Activities of Daily Living (ADLs)

Tilen Rupar

23 6 2021

```
library(tidyverse)
library(readr)
library(lubridate)
library(caret)
library(ggplot2)
library(class)
```

Project goals

I will try to predict human activity based on sensory data with machine learning algorithm.

The machine learning algorithm is a supervised time series classification model. It uses POSIXct class (timestamp) to classify human activity, based on the time of the day.

Data: UCI Machine learning repository: ADL data

```
##  [1] "=============================="
##  [2] "OrdonezA DATASET DESCRIPTION"
##  [3] "=============================="
##  [4] "Home setting: 4 rooms house"
##  [5] "Number of labelled days: 14 days"
##  [6] "Labels (ADLs included):  Leaving, Toileting, Showering, Sleeping, Breakfast, Lunch, Dinner, Sna
##  [7] "Number of sensors: 12 sensors"
##  [8] "Sensors: "
##  [9] "PIR: Shower, Basin, Cooktop"
## [10] "Magnetic: Maindoor, Fridge, Cabinet, Cupboard"
## [11] "Flush: Toilet"
## [12] "Pressure: Seat, Bed"
## [13] "Electric: Microwave, Toaster"
## attr(,"na.action")
##  [1]  6  9 12 13 14 15 17 18 19 20 22 23 24 25 27 28 29 30
## attr(,"class")
## [1] "omit"
```

```
#ADL.uci_file <- download.file('https://archive.ics.uci.edu/ml/machine-learning-databases/00271/UCI ADL
#unzip("./ADL.zip")

# Activities of Daily Living (ADLs) Recognition Using Binary Sensors Data Set (User "A")
ADL <- read.table("OrdonezA_ADLs.txt", skip = 2)
colnames(ADL) <- read.table("OrdonezA_ADLs.txt", nrows = 1)
```

```r
#Creating a date time object for merging with the sensor dataset
ADL$Start_Date_time <- as.POSIXct(paste(ADL$Start, ADL$time), format="%Y-%m-%d %H:%M:%S")
ADL <- subset(ADL, select = c(Activity, Start_Date_time))

#Reading the sensor dataset
ADL_sensor <- read.table("OrdonezA_Sensors.txt", skip = 2)
colnames(ADL_sensor) <- read.table("OrdonezA_Sensors.txt", nrows = 1)
ADL_sensor$Start_Date_time<- as.POSIXct(paste(ADL_sensor$Start, ADL_sensor$time),
                                        format="%Y-%m-%d %H:%M:%S")
ADL_sensor <- subset(ADL_sensor, select = c(Start_Date_time, Location, Type, Place))
# Merging the Activity and sensor data
ADL <- merge(x = ADL, y = ADL_sensor, by = intersect(names(ADL), names(ADL_sensor)))
#add a day name,
ADL$Day <- as.factor(weekdays(as.Date(ADL$Start)))
# a Dummy for weekend
ADL$isWKND <- as.double(ADL$Day %in% c("Saturday", "Sunday"))

# and Time_of_day labels
# create breaks
breaks <- hour(hm("00:00", "6:00", "12:00", "18:00", "23:59"))
# labels for the breaks
labels <- c("Night", "Morning", "Afternoon", "Evening")
ADL$Time_of_day <- cut(x=hour(ADL$Start_Date_time), breaks = breaks, labels = labels, include.lowest=TRU
```

