
Coisas estranhas que podemos fazer com dados coletados em dispositivos móveis: bye-bye privacidade!

Human Activity Recognition

Fabrício J. Barth
fabricio.barth@gmail.com

Primeira versão: Novembro de 2014

Atualização: Março de 2021

Exemplos de dispositivos e dados que podem ser coletados

Celulares



Coleta informações sobre:

- Localização (latitude e longitude);
- Movimentação (acelerômetro, giroscópio);
- Ambiente (audio, proximidade, luminosidade);
- Social (histórico de ligações, contatos).

Relógios



Coleta informações sobre:

- **Frequência cardíaca;**
- Localização (latitude e longitude);
- Movimentação (acelerômetro, giroscópio);
- Ambiente (audio, proximidade, luminosidade);

Monitora atividades físicas



Monitora:

- Que atividade está sendo realizada;
- Qual a duração;
- Qual a frequência.
- Qual a intensidade do exercício.

Disney



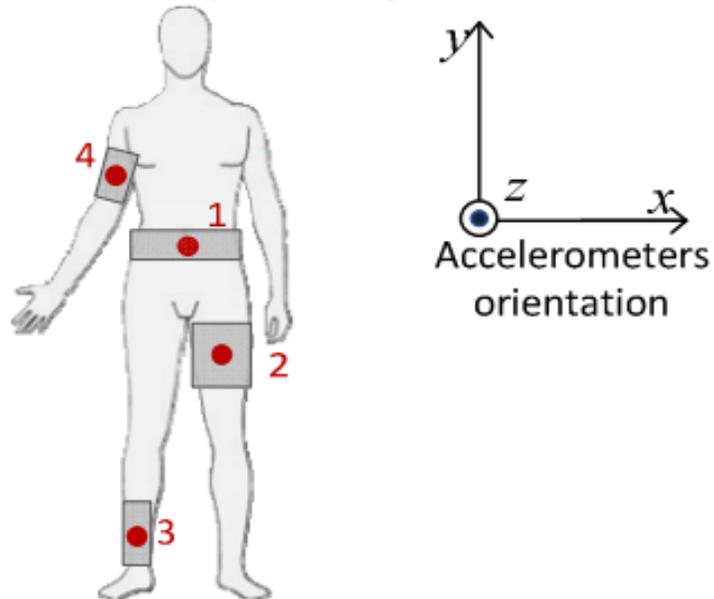
Monitora:

- Quando entrou e saiu do parque;
- Por onde andou;
- O que e quando comprou;
- Em quais parques foi e quando foi.

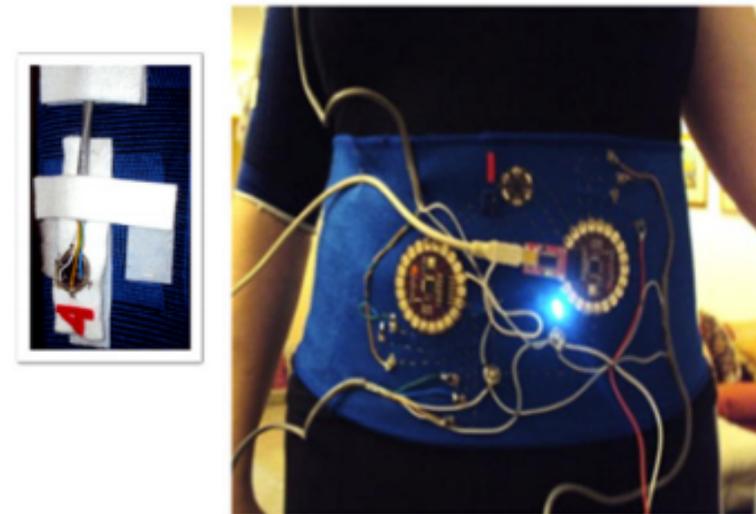
O que podemos fazer
com os dados coletados?

Primeiro exemplo de aplicação [2]

Scheme of positioning and orientation



User wearing the device



O objetivo deste exemplo é construir um classificador capaz de dizer que atividade (**sitting, sitting down, standing, standing up, walking**) uma pessoa está realizando a partir de dados coletados de acelerômetros presentes no corpo desta pessoa.

Pipeline do processo para reconhecimento de atividades [1]



