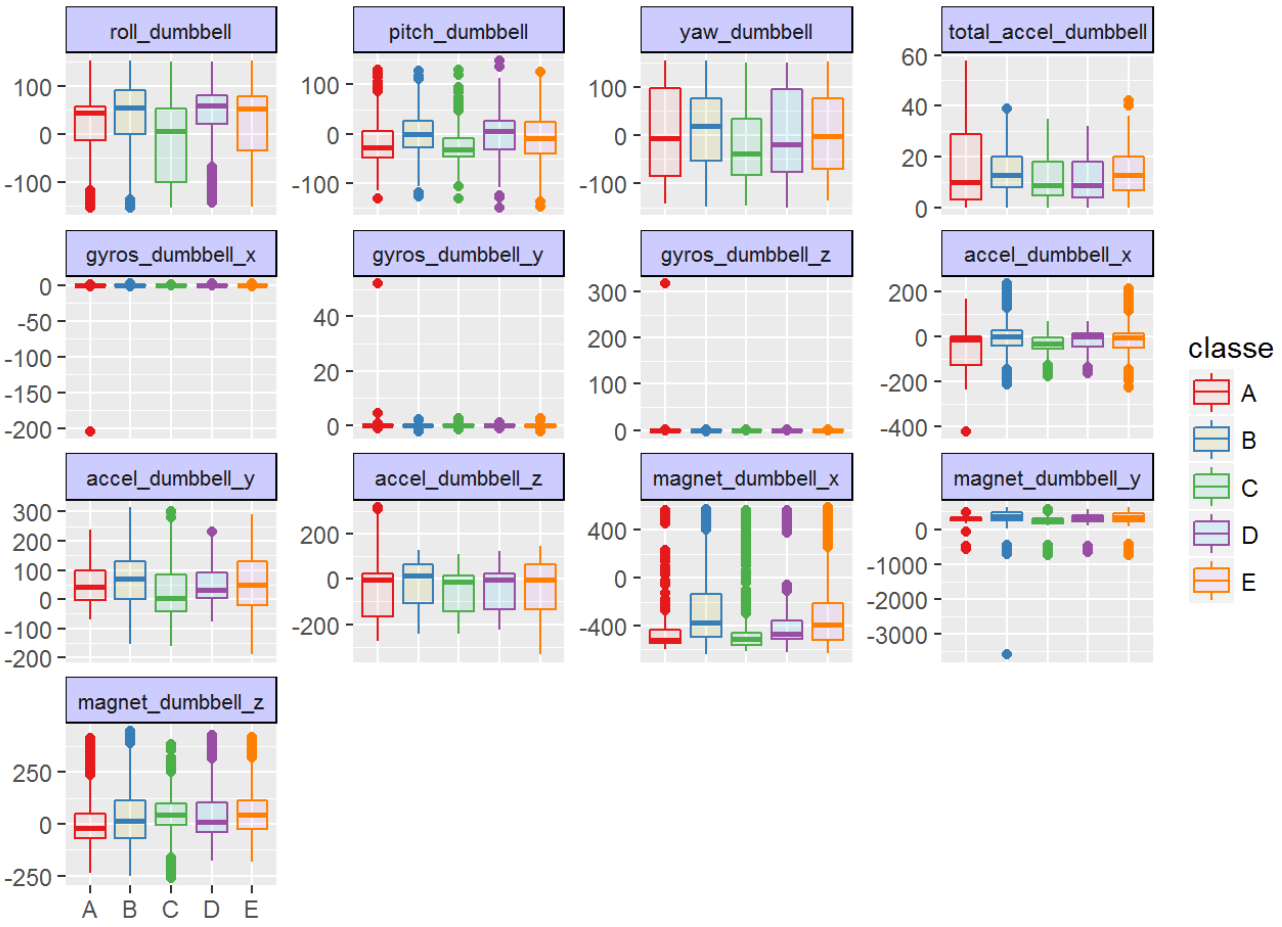

```
plot_sensor("[^(fore)]arm")
```

```
## Using classe as id variables
```



