Machine Learning Project

Martin Rico

2023-07-06

1 step : Loading Data

Loading data from the next links:

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

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

```
urlTraining <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
training<-read.csv(urlTraining)

urlTesting <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
testing<-read.csv(urlTesting)
```

2 step: Selecting predictors variables

Select variables reported by the accelerometer and the variables that bring information about the angles. Create a new dataset with that information, PredictorDataSet.

```
accelerometer_var<-grep("\\_x$|\\_y$|\\_z$",names(training))
angles_var<-grep("^roll|^pitch|^yaw",names(training))
predictorDataSet<-data.frame(training[,accelerometer_var], training[,angles_var], training$classe)
```

Now, selecting variables for testing

```
accelerometer_var_testing<-grep("\\_x$|\\_y$|\\_z$",names(testing))
angles_var_testing<-grep("^roll|^pitch|^yaw",names(testing))
testingDataSet<-data.frame(testing[,accelerometer_var], testing[,angles_var])
```

3 step: Fitting a model

The problem could be handle as Classification problem, because the goal is identify if the records belong to A,B,C,D or E.

The Tree algorithm will be used to build the first model

The Random Forest will be used to build the second model

```
unique(predictorDataSet$training.classe)
```

```
## [1] "A" "B" "C" "D" "E"
```

The validation Cross technique to use is K-Fold with k value set to 10 The trainControl function will be used to set the cross validation type The train function will be used to fit the model.

Model build using Tree algorithm

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.2.3
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
## Loading required package: lattice
```

```
set.seed(125)
cvType<-trainControl(method = "cv", number = 10)
model<-train(training.classe ~ ., data = predictorDataSet, method="rpart", trControl = cvType)
print(model)
```

```
## CART
##
## 19622 samples
##    48 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 17659, 17659, 17659, 17661, 17661, 17660, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.03567868  0.5037212  0.35174785
##   0.05998671  0.4158527  0.20837322
##   0.11515454  0.3326296  0.07350106
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.03567868.
```

Model build with random forest algorithm

```
modelRandomForest<-train(training.classe ~ ., data = predictorDataSet, method="rf", trControl = cvType)
print(modelRandomForest)
```

```
## Random Forest
##
## 19622 samples
##    48 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
```

```
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 17661, 17660, 17659, 17659, 17660, 17659, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa
##    2    0.995770  0.9946492
##   25    0.994649  0.9932313
##   48    0.989247  0.9863967
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

4 step: predict using model

Predict using the best model builded using Random Forest algorithm

```
prediction<-predict(modelRandomForest,testingDataSet)
print(prediction)
```

```
##  [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
```

Summary

The best approach in order to identify if a exercise is do it correctly is using random forest because its accuracy is 0.995. It was determinate using cross validation on the original training data set.

The trainControl function split the data using k=10 avoiding do this task manually.