Dados coletados e filtrados

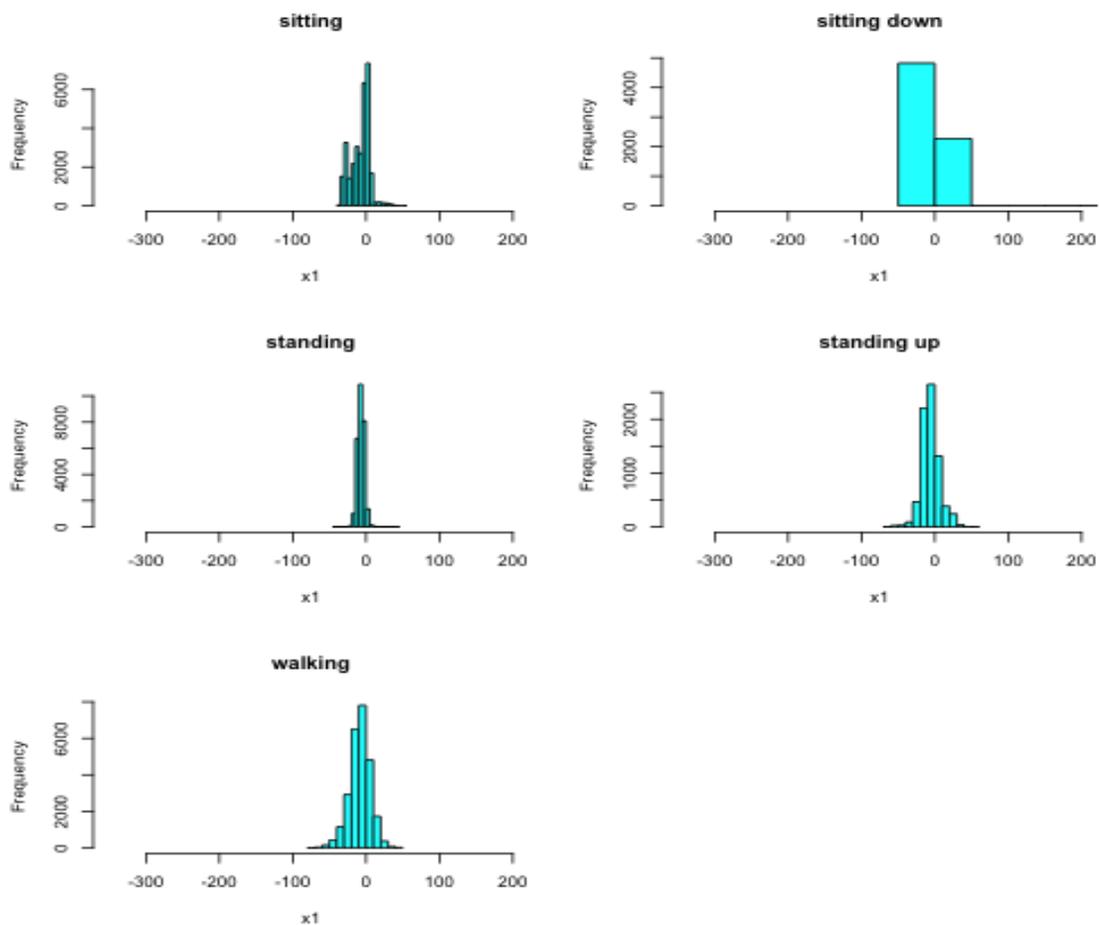
- Dados coletados a partir de 4 tri-axial acelerômetros.
- Foram consideradas janelas de tempo de 1 segundo, com overlapping de 150ms.
- Medidas de roll, pitch e módulo de aceleração foram adquiridas.
- A amostra dentro da janela de tempo foi agrupada e atributos foram gerados (i.e., variância, média).
- Foram filtrados 12 atributos finais - três para cada acelerômetro.

Construção do classificador

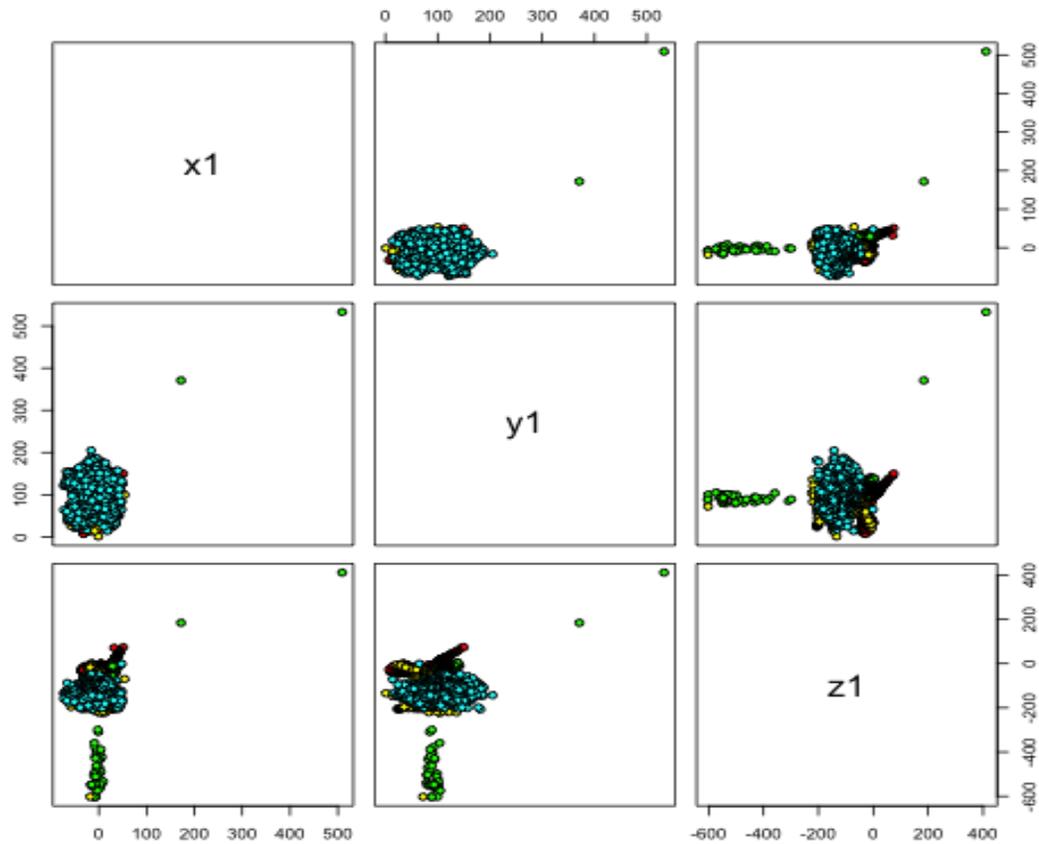
- O dataset possui 165.633 exemplos e 19 atributos:
 - ★ user, gender, age, how_tall_in_meters, weight, body_mass_index, $x_1, y_1, z_1, \dots, x_4, y_4, z_4$, **class**
- O dataset foi dividido em conjunto de treinamento e teste, respeitando a proporção dos valores do atributo **class**.

