Human Activity Recognition (HAR)

Jose Ramon Hernandez Galan

18/11/2019

Introduction

This is a report showing the process and results for the creation of a model for human activity recognition (herafter HAR).

This is the capstone project of the Data Science course pursued by the author in HarvardX.

Dataset

The dataset used in this project is the **HAR Dataset for benchmarking** [1]

The dataset includes measurements of **inertial sensors** attached to several person while doing normal activities during the day. It also includes data related to the person such as weight, height, etc.

A more detailed description of the dataset can be found **here**.

The main goal of this project is to use machine learning techniques in order to predict the human activity. We will compare our results to those obtained by the main **contributor** [1].

We will also observe if all 4 sensors are really necessary or if we can use less sensors in order to predict the activity.

Analysis

In this section we will prepare the data to work with and explore some important characteristics of the dataset.

Data wrangling

The created har dataset has the following structure:

Name	Type	Description
user	Factor	w/ 4 levels "debora","jose_carlos",..: 1 1 1 1 1 1 1 1 1 1 ...
gender	Factor	w/ 2 levels "Man","Woman": 2 2 2 2 2 2 2 2 2 2 ...
age	int	46 46 46 46 46 46 46 46 46 46 ...
how_tall_in_meters	num	1.62 1.62 1.62 1.62 1.62 1.62 1.62 1.62 1.62 1.62 ...
weight	int	75 75 75 75 75 75 75 75 75 75 ...
body_mass_index	num	28.6 28.6 28.6 28.6 28.6 28.6 28.6 28.6 28.6 28.6 ...
x1	int	-3 -3 -1 -2 -1 -2 1 -1 -1 0 ...
y1	int	92 94 97 96 96 95 100 97 98 98 ...
z1	int	-63 -64 -61 -57 -61 -62 -62 -63 -63 -61 ...
x2	int	-23 -21 -12 -15 -13 -14 -10 -13 -14 -11 ...
y2	int	18 18 20 21 20 19 22 20 19 22 ...
z2	int	-19 -18 -15 -16 -15 -16 -12 -15 -17 -13 ...
x3	int	5 -14 -13 -13 -13 -13 -13 -12 -13 -13 ...
y3	int	104 104 104 104 104 104 104 104 104 104 ...

Name	Type	Description
z3	int	-92 -90 -90 -89 -89 -89 -90 -88 -90 -90 ...
x4	int	-150 -149 -151 -153 -153 -153 -151 -151 -152 -151 ...
y4	int	-103 -104 -104 -103 -104 -104 -104 -104 -103 -104 ...
z4	int	49 47 45 43 44 43 44 43 45 45 ...
class	Factor	w/ 5 levels “sitting”,“sittingdown”,..: 1 1 1 1 1 1 1 1 1 1 ...

See the different activites listed in the dataset:

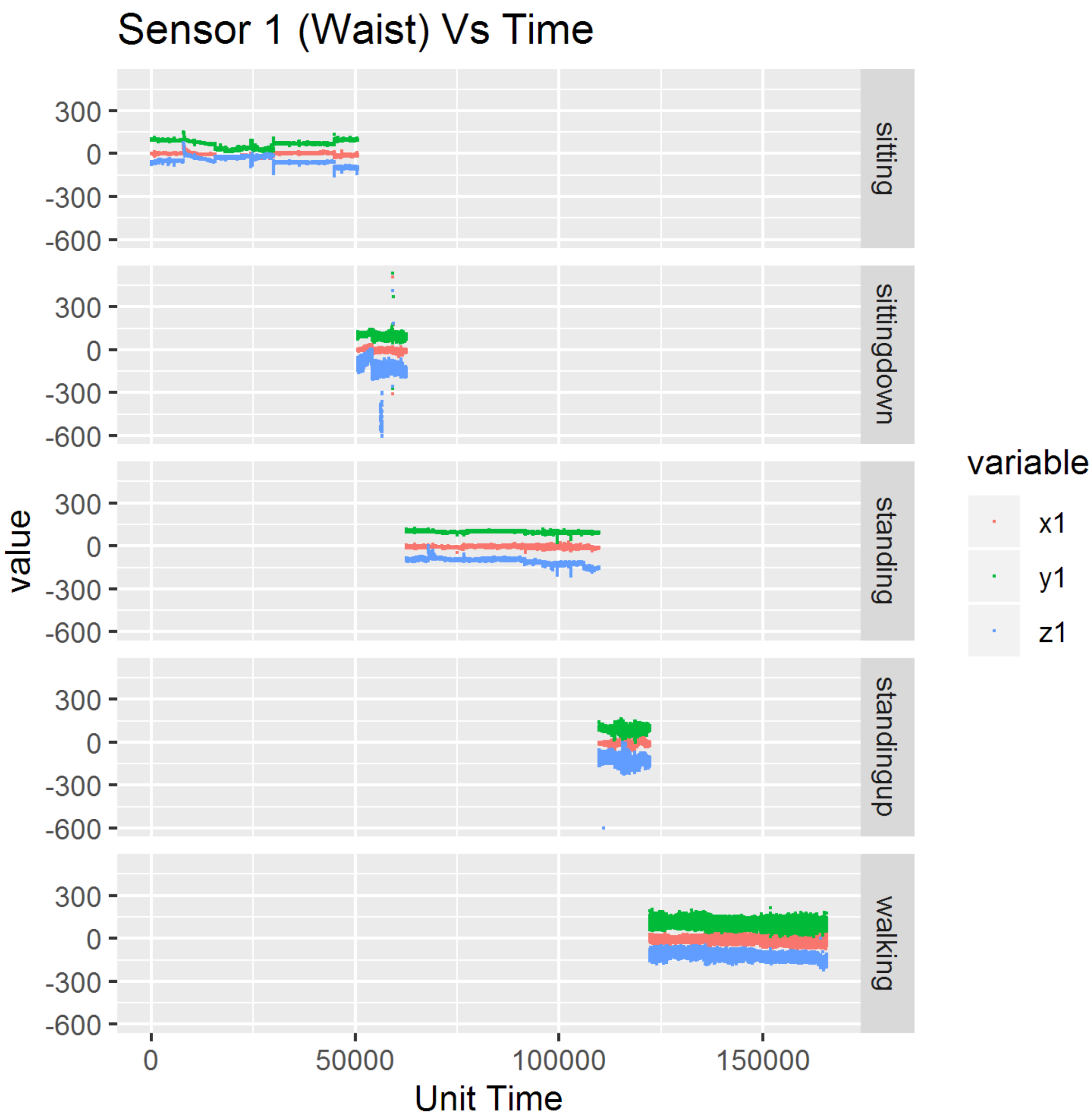
```
levels(har$class)
```

```
## [1] "sitting"     "sittingdown" "standing"    "standingup"  "walking"
```

These classes are associated to the values obtained by the sensors (1 to 4) expressed in m/s2. X, Y, Z are the values obtained for each axis.

Exploratory Data Analysis

In order to understand how th values were taken we will view the sampled data **across the time for sensor 1 values (x,y,z)**. The data was taken during 8 hours. During this time the sensed subject was doing different activites.



This is the sensor located at the waist of the subjects.

Note the **unit time** is not specified. We have included a non-scaled time unit for creating the chart.

