Peer-graded Assignment: Prediction Assignment Writeup

Abdugappaova Kamila

17/04/2024

Summary

This report serves as the culmination of the Peer Assessment project within the Practical Machine Learning course, a component of the Coursera John's Hopkins University Data Science Specialization. Developed in RStudio, the report utilizes knitr functions to generate both HTML and Markdown formats. The primary objective of the project is to forecast the manner in which six participants executed a series of exercises. Employing a machine learning algorithm trained on the 'classe' variable in the training set, predictions are generated for the 20 test cases provided in the test data.

Introduction

In this project, we aim to leverage data gathered from accelerometers embedded in devices like Jawbone Up, Nike FuelBand, and Fitbit. These devices are instrumental in the quantified self movement, where individuals track various metrics to enhance their health, identify behavioral patterns, or simply out of curiosity for technology. While people often quantify the extent of their activities, they rarely assess their performance quality. Therefore, our objective is to analyze data collected from accelerometers placed on the belt, forearm, arm, and dumbell of six participants. These participants were instructed to execute barbell lifts in both correct and incorrect manners across five different variations.

More information is available from the website here: http://groupware.les.inf.puc-rio.br/har.

Data Source

The training and test data for this project are collected using the link below:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.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.

The full reference of this data is as follows:

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. **"Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13)"**. Stuttgart, Germany: ACM SIGCHI, 2013.

Loading and Cleaning of Data

Set working directory.

```
setwd("~/Documents/RProgramming Reference/courses-master/08 PracticalMachineL
earning/027forecasting")
```

Load required R packages and set a seed.

```
library(lattice)
library(ggplot2)
library(caret)
library(rpart)
library(rpart.plot)
library(corrplot)
library(rattle)
library(randomForest)
library(RColorBrewer)

set.seed(222)
```

Load data for training and test datasets.

```
url_train <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training
.csv"
url_quiz  <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.
csv"

data_train <- read.csv(url(url_train), strip.white = TRUE, na.strings = c("NA
",""))
data_quiz  <- read.csv(url(url_quiz),  strip.white = TRUE, na.strings = c("NA
",""))

dim(data_train)
## [1] 19622   160
dim(data_quiz)
## [1]  20 160
```

Create two partitions (75% and 25%) within the original training dataset.

```
in_train  <- createDataPartition(data_train$classe, p=0.75, list=FALSE)
train_set <- data_train[ in_train, ]
test_set  <- data_train[-in_train, ]

dim(train_set)
## [1] 14718   160
dim(test_set)
## [1] 4904  160
```

The two datasets (`train_set` and `test_set`) have a large number of `NA` values as well as near-zero-variance (NZV) variables. Both will be removed together with their ID variables.

```
nzv_var <- nearZeroVar(train_set)


train_set <- train_set[ , -nzv_var]
test_set  <- test_set [ , -nzv_var]


dim(train_set)
[1] 14718   120
dim(test_set)
[1] 4904  120
```

Remove variables that are mostly NA. A threshlod of 95 % is selected.

```
na_var <- sapply(train_set, function(x) mean(is.na(x))) > 0.95
train_set <- train_set[ , na_var == FALSE]
test_set  <- test_set [ , na_var == FALSE]


dim(train_set)
[1] 14718    59
dim(test_set)
[1] 4904   59
```

Since columns 1 to 5 are identification variables only, they will be removed as well.