Referência: <http://rpubs.com/fbarth/har01>

Alguns resultados da análise descritiva



Distribuição das atividades levando-se em consideração dados do sensor



Algoritmo utilizado para criação do modelo: **Random Forest**

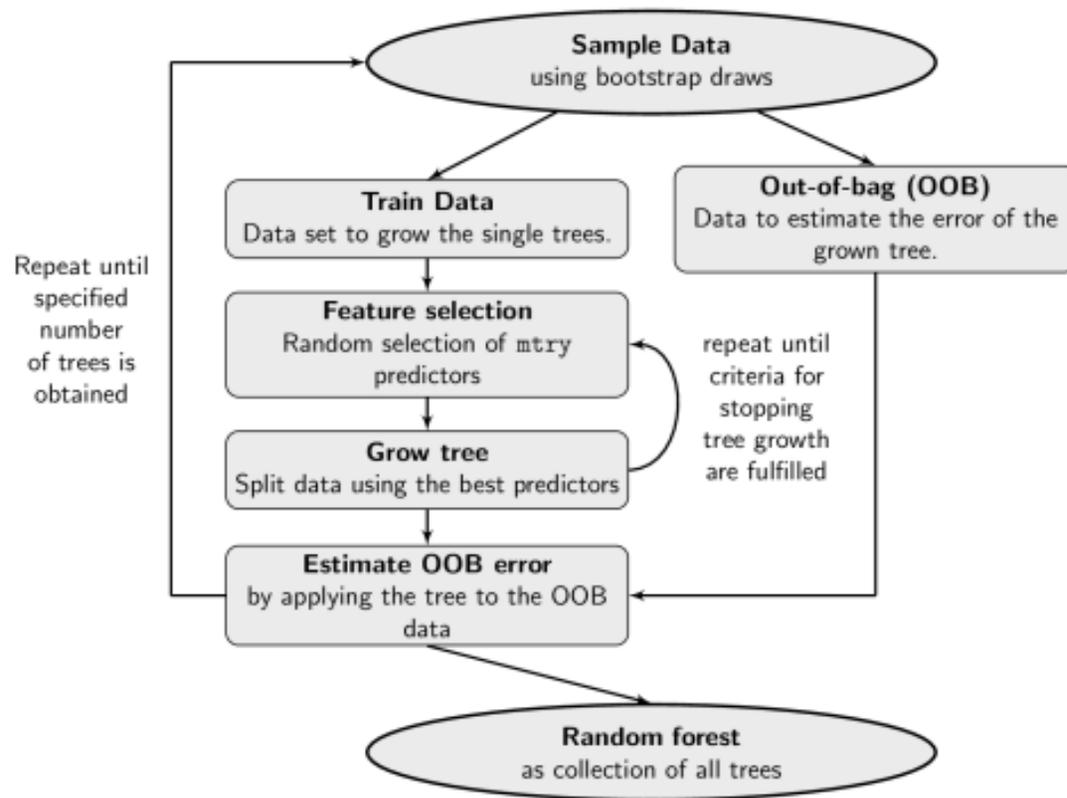


Figure 1: Random Forest Algorithm

Criando o modelo...

```
library(randomForest)
```

```
## randomForest 4.6-7  
## Type rfNews() to see new features/changes/bug fixes.
```

