Practical Machine Learning Project

Marek Chadim

2023-12-23

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).

Loading and Cleaning of Data

Load

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

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

dim(data_train)
```

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

```
dim(data_test)
```

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

Split

```
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
```

Remove NZV, NAs and ID

```
nzv_var <- nearZeroVar(train_set)

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

dim(train_set)
```

```
## [1] 14718   122
```

```
dim(test_set)
```

```
## [1] 4904  122
```

```
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
```

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

dim(train_set)
```

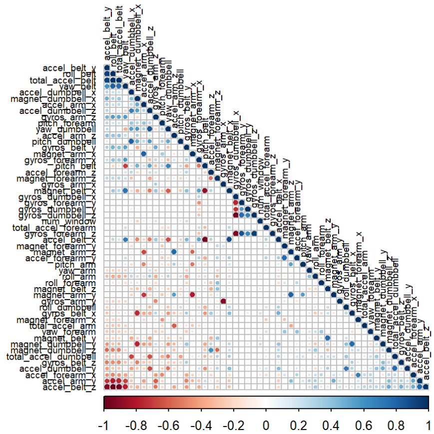
```
## [1] 14718    54
```

```
dim(test_set)
```

```
## [1] 4904   54
```

Correlations

```
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))
```

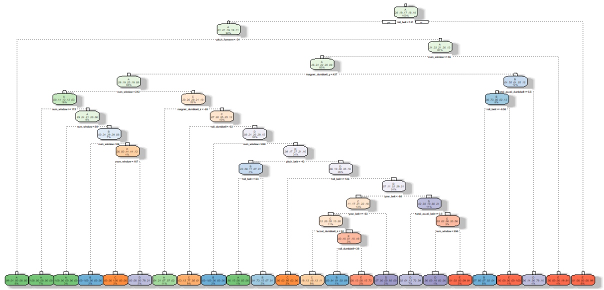


Modelling

Decision Tree Model

```
set.seed(42)
fit_DT <- rpart(classe ~ ., data = train_set, method="class")
fancyRpartPlot(fit_DT)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```

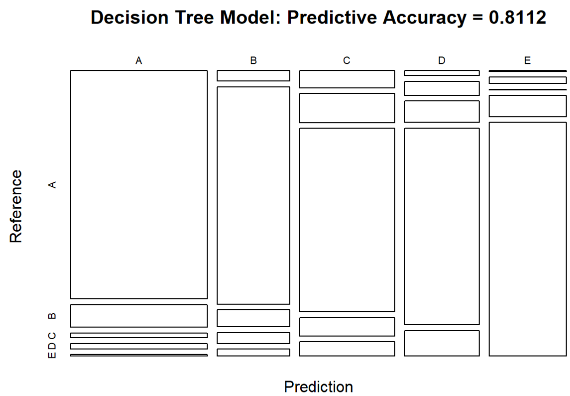


Rattle 2023-pro-23 17:19:39 chadi

```
predict_DT <- predict(fit_DT, newdata = test_set, type="class")
conf_matrix_DT <- confusionMatrix(predict_DT, factor(test_set$classe))
conf_matrix_DT
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1280  127   26   31    8
##          B   31  645   50   32   21
##          C   67  113  713   72   56
##          D   15   43   65  602   78
##          E    2   21    1   67  738
##
## Overall Statistics
##
##                Accuracy : 0.8112
##                  95% CI : (0.7999, 0.822)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7609
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9176   0.6797   0.8339   0.7488   0.8191
## Specificity            0.9453   0.9661   0.9239   0.9510   0.9773
## Pos Pred Value         0.8696   0.8280   0.6983   0.7497   0.8902
## Neg Pred Value         0.9665   0.9263   0.9634   0.9507   0.9600
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2610   0.1315   0.1454   0.1228   0.1505
## Detection Prevalence   0.3002   0.1588   0.2082   0.1637   0.1690
## Balanced Accuracy      0.9314   0.8229   0.8789   0.8499   0.8982
```

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



Predictions

DT model used to predict 20 different test cases

```
predict_test <- as.data.frame(predict(fit_DT, newdata = data_test))
predict_test
```

```
##             A          B          C          D          E
## 1  0.00000000 1.00000000 0.00000000 0.00000000 0.00000000
## 2  0.64367127 0.20635873 0.06538692 0.06778644 0.01679664
## 3  0.15735641 0.13060582 0.46498820 0.13453973 0.11250983
## 4  1.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 5  1.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 6  0.15735641 0.13060582 0.46498820 0.13453973 0.11250983
## 7  0.02249297 0.03936270 0.12464855 0.72352390 0.08997188
## 8  0.15735641 0.13060582 0.46498820 0.13453973 0.11250983
## 9  0.98976982 0.01023018 0.00000000 0.00000000 0.00000000
## 10 0.64367127 0.20635873 0.06538692 0.06778644 0.01679664
## 11 0.15735641 0.13060582 0.46498820 0.13453973 0.11250983
## 12 0.15735641 0.13060582 0.46498820 0.13453973 0.11250983
## 13 0.00000000 1.00000000 0.00000000 0.00000000 0.00000000
## 14 0.98976982 0.01023018 0.00000000 0.00000000 0.00000000
## 15 0.02249297 0.03936270 0.12464855 0.72352390 0.08997188
## 16 0.04000000 0.03076923 0.00000000 0.08615385 0.84307692
## 17 1.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 18 0.64367127 0.20635873 0.06538692 0.06778644 0.01679664
## 19 0.00000000 1.00000000 0.00000000 0.00000000 0.00000000
## 20 0.00000000 1.00000000 0.00000000 0.00000000 0.00000000
```