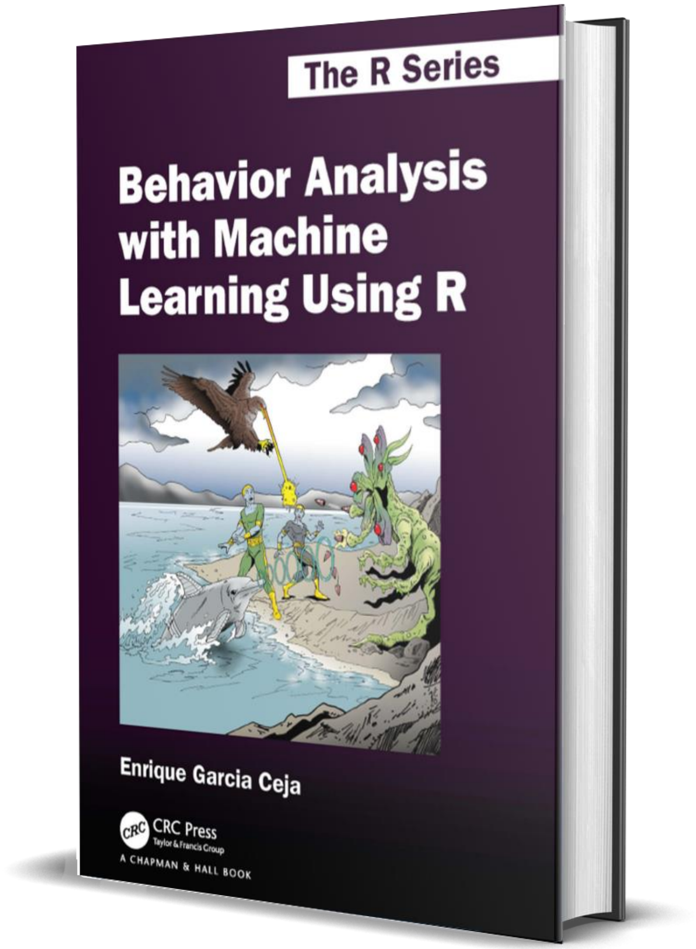
DETECTING ABNORMAL FISH BEHAVIORS WITH ISOLATION FORESTS

Enrique Garcia Ceja (enriquegc.com)
Tecnologico de Monterrey University, Mexico.

CODE

Slides and code available: https://github.com/enriquegit/fish-behaviors

This talk is based on Chapter 10 of my book "*Behavior Analysis with Machine Learning Using R*"



Free online full book: https://enriquegit.github.io/behavior-free/

In marine biology, the analysis of fish behavior is essential since it can be used to detect environmental changes produced by pollution, climate change, etc.

Fish behaviors can be characterized by their trajectories, that is, how they move within the environment.

TRAJECTORY

A trajectory is the path that an object follows through space and time.

A trajectory has two main components:

1. coordinates
2. timesteps

TRAJECTORY

A trajectory is the path that an object follows through space and time.

A trajectory has two main components:

1. coordinates
2. timesteps

FISH TRAJECTORIES

Capturing fish trajectories is a challenging task, specially in underwater conditions.

Thankfully, the Fish4Knowledge project has developed fish analysis tools and methods to ease the task!

Project url: https://homepages.inf.ed.ac.uk/rbf/fish4knowledge/

FISH4KNOWLEDGE

They have processed enormous amounts of video streaming data and have extracted fish information (including trajectories) observed in the Taiwanese coral reef.

THE DATASET

- 3102 trajectories belonging to the *Dascyllus reticulatus* fish.

- Each trajectory is labeled as ‘*normal*’ or ‘*abnormal*’.