Imported data:

```r
head(ADL)
```

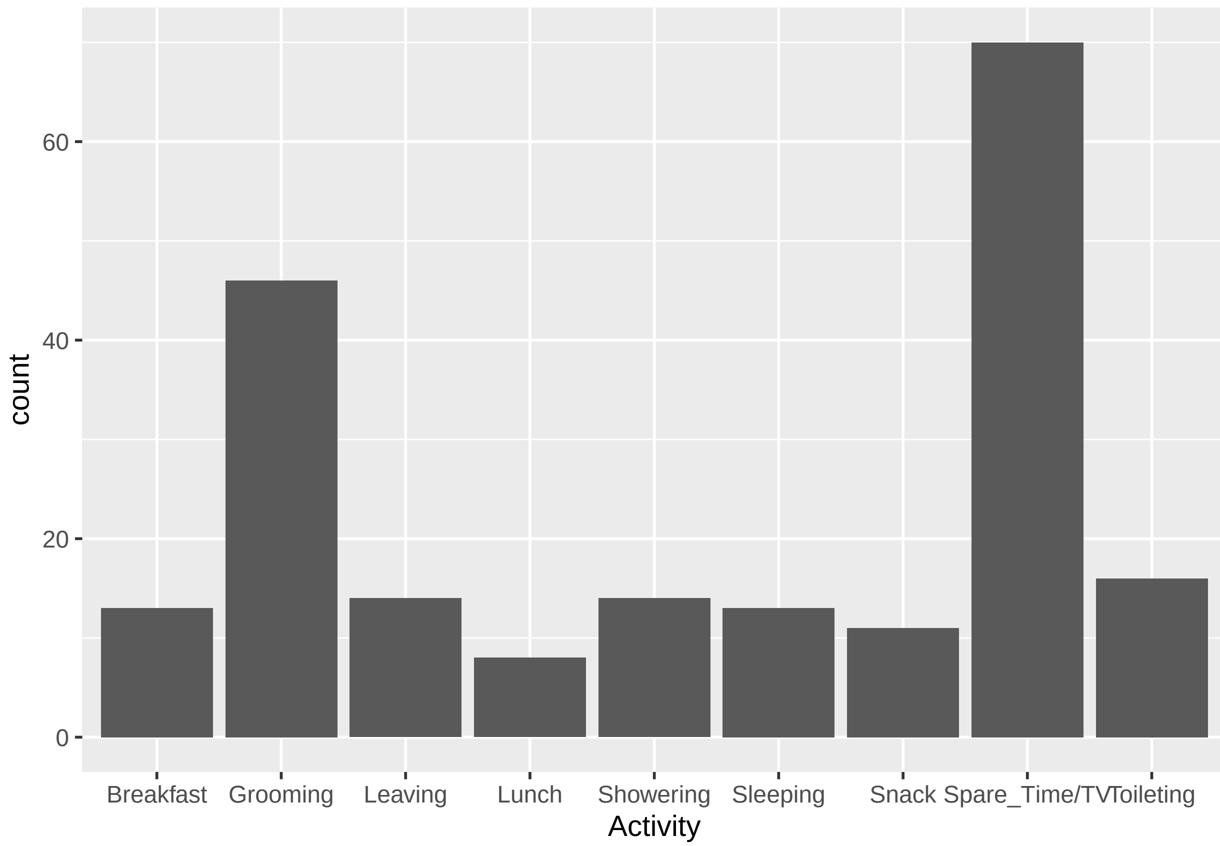
```
##       Start_Date_time      Activity Location     Type    Place    Day isWKND
## 1 2011-11-28 02:27:59      Sleeping      Bed Pressure  Bedroom Monday      0
## 2 2011-11-28 10:21:24     Toileting  Cabinet Magnetic Bathroom Monday      0
## 3 2011-11-28 10:25:44     Showering   Shower      PIR Bathroom Monday      0
## 4 2011-11-28 10:34:23     Breakfast   Fridge Magnetic  Kitchen Monday      0
## 5 2011-11-28 10:49:48      Grooming    Basin      PIR Bathroom Monday      0
## 6 2011-11-28 10:51:41 Spare_Time/TV     Seat Pressure   Living Monday      0
##   Time_of_day
## 1       Night
## 2     Morning
## 3     Morning
## 4     Morning
## 5     Morning
## 6     Morning
```

We are interested in predicting Activity, based on Start and End time. Thus we turn the character vector “Activity” in factor with 9 levels:

```
## [1] "Sleeping"      "Toileting"     "Showering"     "Breakfast"
## [5] "Grooming"      "Spare_Time/TV" "Leaving"       "Lunch"
## [9] "Snack"
```

We visualize the Activity data:

```
ggplot(ADL, aes(x=Activity)) +
  geom_bar()
```