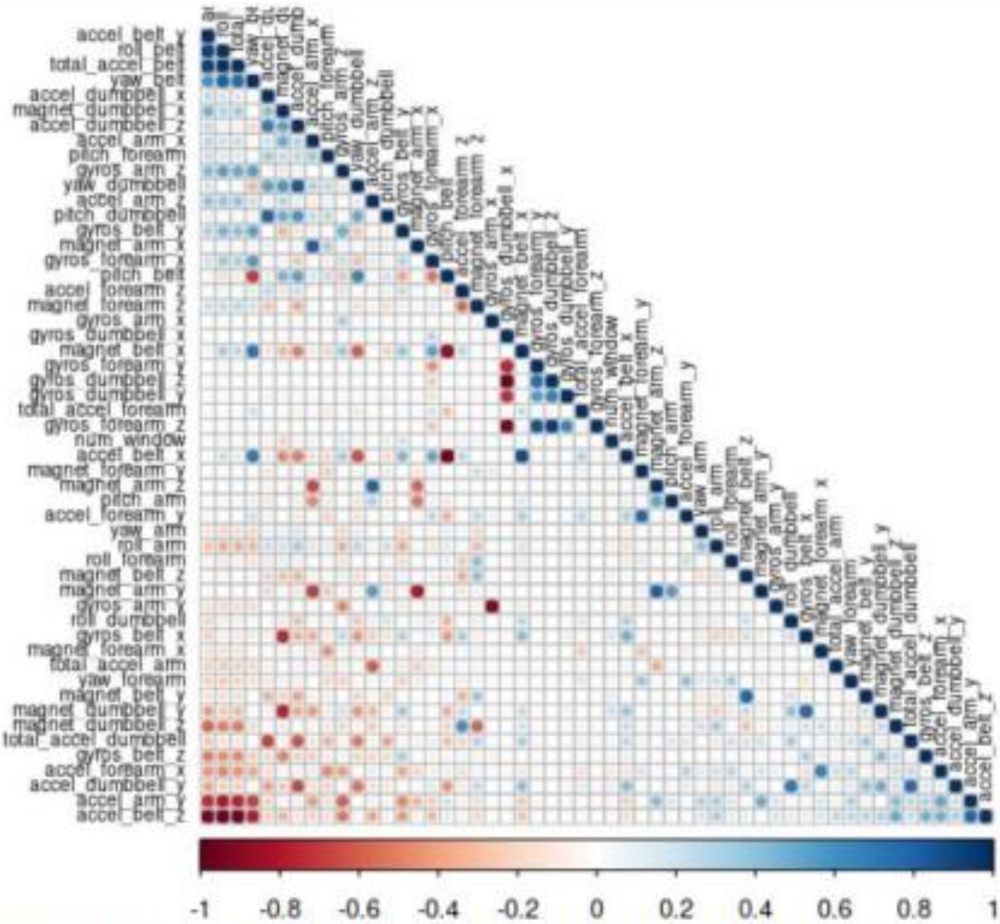
```
train_set <- train_set[ , -(1:5)]
test_set  <- test_set [ , -(1:5)]


dim(train_set)
[1] 14718    54
dim(test_set)
[1] 4904   54
```

The number of variables for the analysis has been reduced from the original 160 down to 54.

Correlation Analysis

Correlation analysis between the variables before the modeling work itself is done. The "FPC" is used as the first principal component order.

```
corr_matrix <- cor(train_set[ , -54])
corrplot(corr_matrix, order = "FPC", method = "circle", type = "lower",
         tl.cex = 0.6, tl.col = rgb(0, 0, 0))
```

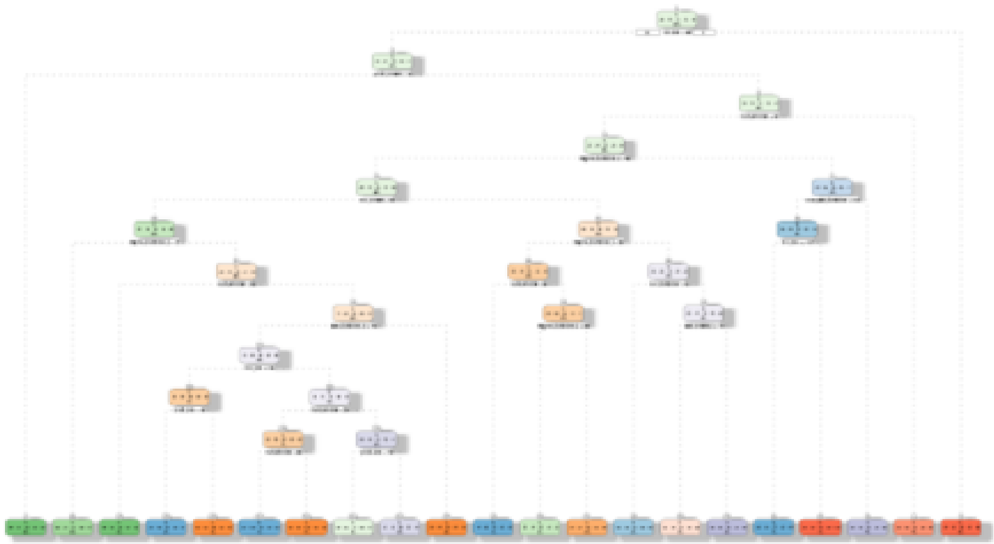


When two variables exhibit high correlation, they are depicted with either dark blue (for positive correlation) or dark red (for negative correlation). However, due to the limited presence of strong correlations among the input variables, Principal Components Analysis (PCA) will not be utilized in this analysis. Instead, we will construct several prediction models to enhance accuracy.