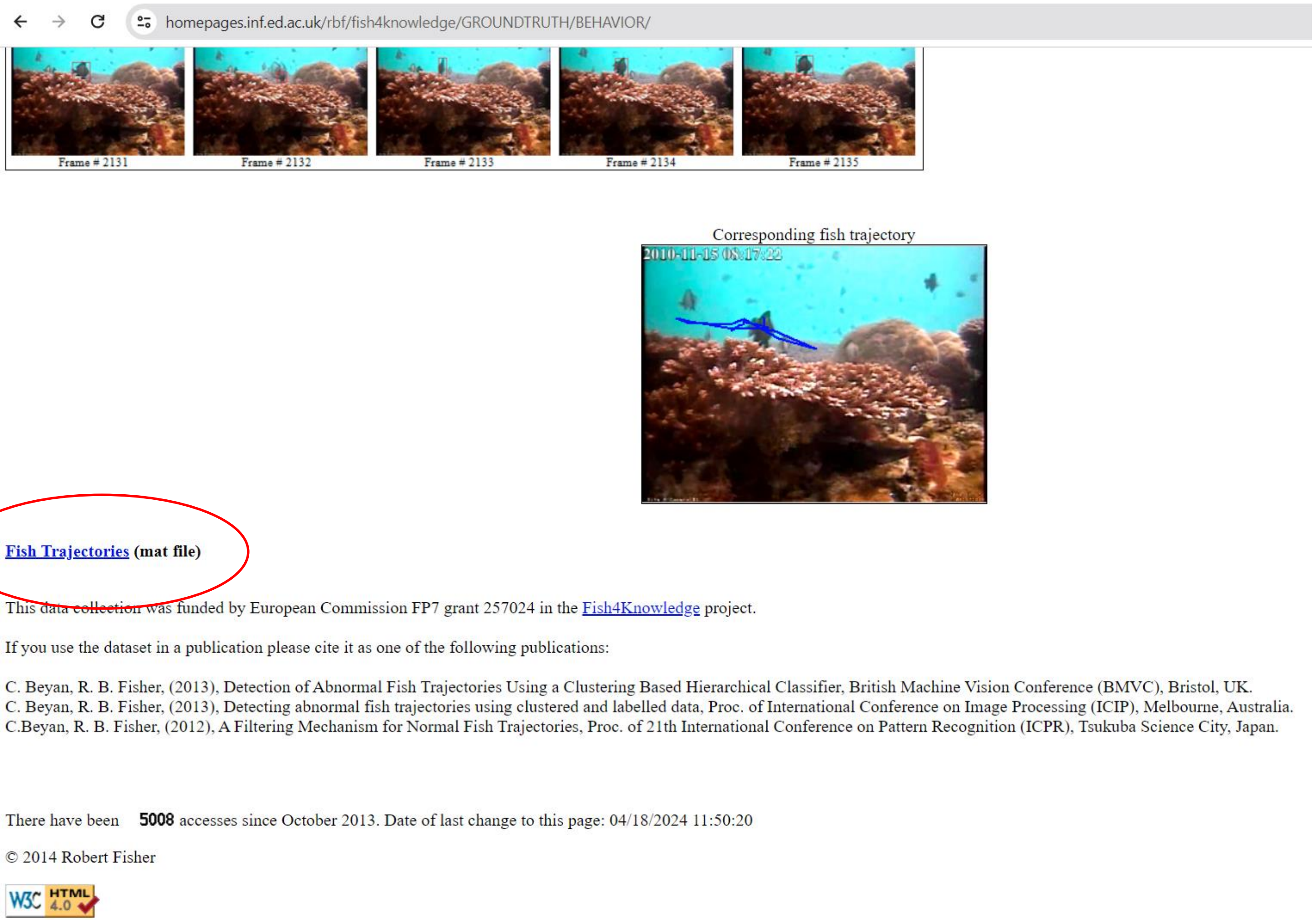
THE DATASET

- The ‘*normal*’ and ‘*abnormal*’ labels were determined by visual inspection by experts.

- The *abnormal* cases include events such as predator avoidance and aggressive movements (due to another fish or because of being frightened).

