Practical Machine Learning

Monika Chuchro

2022-10-30

Description

Data in this project are from accelerometers on the belt,arm, forearm, and dumbell of 6 participants. Models have to predict the manner in which participants moved. Main variable is qualitative variable (5 levels) so classification models will be used. Chosen classification models: (1)decision tree, (2)random forest, (3)support vector machine and (4)generalized boosted model. Model quality will be checked using V-fold cross validation on traing dataset and with accuracy and out of sample error rate. More info: http://groupware.les.inf.puc-rio.br/har

Packages, language

```
Sys.setlocale("LC_ALL", "English")

## [1] "LC_COLLATE=English_United States.1252;LC_CTYPE=English_United
States.1252;LC_MONETARY=English_United
States.1252;LC_NUMERIC=C;LC_TIME=English_United States.1252"

library(readr)
library(caret)

## Ladowanie wymaganego pakietu: ggplot2

## Ladowanie wymaganego pakietu: lattice

library(corrplot)

## corrplot 0.92 loaded

library(rattle)

## Ladowanie wymaganego pakietu: tibble

## Ladowanie wymaganego pakietu: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

library(randomForest)

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Dolaczanie pakietu: 'randomForest'

## Nastepujacy obiekt zostal zakryty z 'package:rattle':
##
##     importance

## Nastepujacy obiekt zostal zakryty z 'package:ggplot2':
##
##     margin

library(kernlab)

##
## Dolaczanie pakietu: 'kernlab'

## Nastepujacy obiekt zostal zakryty z 'package:ggplot2':
##
##     alpha

library(gbm)

## Loaded gbm 2.1.8.1

set.seed(12345)
```

Data import, datasets

Importing data into 2 data sets: train for modeling and quality check, test for prediction.

```
train<- read_delim("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv", col_names=T,)

## New names:
## Rows: 19622 Columns: 160
## -- Column specification
## -------------------------------------------------------- Delimiter: ","
chr
## (34): user_name, cvtd_timestamp, new_window, kurtosis_roll_belt, kurtos...
dbl
## (126): ...1, raw_timestamp_part_1, raw_timestamp_part_2, num_window,
rol...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this
message.
## * `` -> `...1`

test<-read_delim("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
testing.csv", col_names=T)

## New names:
## Rows: 20 Columns: 160
## -- Column specification
```

```
## -------------------------------------------------------- Delimiter: ","
chr
## (3): user_name, cvtd_timestamp, new_window dbl (57): ...1,
## raw_timestamp_part_1, raw_timestamp_part_2, num_window, rol... lgl (100):
## kurtosis_roll_belt, kurtosis_picth_belt, kurtosis_yaw_belt, skewn...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this
message.
## * `` -> `...1`

dim(train)

## [1] 19622   160

dim(test)

## [1]  20 160
```

Preprocessing

Variables have a high number of NA, Near Zero Variance (NZV) and Id. Preprocessing will removed them. Removing NA column (mostly NA values, and columns with metadata).