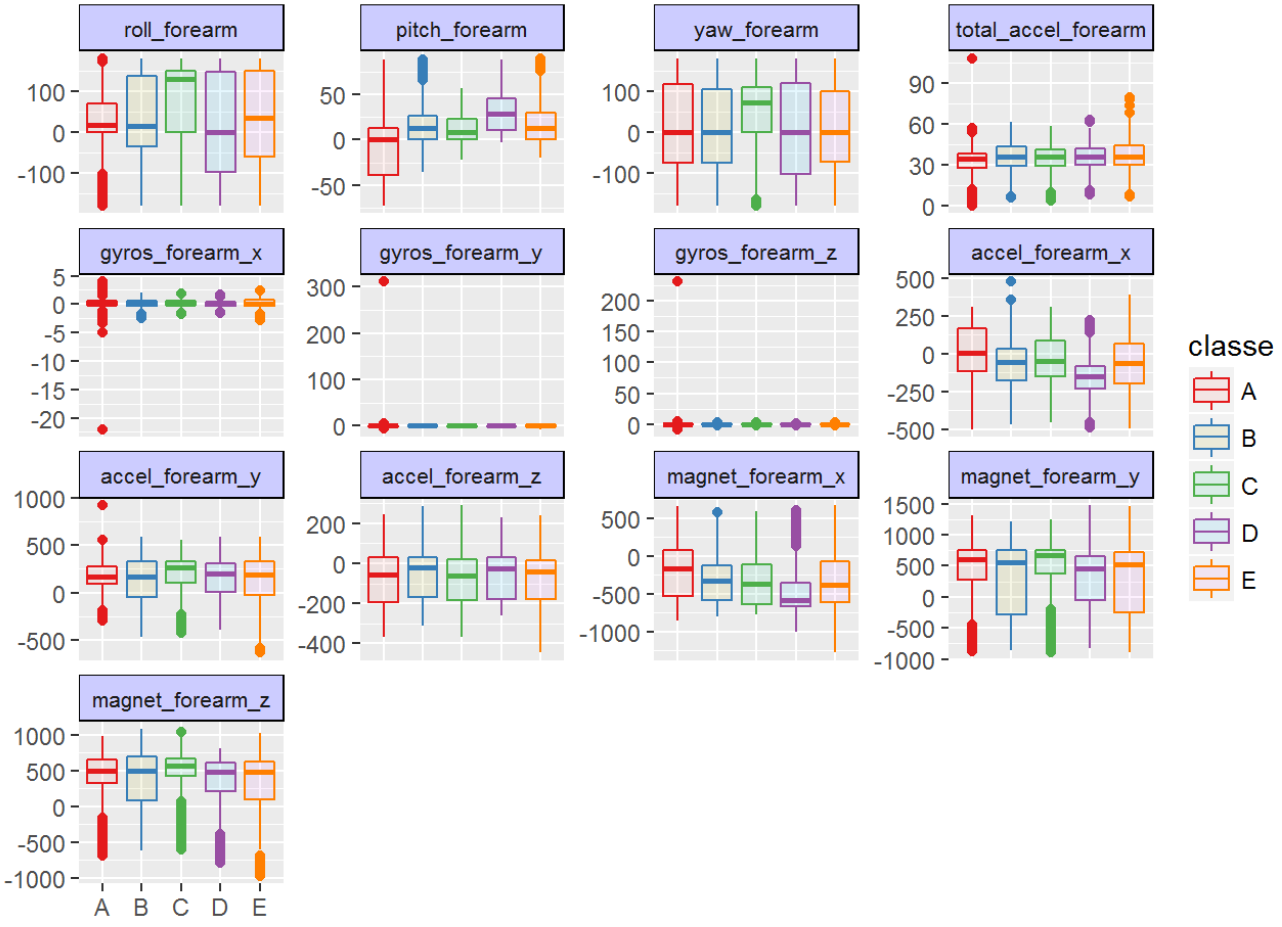

```
plot_sensor("dumbbell")
```

```
## Using classe as id variables
```




```
plot_sensor("forearm")
```

```
## Using classe as id variables
```