Some additional **checkings** have been done in order to verify the consistency of data.

```
# Is there any NAs ?
nas <- apply(har, MARGIN = 2, function(x) any(is.na(x) | is.infinite(x)))
if (any(nas) == FALSE)
{
  cat("Great, no NAs or Infinite value in the dataset")
}else{
  cat("Attention, some NA or Infinite value was found in the dataset")
}
```

```
## Great, no NAs or Infinite value in the dataset
```

```r
# proportion of classes
har %>% group_by(class) %>% summarize(n = n(), p = n()/nrow(.)) %>% arrange(desc(p))
```

```
## # A tibble: 5 x 3
##   class           n        p
##   <fct>       <int>    <dbl>
## 1 sitting     50631 0.30568
## 2 standing    47370 0.28599
## 3 walking     43390 0.26196
## 4 standingup  12415 0.074955
## 5 sittingdown 11827 0.071405
```

```r
# check proportion of classes and users
har %>% group_by(class, user) %>% summarize(n = n(), p = n()/nrow(.)) %>% arrange(desc(p))
```

```
## # A tibble: 20 x 4
## # Groups:   class [5]
##    class       user            n         p
##    <fct>       <fct>       <int>     <dbl>
##  1 sitting     debora      15615 0.094275
##  2 sitting     wallace     14993 0.090519
##  3 standing    debora      14940 0.090199
##  4 standing    wallace     14467 0.087344
##  5 sitting     katia       14280 0.086215
##  6 standing    katia       14234 0.085937
##  7 walking     wallace     14037 0.084748
##  8 walking     debora      13622 0.082242
##  9 walking     katia       13556 0.081844
## 10 sitting     jose_carlos  5743 0.034673
## 11 standingup  wallace      4115 0.024844
## 12 sittingdown katia        4017 0.024252
## 13 standingup  debora       3853 0.023262
## 14 standing    jose_carlos  3729 0.022514
## 15 standingup  katia        3710 0.022399
## 16 sittingdown debora       3547 0.021415
## 17 sittingdown wallace      3486 0.021047
## 18 walking     jose_carlos  2175 0.013131
## 19 sittingdown jose_carlos   777 0.0046911
## 20 standingup  jose_carlos   737 0.0044496
```

Now, we are sure there are **not NA values** in our dataset. Also we are sure **all users (4)** have been measured in **all the different activities (5)**. Note the prevalance in some user/activity. This also can be observed in the time-based chart above.

Data set partitioning

We have splitted the har original dataset in several sets for training, tuning and validatiaon.

Dataset	Observations	Proportions	Description
har	165.633	1	Original
har_val	16.565	1/10 of har	Set for final validation

Dataset	Observations	Proportions	Description
har_set	149.068	9/10 of har	Set for model analysis
har_set_train	134.158	9/10 of har_set	Set for model training
har_set_test	16.565	1/10 of har_set	Set for model testing while optiomizing

Analysis approach

Since the outcome of the dataset is **multi-categorical** variable (class: sitting, sittingdown, standing, standingup, walking) with 5 possible values we have been focused on **classifcation algorithms** that supports such variables. The algorithm chosen is the **decisssion trees** and its derivatives.

We will only focus on **features** provided by the **sensors** (x1..z4).