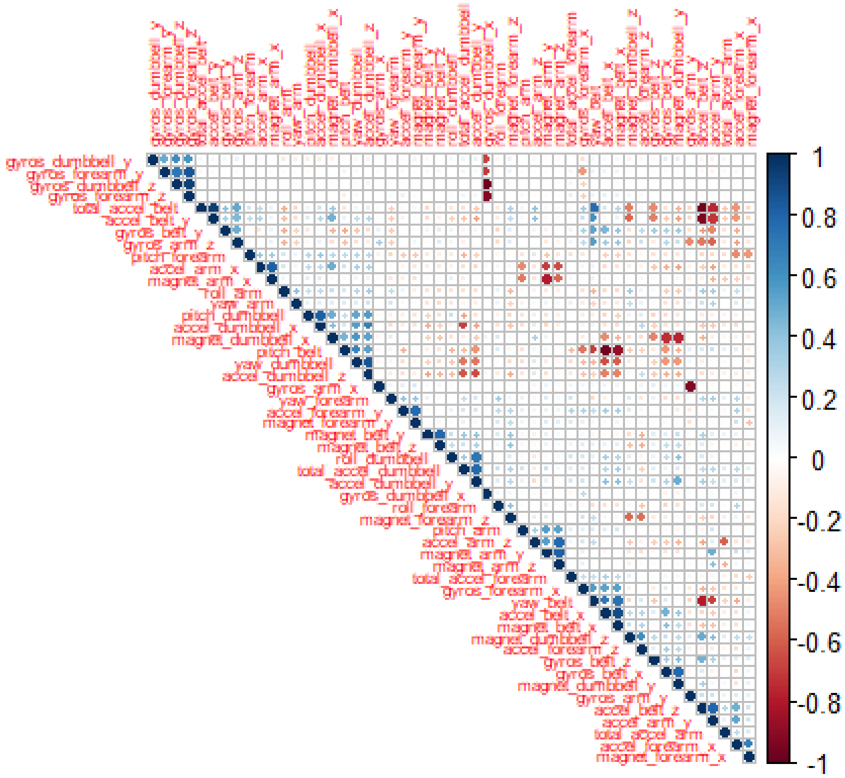
```
nvz <- nearZeroVar(train)
train <- train[,-nvz]

train <- train[,colMeans(is.na(train)) < 0.9]
train <- train[,-c(1:7)]
dim(train)

## [1] 19622    52
```

52 variables left after preprocessing.

Data analysis

Pearson correlation coefficient will present relations between pairs of variables.

```
p_cor<-round(cor(train[,-52]),2)
corrplot(p_cor, order = "hclust" , type = "upper",tl.cex = 0.5)
```




```
high_corr<-findCorrelation(p_cor, cutoff=0.75)
names(train)[high_corr]
```

```
##  [1] "accel_belt_z"      "accel_dumbbell_z"  "accel_belt_y"
##  [4] "accel_arm_y"       "total_accel_belt"  "accel_belt_x"
##  [7] "pitch_belt"        "accel_dumbbell_y"  "magnet_dumbbell_x"
## [10] "magnet_dumbbell_y" "accel_arm_x"       "accel_dumbbell_x"
## [13] "accel_arm_z"       "magnet_arm_y"      "magnet_belt_z"
## [16] "accel_forearm_y"   "gyros_forearm_y"   "gyros_dumbbell_x"
## [19] "gyros_dumbbell_z"  "gyros_arm_x"
```

The more intensive correlation color and the bigger dot is presened, the higher correlation is observed between pair of variables. The highest negative Pearson's correlation coefficient is between pitch_belt and accel_belt_x (-0.97), accel_belt_z and total_accel_belt (-0.97).

Modeling

Dividing data (train dataset) into training and validation dataset. For classification modeles quality we assess on dataset not presented in learning phase. That dataset should contain between 25% to 50% observations. In this project was used validation dataset with 30% of observations.

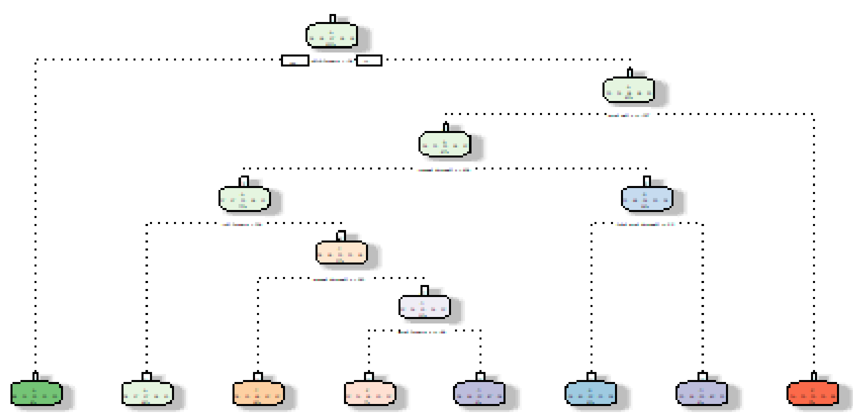
```
partition <- createDataPartition(y=train$classe, p=0.7, list=F)
training <- train[partition,]
validation <- train[-partition,]
```