Prediction Models

Decision Tree Model

```
set.seed(2222)
fit_decision_tree <- rpart(classe ~ ., data = train_set, method="class")
fancyRpartPlot(fit_decision_tree)
```



Predictions of the decision tree model on `test_set`.

```
predict_decision_tree <- predict(fit_decision_tree, newdata = test_set, type=
"class")

conf_matrix_decision_tree <- confusionMatrix(predict_decision_tree, factor(te
st_set$classe))

conf_matrix_decision_tree
```

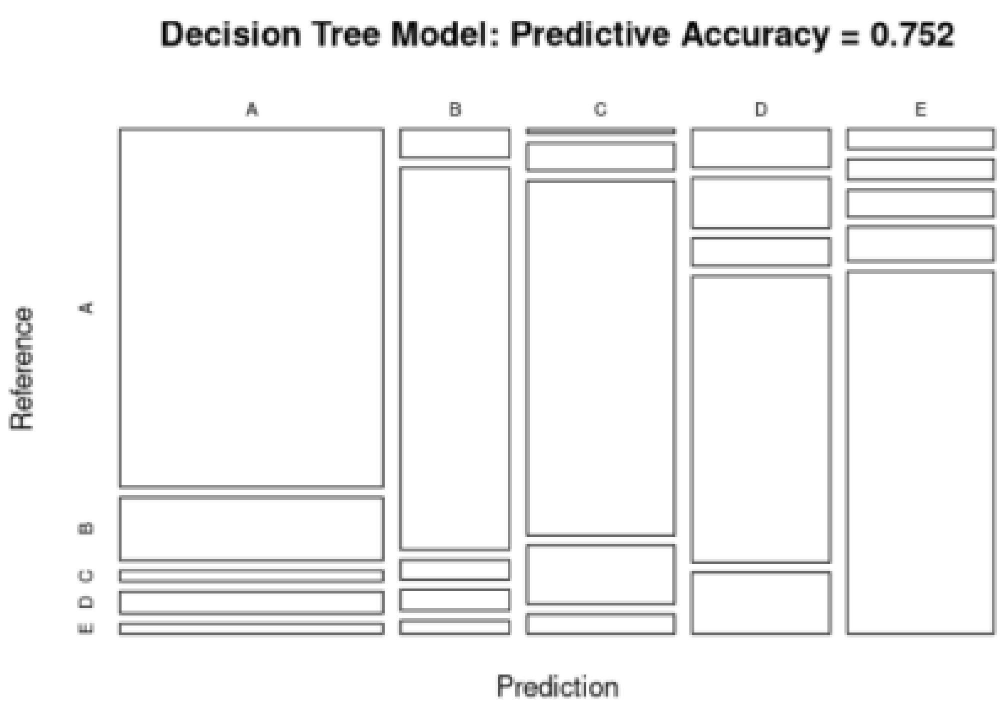
```
Confusion Matrix and Statistics


          Reference

Prediction    A    B    C    D    E

         A 1238  218   37   76   36

         B   41  547   28   30   19

         C    8   53  688  114   38

         D   70   91   50  518  111

         E   38   40   52   66  697


Overall Statistics


               Accuracy : 0.752

                 95% CI : (0.7397, 0.7641)

    No Information Rate : 0.2845

    P-Value [Acc > NIR] : < 2.2e-16


                  Kappa : 0.685
```

```
 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.8875   0.5764   0.8047   0.6443   0.7736
Specificity            0.8954   0.9702   0.9474   0.9215   0.9510
Pos Pred Value         0.7713   0.8226   0.7636   0.6167   0.7805
Neg Pred Value         0.9524   0.9052   0.9583   0.9296   0.9491
Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
Detection Rate         0.2524   0.1115   0.1403   0.1056   0.1421
Detection Prevalence   0.3273   0.1356   0.1837   0.1713   0.1821
Balanced Accuracy      0.8914   0.7733   0.8760   0.7829   0.8623
```

The predictive accuracy of the decision tree model is relatively low at `75.2 %`.