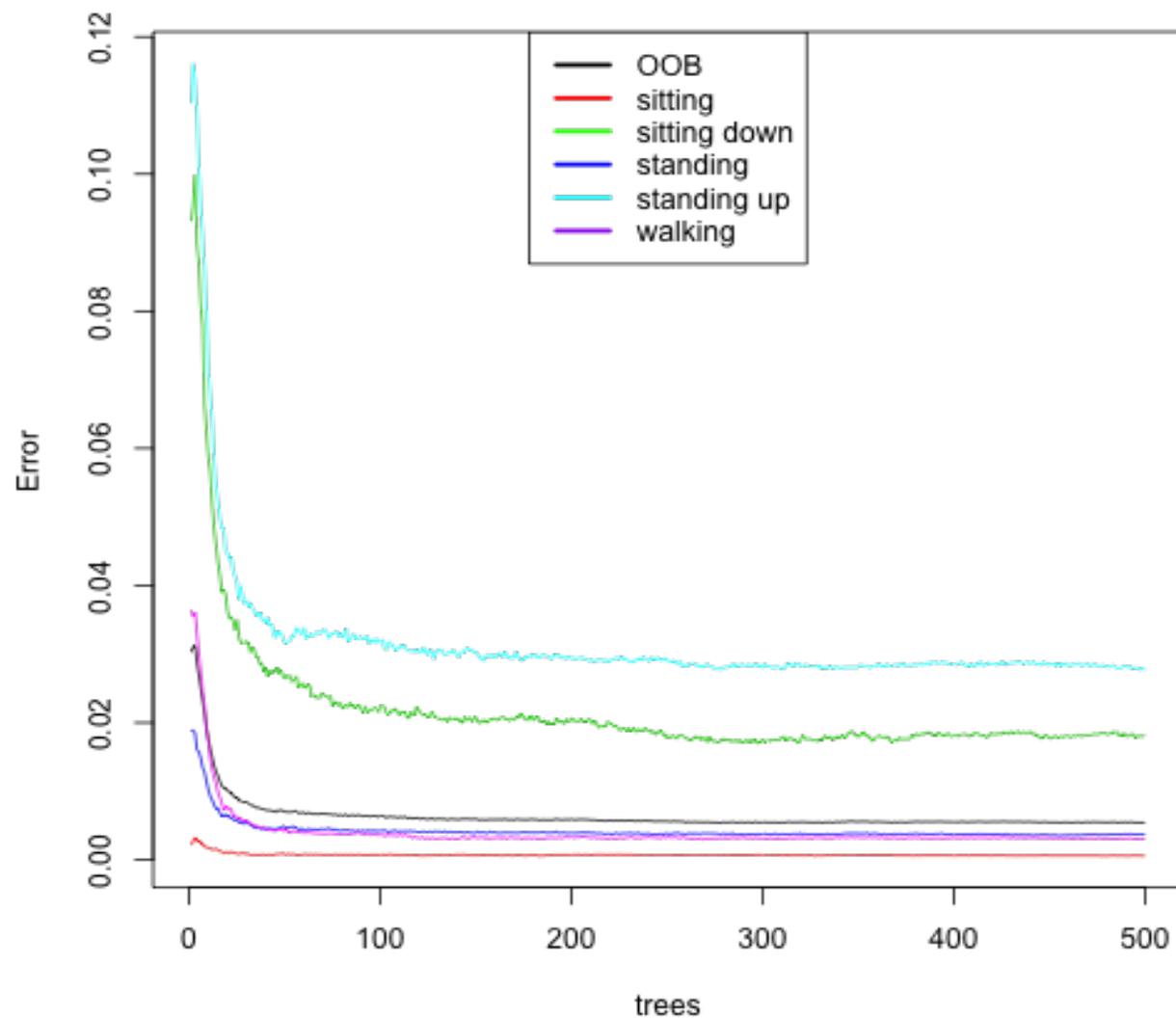
```
formula <- class ~ x1 + y1 + z1 + x2 + y2 + z2 + x3 + y3 + z3 + x4 + y4 + z4  
model <- randomForest(formula, data = treinamento, do.trace = 100, importance = TRUE)
```

```
## ntree      OOB      1      2      3      4      5  
##  100:  0.64%  0.07%  2.25%  0.43%  3.20%  0.37%  
##  200:  0.59%  0.07%  2.07%  0.39%  2.94%  0.34%  
##  300:  0.54%  0.07%  1.76%  0.37%  2.81%  0.30%  
##  400:  0.55%  0.06%  1.82%  0.37%  2.85%  0.32%  
##  500:  0.54%  0.06%  1.82%  0.37%  2.77%  0.30%
```