Models

In models I used random seed number (12345). Models were created with V-fold validation. I used 3-fold cross validation randomly splits the data into 3 groups of roughly equal size. A resample of the analysis data consists of 2 of the folds while the assessment set contains the final fold. Models quality were checked using validation dataset with confusion matrix, accuracy and out of sample error.

##Model 1: Decision tree First model is binary decision tree created using 13737 observations.

```
set.seed(12345)
control <- trainControl(method="cv", number=3, verboseIter=FALSE)

tree1 <- train(classe~., data=training, method="rpart", trControl = control,
tuneLength = 5)
fancyRpartPlot(tree1$finalModel)
```



Rattle 2022-Nov-01 18:21:07 monik

Model quality:

```
valid_tree1 <- predict(tree1, validation)
confmat_tree1<- confusionMatrix(valid_tree1, as.factor(validation$classe))
confmat_tree1

## Confusion Matrix and Statistics
##
##             Reference
## Prediction     A    B    C    D    E
##          A 1527  482  498  423  243
```

```
##          B   31  353   37   10  176
##          C   77  124  423  126  150
##          D   19   59    7  344   70
##          E   20  121   61   61  443
##
## Overall Statistics
##
##                Accuracy : 0.5251
##                  95% CI : (0.5122, 0.5379)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3784
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9122  0.30992  0.41228  0.35685  0.40943
## Specificity            0.6091  0.94648  0.90183  0.96850  0.94524
## Pos Pred Value         0.4812  0.58155  0.47000  0.68938  0.62748
## Neg Pred Value         0.9458  0.85108  0.87904  0.88489  0.87662
## Prevalence             0.2845  0.19354  0.17434  0.16381  0.18386
## Detection Rate         0.2595  0.05998  0.07188  0.05845  0.07528
## Detection Prevalence   0.5392  0.10314  0.15293  0.08479  0.11997
## Balanced Accuracy      0.7607  0.62820  0.65706  0.66267  0.67733
```

Decision tree accuracy is quite low: 0.5251, and 95% CI: (0.5122, 0.5379). Good prediction only for Class A.

Model 2: Random Forest

The second model is Random Forest, with n=500 trees.

```
set.seed(12345)
tree2 <- train(classe~., data=training, method="rf", trControl = control,
tuneLength = 5)
```

Model quality:

```
valid_tree2<- predict(tree2, validation)
confmat_tree2<- confusionMatrix(valid_tree2, as.factor(validation$classe))
confmat_tree2
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    4    0    0    0
##          B    1 1133    3    0    0
```

```
##          C    0    2 1022    8    0
##          D    0    0    1  955    1
##          E    0    0    0    1 1081
##
## Overall Statistics
##
##                Accuracy : 0.9964
##                  95% CI : (0.9946, 0.9978)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9955
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9947   0.9961   0.9907   0.9991
## Specificity            0.9991   0.9992   0.9979   0.9996   0.9998
## Pos Pred Value         0.9976   0.9965   0.9903   0.9979   0.9991
## Neg Pred Value         0.9998   0.9987   0.9992   0.9982   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1925   0.1737   0.1623   0.1837
## Detection Prevalence   0.2850   0.1932   0.1754   0.1626   0.1839
## Balanced Accuracy      0.9992   0.9969   0.9970   0.9951   0.9994
```

Random Forest has very high accuracy : 0.9961 and 95% CI : (0.9941, 0.9975). Like the first model the best prediction results were obtaoned for Class A.