How results are measured

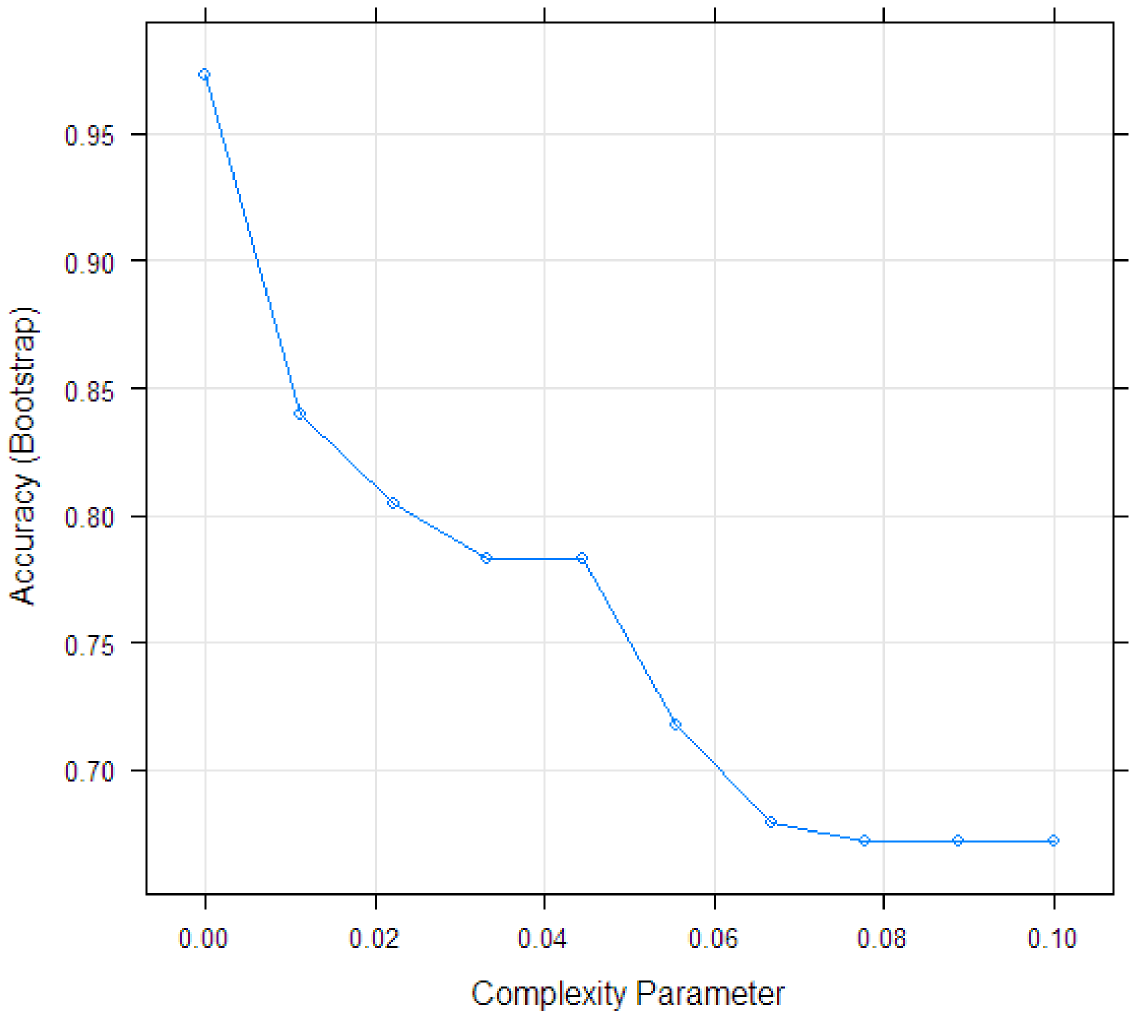
The results are measured by the **accuracy** when classsifying the sensor observations.

As final resut we will provide the confusion matrix so it can be compared to hte results in [1].

Model based on Classification Trees

Decission Trees

```
fit_part <- train(class ~ x1 + y1 + z1 + x2 + y2 + z2 + x3 + y3 + z3 + x4 + y4 + z4 ,
                  method = "rpart",
                  tuneGrid = data.frame(cp = seq(0.0, 0.1, len = 10)),
                  data = har_set_train)
plot(fit_part)
```