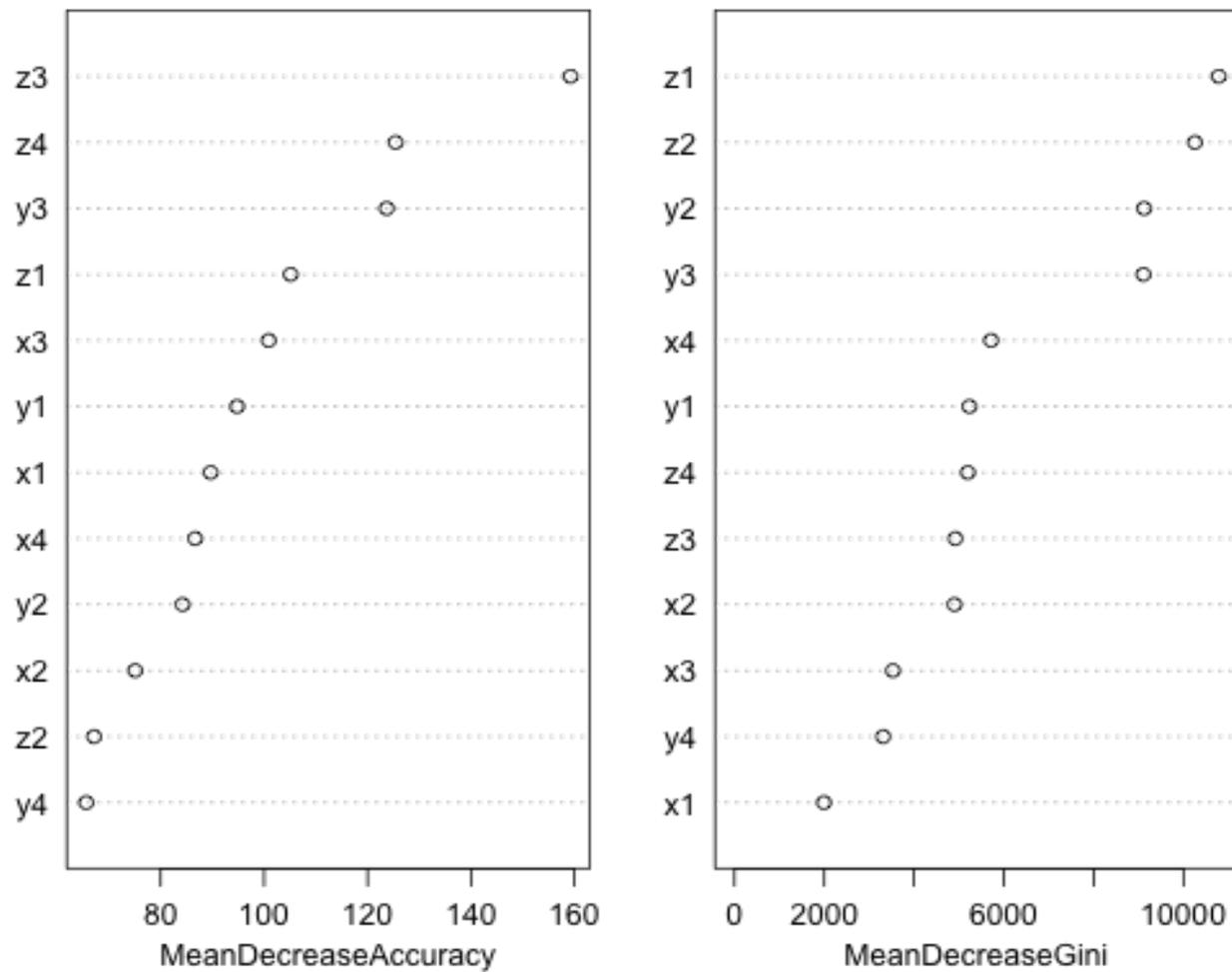
```
model
```

```
##  
## Call:  
## randomForest(formula = formula, data = treinamento, do.trace = 100,      importance =  
TRUE)  
##           Type of random forest: classification  
##           Number of trees: 500  
## No. of variables tried at each split: 3  
##  
##           OOB estimate of error rate: 0.54%  
## Confusion matrix:  
##           sitting sittingdown standing standingup walking class.error  
## sitting      30361          4          0          14          0  0.0005925  
## sittingdown   3         6968         23          60         43  0.0181767  
## standing      0          0        28316          8         98  0.0037295  
## standingup    8          77          53        7243         68  0.0276547  
## walking       0          14          47          18       25955  0.0030345
```

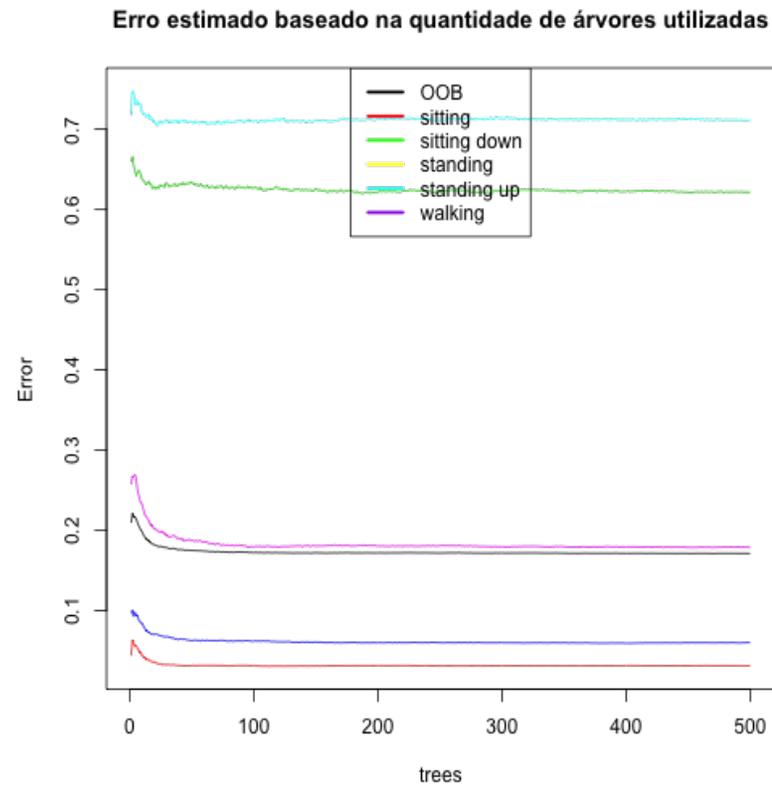
Erro estimado baseado na quantidade de árvores utilizadas