DOWNLOADING THE DATASET

The dataset can be downloaded from:
https://homepages.inf.ed.ac.uk/rbf/fish4knowledge/GROUNDTRUTH/BEHAVIOR/



Corresponding fish trajectory

Fish Trajectories **(mat file)**

This data collection was funded by European Commission FP7 grant 257024 in the Fish4Knowledge project.

If you use the dataset in a publication please cite it as one of the following publications:

C. Beyan, R. B. Fisher, (2013), Detection of Abnormal Fish Trajectories Using a Clustering Based Hierarchical Classifier, British Machine Vision Conference (BMVC), Bristol, UK.
C. Beyan, R. B. Fisher, (2013), Detecting abnormal fish trajectories using clustered and labelled data, Proc. of International Conference on Image Processing (ICIP), Melbourne, Australia.
C.Beyan, R. B. Fisher, (2012), A Filtering Mechanism for Normal Fish Trajectories, Proc. of 21th International Conference on Pattern Recognition (ICPR), Tsukuba Science City, Japan.

There have been **5008** accesses since October 2013. Date of last change to this page: 04/18/2024 11:50:20

© 2014 Robert Fisher

READING THE TRAJECTORIES

script: visualize_fish.R

```
library(R.matlab)
# Read data.
df <- readMat("fishDetections_total3102.mat")$fish.detections


# Print data frame dimensions.
dim(df)
```

Output:

```
[1]   7    1 3102
```

READING THE TRAJECTORIES

script: visualize_fish.R

```
trj <- df[,,1] # Read one of the trajectories.

str(trj) # Inspect its structure.
```

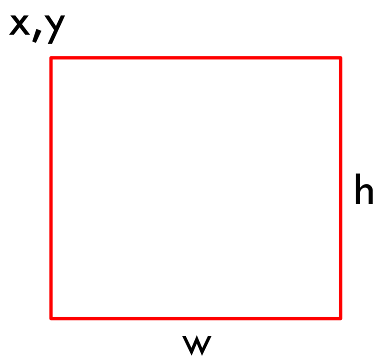
Output:

```
List of 7
 $ frame.number    : num [1:37, 1] 826 827 828 829 833 834 835 836 837 840 ...
 $ bounding.box.x  : num [1:37, 1] 167 165 162 159 125 124 126 126 127 128 ...
 $ bounding.box.y  : num [1:37, 1] 67 65 65 66 58 61 65 71 71 62 ...
 $ bounding.box.w  : num [1:37, 1] 40 37 39 34 39 39 38 38 37 31  ...
 $ bounding.box.h  : num [1:37, 1] 38 40 40 38 35 34 34 33 34 35 ...
 $ class           : num [1, 1] 1
 $ classDescription: chr [1, 1] "normal"
```

ATTRIBUTES

script: visualize_fish.R

1. frame.number: Frame number in original video.
2. bounding.box.x: Bounding box leftmost edge.
3. bounding.box.y: Bounding box topmost edge.
4. bounding.box.w: Bounding box width.
5. bounding.box.h: Bounding box height.
6. class: 1=normal, 2=rare.
7. classDescription: ‘normal’ or abnormal’.

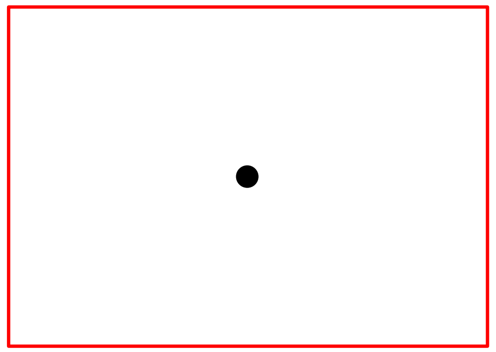


Note that the .mat file does not contain the images but only the bounding boxes’ information.