For the modelling purposes we need to convert the variables into numeric objects.

```
ADL_modData <- data.frame(Activity = as.factor(ADL$Activity),
                          Location = as.numeric(as.factor(ADL$Location)),
                          Type = as.numeric(as.factor(ADL$Type)),
                          Place = as.numeric(as.factor(ADL$Place)),
                          Day = as.numeric(ADL$Day),
                          Time_of_day = as.numeric(ADL$Time_of_day),
                          isWKND = ADL$isWKND)
str(ADL_modData)
```

```
## 'data.frame':    205 obs. of  7 variables:
##  $ Activity   : Factor w/ 9 levels "Breakfast","Grooming",..: 6 9 5 1 2 8 9 3 8 4 ...
##  $ Location   : num  2 3 9 6 1 8 1 7 8 6 ...
##  $ Type       : num  5 3 4 3 4 5 4 3 5 3 ...
##  $ Place      : num  2 1 1 4 1 5 1 3 5 4 ...
##  $ Day        : num  2 2 2 2 2 2 2 2 2 2 ...
##  $ Time_of_day: num  1 2 2 2 2 2 3 3 3 3 ...
##  $ isWKND     : num  0 0 0 0 0 0 0 0 0 0 ...
```

Data partitioning (using caret package) - 0.8 data goes into training

```
set.seed(100)
sample_rows <- as.vector(createDataPartition(ADL_modData$Activity, p=0.8, list = F))
ADL_train <- ADL_modData[sample_rows,]
ADL_test <- ADL_modData[-sample_rows,]
```

Applying the **k-Nearest neighbours model** to the ADL data

```
ADL_kNN <- knn(ADL_train[,-1], ADL_test[,-1], cl=ADL_train$Activity, k = 7)
```

Creating the confusion matrix for the kNN model (Accuracy check)

```
table(ADL_test$Activity, ADL_kNN)
```

```
##                  ADL_kNN
##                   Breakfast Grooming Leaving Lunch Showering Sleeping Snack
##    Breakfast              1        0       1     0         0        0     0
##    Grooming               0        9       0     0         0        0     0
##    Leaving                0        0       0     0         0        0     2
##    Lunch                  0        0       0     1         0        0     0
##    Showering              0        0       0     0         2        0     0
##    Sleeping               0        0       0     0         0        2     0
##    Snack                  0        0       1     0         0        0     1
##    Spare_Time/TV          0        0       0     0         0        0     0
##    Toileting              0        1       0     0         0        0     0
##                  ADL_kNN
##                   Spare_Time/TV Toileting
##    Breakfast                  0         0
##    Grooming                   0         0
##    Leaving                    0         0
##    Lunch                      0         0
##    Showering                  0         0
##    Sleeping                   0         0
##    Snack                      0         0
##    Spare_Time/TV             14         0
##    Toileting                  0         2
```

Next, we check the accuracy rate of the kNN model:

```
mean(ADL_test$Activity == ADL_kNN)
```

```
## [1] 0.8648649
```

The kNN model predicted with **86,4% accuracy rate**.

Next, we try to fit the **random forest model** to the data.

```r
#Before applying machine learning algorithms to train our model, let us first tune the cross-validation
#We expect our out-of-sample error to be low because 5-fold CV should take its effect and avoid overfit
fitControl <- trainControl(method = "cv", number = 5, returnResamp = "all")

#Random forest model
ADL_rf <- train(Activity ~ Time_of_day + isWKND + Day + Location + Type + Place, method="rf", data=ADL_
```