Importância dos atributos ao classificar as observações



Outro modelo



Modelo que utiliza apenas dados do acelerômetro localizado na cintura (Erro estimado: 17.12%)

Validando o modelo completo com o conjunto de testes

```
testPred <- predict(model, newdata = teste)
t <- table(testPred, teste$class)
confusionMatrix(t)
```

```
## Confusion Matrix and Statistics
##
##
## testPred      sitting sittingdown standing standingup walking
## sitting      20236          2          0           3          0
## sittingdown   0         4651          0          54         20
## standing      0          14      18879          39         43
## standingup   16          39           6      4819          9
## walking       0          24          63          51      17284
##
## Overall Statistics
##
##              Accuracy : 0.994
##              95% CI   : (0.994, 0.995)
## No Information Rate : 0.306
## P-Value [Acc > NIR] : <2e-16
##
##
```

Resultados do trabalho original [2]

We used AdaBoost with 10 iterations and configured the C4.5 tree for a confidence factor of 0.25. The overall recognition performance was of 99.4% (weighted average) using a 10-fold cross validation testing mode, with the following accuracies per class: “sitting” 100%, “sitting down” 96.9%, “standing” 99.8%, “standing up” 96.9%, and “walking” 99.8%. The confusion matrix is presented in Table 3.

Table 3. Confusion Matrix

		Predicted class					Actual class
		Sitting	Sitting down	Standing	Standing Up	Walking	
Actual class	Sitting	50,601	9	0	20	1	
	Sitting down	10	11,484	29	297	7	
	Standing	0	4	47,342	11	13	
	Standing up	14	351	24	11,940	85	
	Walking	0	8	27	60	43,295	

The results obtained in this research are very close to the top results of the literature (99.4% in [14], and 99.6% in [15]), even though, it is hard to compare them. Each research used a different dataset, a different set of classes, and different test modes.

Dado o mesmo dataset, será que é possível determinar quem está realizando a atividade?

http://fbarth.net.br/humanActivityRecognition/scripts/har_case01_user.html

Segundo exemplo de aplicação [1]

Os experimentos foram realizados com um grupo de 30 voluntários entre 19-48 anos. Cada pessoa executou seis atividades:

- WALKING: andando
- WALKING_UPSTAIRS: andando escada acima
- WALKING_DOWNSTAIRS: andando escada abaixo
- SITTING: sentado
- STANDING: em pé
- LAYING: deitado

usando um smartphone (Samsung Galaxy II) na cintura.

Com base nos sensores do smartphone, acelerômetro e giroscópio, foram capturados a aceleração linear nos três eixos e a velocidade angular nos três eixos.

Adquirindo os dados

```
load("../data/samsungData.rda")
names(samsungData) <- gsub("\\\\", "_", gsub("\\\\(", "_", gsub(",", "_", gsub("-",
  "_", names(samsungData))))))
samsungData$activity <- as.factor(samsungData$activity)
```

<http://rpubs.com/fbarth/har02>

Separando os dados

```
train <- subset(samsungData, samsungData$subject < 20)
test <- subset(samsungData, samsungData$subject > 20)
train$subject <- NULL
test$subject <- NULL
```

Construindo o modelo

```
library(randomForest)
```

```
## randomForest 4.6-7  
## Type rfNews() to see new features/changes/bug fixes.
```

```
model <- randomForest(activity ~ ., data = train, importance = TRUE, do.trace = 100)
```

```
## ntree      OOB      1      2      3      4      5      6  
##   100:    2.17%  0.00%  3.31%  4.84%  1.41%  2.39%  1.01%  
##   200:    1.79%  0.00%  3.31%  4.15%  1.41%  1.10%  0.50%  
##   300:    1.72%  0.00%  3.01%  4.01%  1.13%  1.10%  0.84%  
##   400:    1.59%  0.00%  2.26%  3.87%  1.41%  1.10%  0.67%  
##   500:    1.51%  0.00%  2.26%  3.73%  1.41%  0.92%  0.50%
```