PLOTTING TRAJECTORIES

script: visualize_fish.R

Each trajectory is represented as a rectangle. Before plotting, we need the coordinates of the middle point (the center of the rectangle).



```r
# Compute center of bounding box.

x.coord <- trj$bounding.box.x + (trj$bounding.box.w / 2)
y.coord <- trj$bounding.box.y + (trj$bounding.box.h / 2)
```

PLOTTING TRAJECTORIES

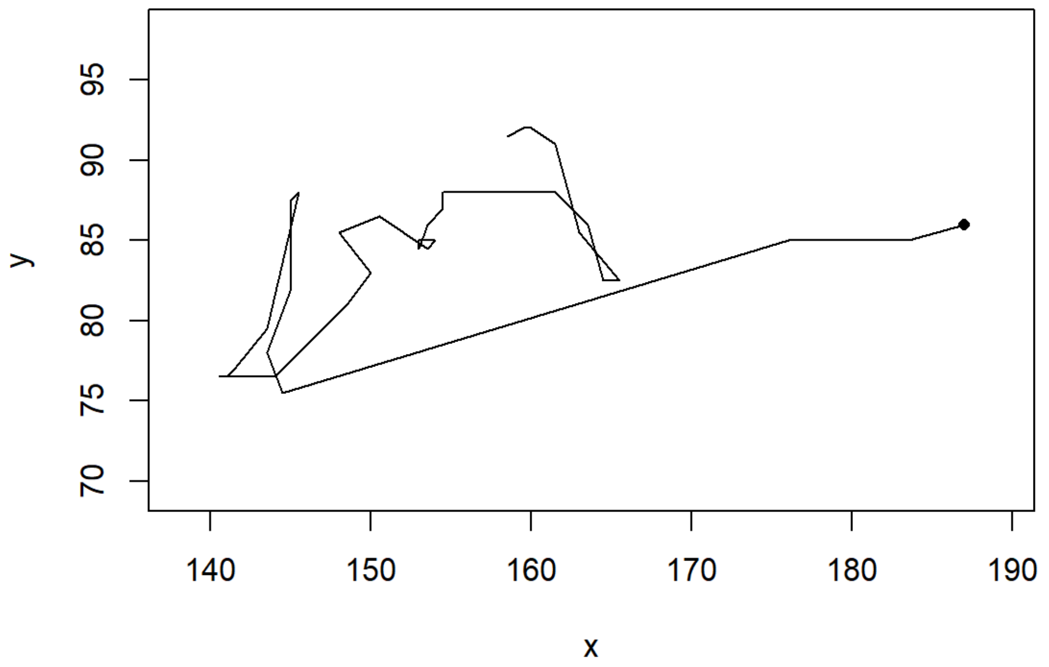
script: visualize_fish.R

```
library(trajr)

# Make times start at 0.
times <- trj$frame.number - trj$frame.number[1]

# Store trajectory in a data frame.
tmp <- data.frame(x.coord, y.coord, time=times)

# Create a trajectory object.
trj.obj <- TrajFromCoords(tmp, fps = 1)
```