Processing the test data

Doing the data transformation with test data set

```
pml_test<-pml_test[,-c(1:6)]

nzv_test <- nearZeroVar(pml_test, saveMetrics=TRUE)
pml_test<-pml_test[,nzv_test$nzv== FALSE]

NAcols_test<- as.data.frame(colSums(is.na(pml_test)/nrow(pml_test)))
NAcols_test<- subset(NAcols_test, NAcols_test<0.7)
pml_test <- pml_test[,rownames(NAcols_test)]
dim(pml_test)
```

```
## [1] 20 54
```

Model building, selection and Cross Validation

Building training and probe data set

```
inTrain = createDataPartition(pml_train$classe, p = 0.7, list=FALSE)
data_training = pml_train[inTrain,]
data_probe = pml_train[-inTrain,]
```

PCA Processing and Cross validation

Using the trainControl function for

- 7 fold Cross Validation
- PCA for pre prosessing

```
train_trControl <- trainControl(method = "cv", number = 7,  preProcOptions="pca")
```

Model 1: Tree method for prediction

```
mod_tree<- train(classe~., data = data_training, method = "rpart", trControl= train_trC
ontrol)
```

Model 2: GBM method for prediction

```
mod_gbm <- train(classe ~ ., data = data_training, method = "gbm", trControl= train_trC
ontrol, verbose = FALSE)
```

Model 3: Randon Forest method for prediction

```
mod_rf <- train(classe ~ ., data = data_training, method = "rf", trControl= train_trCon
trol)
```

Comparing the models

Using highest accurary for comparision

```
Type_model<- c("Tree","GBM", "Random Forest")
Accuracy_model<- c(max(mod_tree$results$Accuracy),max(mod_gbm$results$Accuracy), max(mo
d_rf$results$Accuracy))
model_compare<- cbind(Type_model, Accuracy_model)
kable(model_compare)
```

Type_model	Accuracy_model
Tree	0.590159170814132
GBM	0.987988579458292
Random Forest	0.997233551469063

The Random Forest model gives a better performance though GBM model is very close.

Validation on probe data set

```
validation_probe <- predict(mod_rf, data_probe)
confusionMatrix(validation_probe, data_probe$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    1    0    0    0
##          B    0 1135    0    0    2
##          C    0    2 1026    4    0
##          D    0    1    0  960    2
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.998
##                  95% CI : (0.9964, 0.9989)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9974
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9965   1.0000   0.9959   0.9963
## Specificity            0.9998   0.9996   0.9988   0.9994   1.0000
## Pos Pred Value         0.9994   0.9982   0.9942   0.9969   1.0000
## Neg Pred Value         1.0000   0.9992   1.0000   0.9992   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1929   0.1743   0.1631   0.1832
## Detection Prevalence   0.2846   0.1932   0.1754   0.1636   0.1832
## Balanced Accuracy      0.9999   0.9980   0.9994   0.9976   0.9982
```

Out of sample error from probe data set

```
Out_sample_error<- (1- confusionMatrix(validation_probe,data_probe$classe)$overall
[[1]])
print(Out_sample_error,digits= 4)
```

```
## [1] 0.002039
```

Final Model

```
mod_rf$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.2%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3905    0    0    0    1 0.0002560164
## B    5 2649    4    0    0 0.0033860045
## C    0    5 2391    0    0 0.0020868114
## D    0    0    6 2245    1 0.0031083481
## E    0    0    0    6 2519 0.0023762376
```

Error rate is less than 1%

Prediction on test data set

The Random Forest model will be used for prediction of classe for test data set

```
test_predict <- predict(mod_rf, pml_test)
print(cbind(pml_test[54],test_predict))
```

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

This completes the assignment for Practical Machine Learning course.