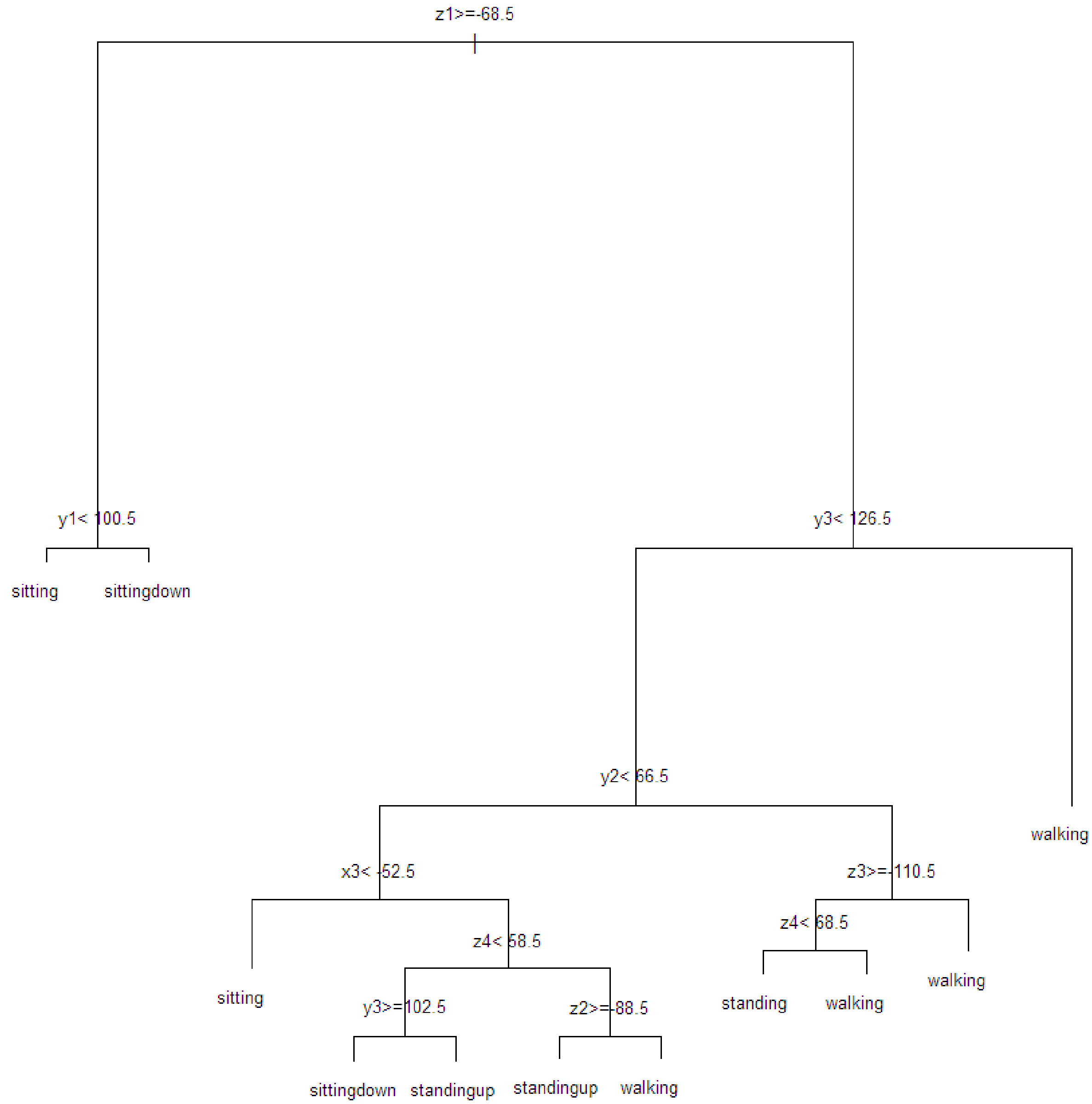


Since the created tree (for max accuracy) is pretty complex and dense (with 690 splits) we will ommit it.

When predicting over the har_set_test we obtain following accuracy:

```
## # A tibble: 1 x 3
##   METHOD                     TUNING               ACCURACY
##   <chr>                      <chr>                   <dbl>
## 1 Classification tree (rpart) CP = 0 (690 splits)  0.97632
```

In order to view a real example of a tree we will create a model with few branches by **pruning** the original tree.