PLOTTING TRAJECTORIES

script: visualize_fish.R

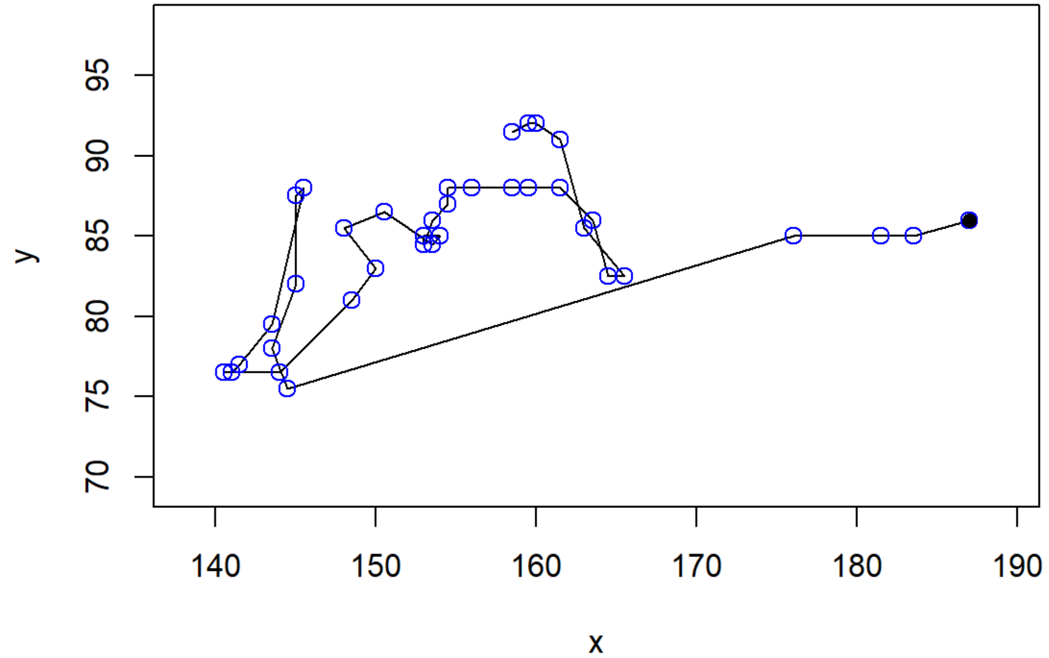
```
# Plot the  trajectory object.
plot(trj.obj, lwd = 1, xlab="x", ylab="y")
```



PLOTTING TRAJECTORIES

script: visualize_fish.R

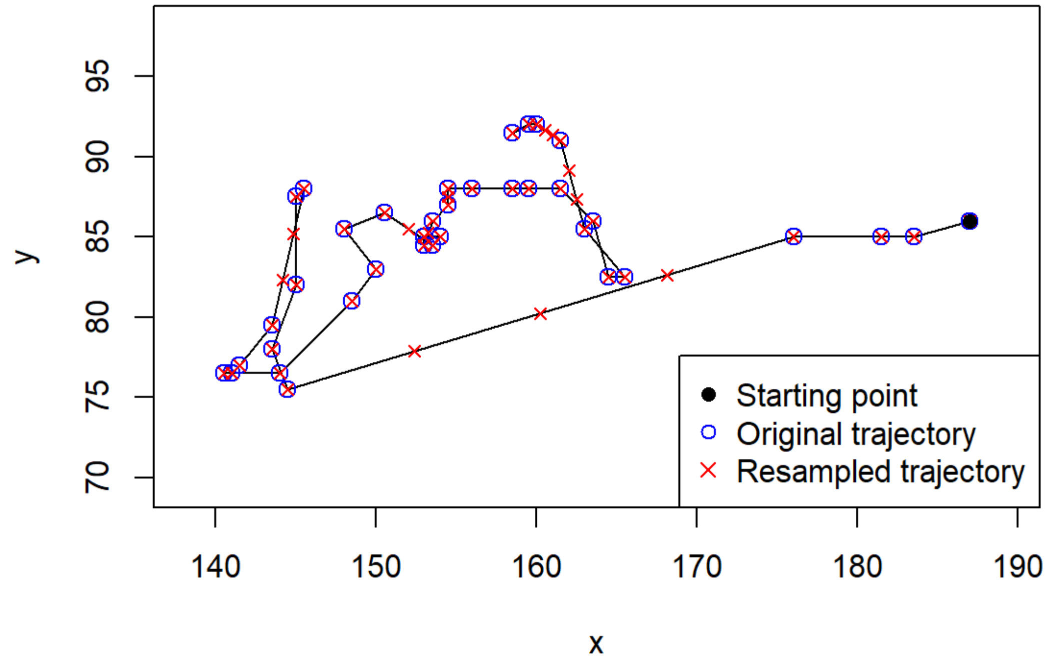
```
# Plot the  trajectory object.
plot(trj.obj, lwd = 1, xlab="x", ylab="y")
points(trj.obj, draw.start.pt = T,
                pch = 1, col = "blue", cex = 1.2)
```



LINEAR INTERPOLATION

script: visualize_fish.R

```
# Linear interpolation.

resampled <- TrajResampleTime(trj.obj, 1)
points(resampled, pch = 4, col = "red", cex = 0.8)
```



ISOLATION FORESTS

An isolation forest is an algorithm for anomaly detection (Liu, F. T., Ting, K. M., & Zhou, Z. H., 2008).