Construindo o modelo

```
model
```

```
##  
## Call:  
## randomForest(formula = activity ~ ., data = train, importance = TRUE,      do.trace =  
100)  
##           Type of random forest: classification  
##           Number of trees: 500  
## No. of variables tried at each split: 23  
##  
##           OOB estimate of  error rate: 1.51%  
## Confusion matrix:  
##           laying sitting standing walk walkdown walkup class.error  
## laying      731         0         0         0         0         0         0.000000  
## sitting      0        649         15         0         0         0         0.022590  
## standing     0         27        696         0         0         0         0.037344  
## walk         0         0         0        697         7         3         0.014144  
## walkdown     0         0         0         5        538         0         0.009208  
## walkup       0         0         0         1         2        594         0.005025
```

Validando o modelo

```
testPred <- predict(model, newdata = test)
t <- table(testPred, test$activity)
confusionMatrix(t)
```

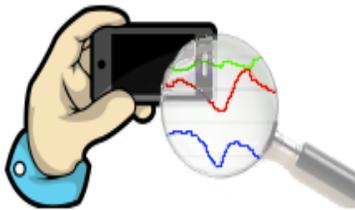
```
## Confusion Matrix and Statistics
##
##
## testPred      laying sitting standing walk walkdown walkup
## laying        676      0      0      0      0      0
## sitting        0      578     70      0      0      0
## standing       0      44     581     0      0     23
## walk           0      0      0     471     14      9
## walkdown      0      0      0      4     335      6
## walkup        0      0      0     44      94     438
##
## Overall Statistics
##
##              Accuracy : 0.909
##              95% CI   : (0.899, 0.919)
## No Information Rate : 0.2
## P-Value [Acc > NIR] : <2e-16
##
## .....
```

Comparando com o artigo original [1]

Method	MC-SVM							MC-HF-SVM $k = 8$ bits						
Activity	Walking	Upstairs	Downstairs	Standing	Sitting	Laying	Recall %	Walking	Upstairs	Downstairs	Standing	Sitting	Laying	Recall %
Walking	109	0	5	0	0	0	95.6	109	2	3	0	0	0	95.6
Upstairs	1	95	40	0	0	0	69.8	1	98	37	0	0	0	72.1
Downstairs	15	9	119	0	0	0	83.2	15	14	114	0	0	0	79.7
Standing	0	5	0	132	5	0	93.0	0	5	0	131	6	0	92.2
Sitting	0	0	0	4	108	0	96.4	0	1	0	3	108	0	96.4
Laying	0	0	0	0	0	142	100	0	0	0	0	0	142	100
<i>Precision %</i>	87.2	87.2	72.6	97.1	95.6	100	89.3	87.2	81.7	74.0	97.8	94.7	100	89.0

Table 1. Confusion Matrix of the classification results on the test data using the traditional floating-point MC-SVM (Left) and the MC-HF-SVM with $k = 8$ bits (Right). Rows represent the actual class and columns the predicted class. The diagonal entries (in bold) show the number of test samples correctly classified.

Kaggle: Accelerometer Biometric Competition



Completed • \$5,000 • 633 teams

Accelerometer Biometric Competition

Tue 23 Jul 2013 – Fri 22 Nov 2013 (12 months ago)

Dashboard

Home



Data



Make a submission



Information



Description

Evaluation

Rules

Prizes

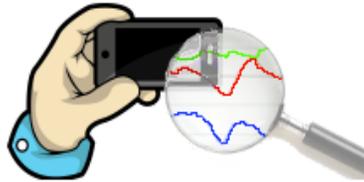
About the Sponsor

More on Accelerometer D...

Competition Details » [Get the Data](#) » [Make a submission](#)

Recognize users of mobile devices from accelerometer data

Since everyone moves differently and accelerometers are fast becoming ubiquitous, this competition is designed to investigate the feasibility of using accelerometer data as a biometric for identifying users of mobile devices.



Completed • \$5,000 • 633 teams

Accelerometer Biometric Competition

Tue 23 Jul 2013 – Fri 22 Nov 2013 (12 months ago)

Dashboard

Public Leaderboard - Accelerometer Biometric Competition

This leaderboard is calculated on approximately 30% of the test data.
The final results will be based on the other 70%, so the final standings may be different.