Note accuracy is lower with a higher Complexity Parameter (CP) as observed in the chart above. A higher CP means that lesser branches will be used to compose the tree.

```
## # A tibble: 2 x 3
##   METHOD                              TUNING                 ACCURACY
##   <chr>                               <chr>                     <dbl>
## 1 Classification tree (rpart)         CP = 0 (690 splits)     0.97632
## 2 Classification tree (rpart) pruned  CP = 0.01 (10 splits)   0.84997
```

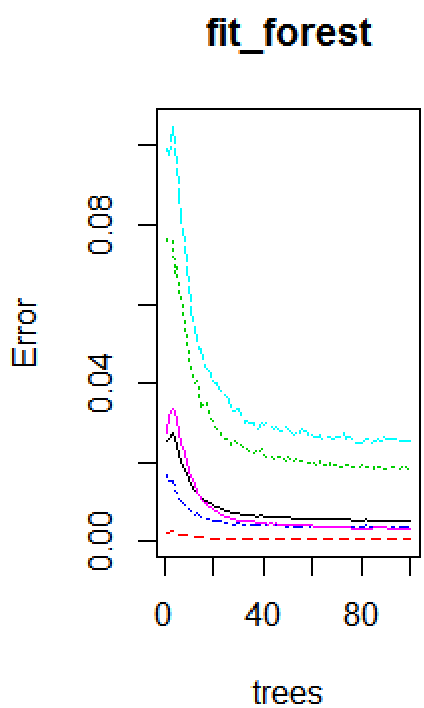
Random Forest

A more optimized algorithm based on classification trees is the random forest. This algorithm creates several trees with randomly selected features and the outcome is calculated by voting for the most probable outcome across all trees.

```r
set.seed(14)
mtry <- seq(1,12,1) # number of variables randomly selected for each tree

fit_forest <- randomForest(class ~ x1 + y1 + z1 + x2 + y2 + z2 + x3 + y3 + z3 + x4 + y4 + z4,
                          data = har_set_train,
                          ntree = 100,
                          do.trace = 10,
                          tuneGrid = data.frame(mtry = mtry))

plot(fit_forest)
```



We choose 50 trees as number of trees to grow and 3 as the number of features to be randomly selected for each tree (parameter called mtry).

```r
fit_forest1$mtry
```

```
## [1] 3
```

Some train control is needed so as to speed up the run time of the model training. As we can see the accuracy **improves in 2%** with respect to single tree algorithm.

```
## # A tibble: 3 x 3
##   METHOD                              TUNING                ACCURACY
##   <chr>                               <chr>                    <dbl>
## 1 Classification tree (rpart)         CP = 0 (690 splits)    0.97632
## 2 Classification tree (rpart) pruned  CP = 0.01 (10 splits)  0.84997
## 3 Classification tree (Random Forest) Trees = 50, mtry = 3   0.99497
```

Final Results

Find below the **confusion matrix** obtained from the prediction on the **har_val** set using the randm forest algorithm. Note this set is composed by 16.565 observations.

```
##              Reference
## Prediction    sitting sittingdown standing standingup walking
##   sitting        5059           1        0          0       0
##   sittingdown       2        1165        0         10       4
##   standing          0           2     4720         10       6
##   standingup        3           5        3       1204       4
##   walking           0          10       14         18    4325
```

We can observe how the accuracy ir lower on those classes with lesser observations.

The **accuracy** results obatined on the validation set (har_val) across the different algorithms are shown below:

METHOD	TUNING	ACCURACY
1 Classification tree (rpart)	CP = 0 (690 splits)	0.97609
2 Classification tree (rpart)	CP = 0.01 (10 splits)	0.80730
3 Classification tree (Random Forest)	Trees = 50, mtry = 3	0.99445

Conclusions

- Modeling approach. As the outcome is a multi-categorical variable we have decided to use **classification algorithms**, which have provided pretty **high accurate resutls**.
- **Results** compared to the author. The accuracy reported by the main contributor [1] is 0.99414, which was predicted on the full har dataset. Our prediction, which was created on a subset of the har dataset (without overtraining), provided an accuracy of **0.99445**.
- **Future work**. The features related to the user are not included in the described models. The **user effect** is something to be analysed. The testbench was used on the data extracted from the same subjects. But what would happen if we test our model on values from different users ? what about different **sensors**?

References

1. Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6

Read more: http://groupware.les.inf.puc-rio.br/har#sbia_paper_section#ixzz65cgnrXLU