Model 3: Support Vector Machine

```
set.seed(12345)
svm1<-train(classe~., data=training, method="svmLinear", trControl = control,
tuneLength = 5, verbose = F)
```

Model quality:

```
valid_svm1<- predict(svm1, validation)
confmat_svm1<- confusionMatrix(valid_svm1, factor(validation$classe))
confmat_svm1

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1556  160   97   70   76
##          B   32  808  114   47  147
##          C   39   65  761  114   60
##          D   38   21   37  691   75
##          E    9   85   17   42  724
##
```

```
## Overall Statistics
##
##                Accuracy : 0.7715
##                  95% CI : (0.7605, 0.7821)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.709
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9295   0.7094   0.7417   0.7168   0.6691
## Specificity            0.9043   0.9284   0.9428   0.9653   0.9681
## Pos Pred Value         0.7943   0.7038   0.7324   0.8016   0.8255
## Neg Pred Value         0.9699   0.9301   0.9453   0.9457   0.9285
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2644   0.1373   0.1293   0.1174   0.1230
## Detection Prevalence   0.3329   0.1951   0.1766   0.1465   0.1490
## Balanced Accuracy      0.9169   0.8189   0.8423   0.8410   0.8186
```

SVM model has better result than 1 model. Accuracy is 0.7715 and 95% CI : (0.7605, 0.7821). We obtain very good result in prediction Class A: 0.9295.

Model 4:Generalized Boosted Model

GBM A gradient boosted model with multinomial loss function with 150 iterations. There were 51 predictors of which 51 had non-zero influence

```
set.seed(12345)
gbm1<- train(classe ~ ., data = training, method = "gbm",
                  trControl = control, verbose = FALSE)
gbm1$finalModel

## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 51 predictors of which 51 had non-zero influence.
```

Model quality:

```
valid_gbm1 <- predict(gbm1, newdata = validation)
confmat_gbm1<- confusionMatrix(valid_gbm1, factor(validation$classe))
confmat_gbm1

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1641   46    0    0    1
```

```
##          B   24 1054   36    6   14
##          C    7   34  977   32    4
##          D    2    1   13  918   12
##          E    0    4    0    8 1051
##
## Overall Statistics
##
##                Accuracy : 0.9585
##                  95% CI : (0.9531, 0.9635)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9475
##
##  Mcnemar's Test P-Value : 2.621e-05
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9803   0.9254   0.9522   0.9523   0.9713
## Specificity            0.9888   0.9831   0.9842   0.9943   0.9975
## Pos Pred Value         0.9722   0.9295   0.9269   0.9704   0.9887
## Neg Pred Value         0.9921   0.9821   0.9899   0.9907   0.9936
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2788   0.1791   0.1660   0.1560   0.1786
## Detection Prevalence   0.2868   0.1927   0.1791   0.1607   0.1806
## Balanced Accuracy      0.9846   0.9543   0.9682   0.9733   0.9844
```

GBM: quality in validation dateset is very high: Accuracy is 0.9585 and 95% CI : (0.9531, 0.9635).

Quality in validation datasets (ACCURACY and OUT-OF_SAMPLE ERROR):

Decision trees: 0.5251 Random Forest: 0.9961 - 1st place Support Vector Machine: 0.7715 Generalized Boosted Model: 0.9585

The expected out-of-sample error correspond to the quantity: 1-accuracy in the cross-validation data. Expected value of the out-of-sample error correspond to the expected number of missclassified observations/total observations in the validation dataset. Decision trees: ~0.48 Random Forest: ~0.004 Support Vector Machine: ~0.23 Generalized Boosted Model: ~0.04

There is posibility that Random Forest model is overfitted.

For validation dataset the best results were obtained with Random Forest.

Testing Random Forest model on test dataset (20 observations)

```
pred_tree2<-predict(tree2, test)
pred_tree2
```

```
##  [1] 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
```

```
table(pred_tree2)
```

```
## pred_tree2
## A B C D E
## 7 8 1 1 3
```