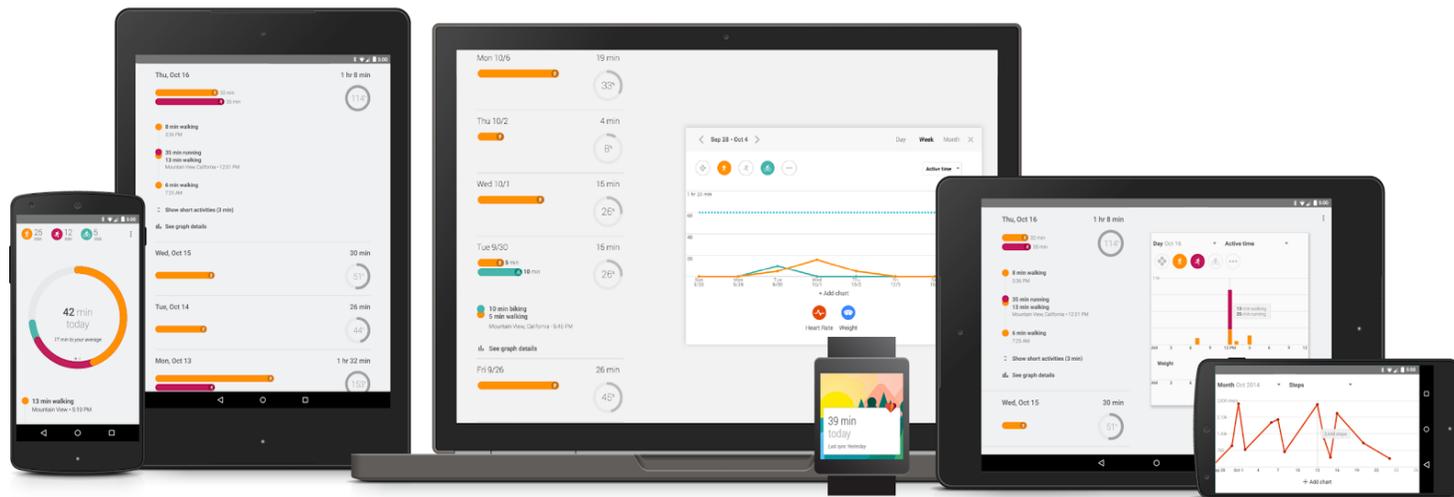
See someone using multiple accounts?
[Let us know.](#)

#	Δ1w	Team Name <small>* in the money</small>	Score <small>👁</small>	Entries	Last Submission UTC (Best – Last Submission)
1	—	gaddawin *	0.99905	108	Fri, 22 Nov 2013 19:34:13 (-1.6h)
2	↑1	Solórzano *	0.99898	62	Fri, 22 Nov 2013 23:26:39
3	↓1	Target Leak Model *	0.99782	63	Fri, 22 Nov 2013 07:17:54 (-23.8h)
4	↑1	Dan Stahlke *	0.99659	48	Thu, 21 Nov 2013 00:55:00 (-22.4h)
5	↓1	Dieselboy	0.99555	338	Fri, 22 Nov 2013 22:56:48 (-3.7h)
6	↑1	Victor	0.99456	43	Fri, 22 Nov 2013 23:01:34
7	↑10	YS-L 👤	0.99289	36	Thu, 21 Nov 2013 15:10:37 (-0.1h)
8	↑44	Guillaume BS	0.98853	36	Fri, 22 Nov 2013 22:37:44 (-0.4h)

Google Fit, Google Fit
SDK, funf

Google Fit: lançado em 28/10/2014

Google Fit
on phone, tablet, web & Wear



Google Fit SDK

The Google Fit SDK

New APIs to make building fitness apps and devices easier.



Discover sensors

Easily view available sensor data sources from connected apps and devices with the Sensors API.



Collect activity data

Connect your app and devices to Google Fit with the Recording API.



Help users keep track

Access and edit the user's fitness history with the History API.

funf: Open Sensing Framework



[About](#) [Funf Journal](#) [Funf In a Box](#) [Developers](#) [Blog](#) [Contact](#)

Introducing, Behavior!

With **funding** from the **Knight Foundation** we are proud to announce that we are embarking on an exciting new venture.

Behavior will work towards opening access to, and helping make sense of, the data routinely collected by mobile phones. This will also allow us to continue developing the open source Funf framework and support its growing community while showing what can be done when our smartphone apps are actually smart.

[Learn More >](#)

behavior

Perguntas?

Obrigado!

- Todo o material (slides e código) aqui apresentado é *Copyleft*
- O código fonte é encontrado em <http://github.com/fbarth>
- Demais materias são encontrado em <http://fbarth.net.br>
- fabricio dot barth at gmail dot com

Disclaimer

- Muita coisa aconteceu de 2014 até 2021 sobre este assunto!
- A revisão deste documento em 2021 foi apenas uma revisão tipográfica. Não fez parte deste trabalho nenhuma revisão bibliográfica.

References

- [1] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and JorgeL. Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In José Bravo, Ramón Hervás, and Marcela Rodríguez, editors, *Ambient Assisted Living and Home Care*, volume 7657 of *Lecture Notes in Computer Science*, pages 216–223. Springer Berlin Heidelberg, 2012.

- [2] Wallace Ugulino, Débora Cardador, Katia Vega, Eduardo Velloso, Ruy Milidiú, and Hugo Fuks. Wearable computing: Accelerometers' data classification of body postures and movements. In LelianeN. Barros, Marcelo Finger, AuroraT. Pozo, GustavoA. Giménez-Lugo, and Marcos Castilho, editors, *Advances in Artificial Intelligence - SBIA 2012*, Lecture Notes in Computer Science, pages 52–61. Springer Berlin Heidelberg, 2012.