Plot the predictive accuracy of the decision tree model.

```
plot(conf_matrix_decision_tree$table, col = conf_matrix_decision_tree$byClass
,
     main = paste("Decision Tree Model: Predictive Accuracy =",
                  round(conf_matrix_decision_tree$overall['Accuracy'], 4)))
```



Generalized Boosted Model (GBM)

```
set.seed(2222)
ctrl_GBM <- trainControl(method = "repeatedcv", number = 5, repeats = 2)
fit_GBM  <- train(classe ~ ., data = train_set, method = "gbm",
                  trControl = ctrl_GBM, verbose = FALSE)
fit_GBM$finalModel
```

```
A gradient boosted model with multinomial loss function.
150 iterations were performed.
There were 53 predictors of which 52 had non-zero influence.
```

Predictions of the GBM on `test_set`.

```
predict_GBM <- predict(fit_GBM, newdata = test_set)
conf_matrix_GBM <- confusionMatrix(predict_GBM, factor(test_set$classe))
conf_matrix_GBM
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1392    6    0    1    0
         B    3  927    4    3    3
         C    0   12  842   12    2
         D    0    4    9  786    9
         E    0    0    0    2  887

Overall Statistics

               Accuracy : 0.9857
                 95% CI : (0.982, 0.9889)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9819

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
```

```
Sensitivity            0.9978   0.9768   0.9848   0.9776   0.9845
Specificity            0.9980   0.9967   0.9936   0.9946   0.9995
Pos Pred Value         0.9950   0.9862   0.9700   0.9728   0.9978
Neg Pred Value         0.9991   0.9945   0.9968   0.9956   0.9965
Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
Detection Rate         0.2838   0.1890   0.1717   0.1603   0.1809
Detection Prevalence   0.2853   0.1917   0.1770   0.1648   0.1813
Balanced Accuracy      0.9979   0.9868   0.9892   0.9861   0.9920
```

The predictive accuracy of the GBM is relatively high at `98.57 %`.

Random Forest Model

```
set.seed(2222)
ctrl_RF <- trainControl(method = "repeatedcv", number = 5, repeats = 2)
fit_RF  <- train(classe ~ ., data = train_set, method = "rf",
                  trControl = ctrl_RF, verbose = FALSE)
fit_RF$finalModel
```

```

Call:
 randomForest(x = x, y = y, mtry = param$mtry, verbose = FALSE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 27

        OOB estimate of  error rate: 0.24%
Confusion matrix:
     A    B    C    D    E  class.error
A 4183    1    0    0    1 0.0004778973
B    8 2836    3    1    0 0.0042134831
C    0    6 2561    0    0 0.0023373588
D    0    0    7 2404    1 0.0033167496
E    0    1    0    7 2698 0.0029563932
```

Predictions of the random forest model on `test_set`.

```
predict_RF <- predict(fit_RF, newdata = test_set)
conf_matrix_RF <- confusionMatrix(predict_RF, factor(test_set$classe))
conf_matrix_RF
```

```
Confusion Matrix and Statistics
```

```
          Reference
Prediction    A    B    C    D    E
         A 1395    3    0    0    0
         B    0  946    2    0    0
         C    0    0  853    6    0
         D    0    0    0  798    1
         E    0    0    0    0  900

Overall Statistics

               Accuracy : 0.9976
                 95% CI : (0.9957, 0.9987)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9969

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9968   0.9977   0.9925   0.9989
Specificity            0.9991   0.9995   0.9985   0.9998   1.0000
Pos Pred Value         0.9979   0.9979   0.9930   0.9987   1.0000
Neg Pred Value         1.0000   0.9992   0.9995   0.9985   0.9998
Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
Detection Rate         0.2845   0.1929   0.1739   0.1627   0.1835
Detection Prevalence   0.2851   0.1933   0.1752   0.1629   0.1835
Balanced Accuracy      0.9996   0.9982   0.9981   0.9961   0.9994
```

The predictive accuracy of the Random Forest model is excellent at `99.8 %`.

Applying the Best Predictive Model to the Test Data

The following are the predictive accuracy of the three models:

- Decision Tree Model: **75.20 %**

- Generalized Boosted Model: **98.57 %**
- Random Forest Model: **99.80 %**

The Random Forest model is selected and applied to make predictions on the 20 data points from the original testing dataset (data_quiz).

```
predict_quiz <- as.data.frame(predict(fit_RF, newdata = data_quiz))
predict_quiz
```

```
   predict(fit_RF, newdata = data_quiz)
1                                     B
2                                     A
3                                     B
4                                     A
5                                     A
6                                     E
7                                     D
8                                     B
9                                     A
10                                    A
11                                    B
12                                    C
13                                    B
14                                    A
15                                    E
16                                    E
17                                    A
18                                    B
19                                    B
20                                    B
```