Creating a prediction based on RF training data and analyzing the confusion matrix

```r
predictedADL_rf <- predict(ADL_rf, ADL_test)

confusionMatrix(ADL_test$Activity, predictedADL_rf)
```

```
## Confusion Matrix and Statistics
##
##                  Reference
## Prediction       Breakfast Grooming Leaving Lunch Showering Sleeping Snack
##   Breakfast              2        0       0     0         0        0     0
##   Grooming               0        9       0     0         0        0     0
##   Leaving                0        0       2     0         0        0     0
##   Lunch                  0        0       0     1         0        0     0
##   Showering              0        0       0     0         2        0     0
##   Sleeping               0        0       0     0         0        2     0
##   Snack                  0        0       0     1         0        0     1
##   Spare_Time/TV          0        0       0     0         0        0     0
##   Toileting              0        1       0     0         0        0     0
##                  Reference
## Prediction       Spare_Time/TV Toileting
##   Breakfast                  0         0
##   Grooming                   0         0
##   Leaving                    0         0
##   Lunch                      0         0
##   Showering                  0         0
##   Sleeping                   0         0
##   Snack                      0         0
##   Spare_Time/TV             14         0
##   Toileting                  0         2
##
## Overall Statistics
##
##                Accuracy : 0.9459
##                  95% CI : (0.8181, 0.9934)
##     No Information Rate : 0.3784
##     P-Value [Acc > NIR] : 4.494e-13
##
##                   Kappa : 0.93
##
##  Mcnemar's Test P-Value : NA
##
```

```
## Statistics by Class:
##
##                      Class: Breakfast Class: Grooming Class: Leaving
## Sensitivity                   1.00000          0.9000        1.00000
## Specificity                   1.00000          1.0000        1.00000
## Pos Pred Value                1.00000          1.0000        1.00000
## Neg Pred Value                1.00000          0.9643        1.00000
## Prevalence                    0.05405          0.2703        0.05405
## Detection Rate                0.05405          0.2432        0.05405
## Detection Prevalence          0.05405          0.2432        0.05405
## Balanced Accuracy             1.00000          0.9500        1.00000
##                      Class: Lunch Class: Showering Class: Sleeping Class: Snack
## Sensitivity               0.50000          1.00000         1.00000      1.00000
## Specificity               1.00000          1.00000         1.00000      0.97222
## Pos Pred Value            1.00000          1.00000         1.00000      0.50000
## Neg Pred Value            0.97222          1.00000         1.00000      1.00000
## Prevalence                0.05405          0.05405         0.05405      0.02703
## Detection Rate            0.02703          0.05405         0.05405      0.02703
## Detection Prevalence      0.02703          0.05405         0.05405      0.05405
## Balanced Accuracy         0.75000          1.00000         1.00000      0.98611
##                      Class: Spare_Time/TV Class: Toileting
## Sensitivity                        1.0000          1.00000
## Specificity                        1.0000          0.97143
## Pos Pred Value                     1.0000          0.66667
## Neg Pred Value                     1.0000          1.00000
## Prevalence                         0.3784          0.05405
## Detection Rate                     0.3784          0.05405
## Detection Prevalence               0.3784          0.08108
## Balanced Accuracy                  1.0000          0.98571
```

AS we can see, the random forest model did not capture the data successfully, the accuracy is 94,59%.

Presentation of the predicted vs. actual Activity

```
data.frame(Actual_data = ADL_test$Activity, Random_Forest_prediction = predictedADL_rf, kNN_prediction =
```

```
##      Actual_data Random_Forest_prediction kNN_prediction
## 1      Toileting                 Grooming       Grooming
## 2      Showering                Showering      Showering
## 3      Breakfast                Breakfast        Leaving
## 4          Snack                    Snack        Leaving
## 5       Grooming                 Grooming       Grooming
## 6  Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 7      Toileting                Toileting      Toileting
## 8        Leaving                  Leaving          Snack
## 9       Grooming                 Grooming       Grooming
## 10     Toileting                Toileting      Toileting
## 11 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 12         Lunch                    Lunch          Lunch
## 13 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 14      Grooming                 Grooming       Grooming
```

```
## 15      Sleeping                     Sleeping      Sleeping
## 16 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 17 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 18     Breakfast                Breakfast     Breakfast
## 19      Grooming                 Grooming      Grooming
## 20      Sleeping                 Sleeping      Sleeping
## 21     Showering                Showering     Showering
## 22 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 23 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 24      Grooming                 Grooming      Grooming
## 25 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 26 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 27      Grooming                 Grooming      Grooming
## 28 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 29      Grooming                 Grooming      Grooming
## 30      Grooming                 Grooming      Grooming
## 31       Leaving                  Leaving         Snack
## 32 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 33         Snack                    Lunch         Snack
## 34 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 35 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
## 36      Grooming                 Grooming      Grooming
## 37 Spare_Time/TV            Spare_Time/TV  Spare_Time/TV
```