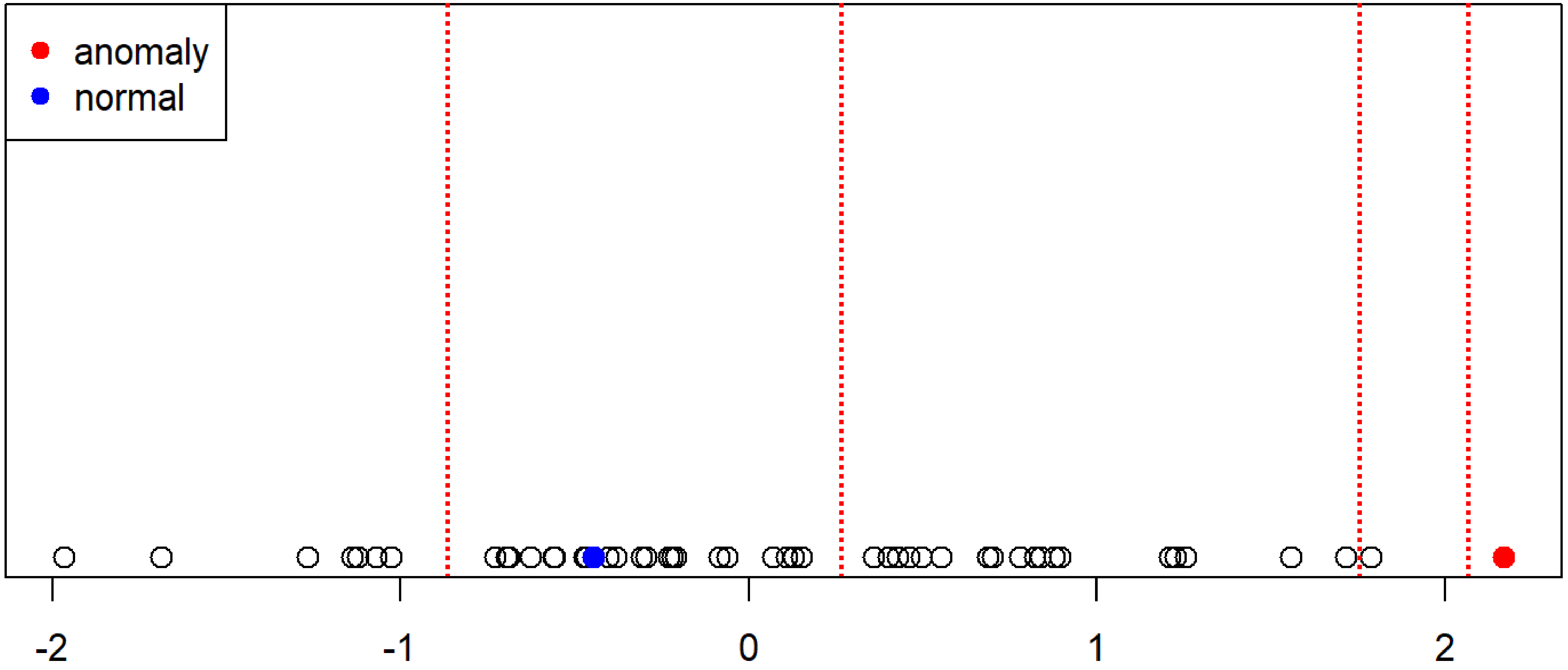
- The algorithm works by *isolating* anomalous points.
- This method is based on trees and consequently, no variable normalization is required.

Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008, December). Isolation forest. In 2008 eighth ieee international conference on data mining (pp. 413-422). IEEE.

ISOLATION FORESTS

Key idea: Try to isolate every data point by drawing a line at a random position.

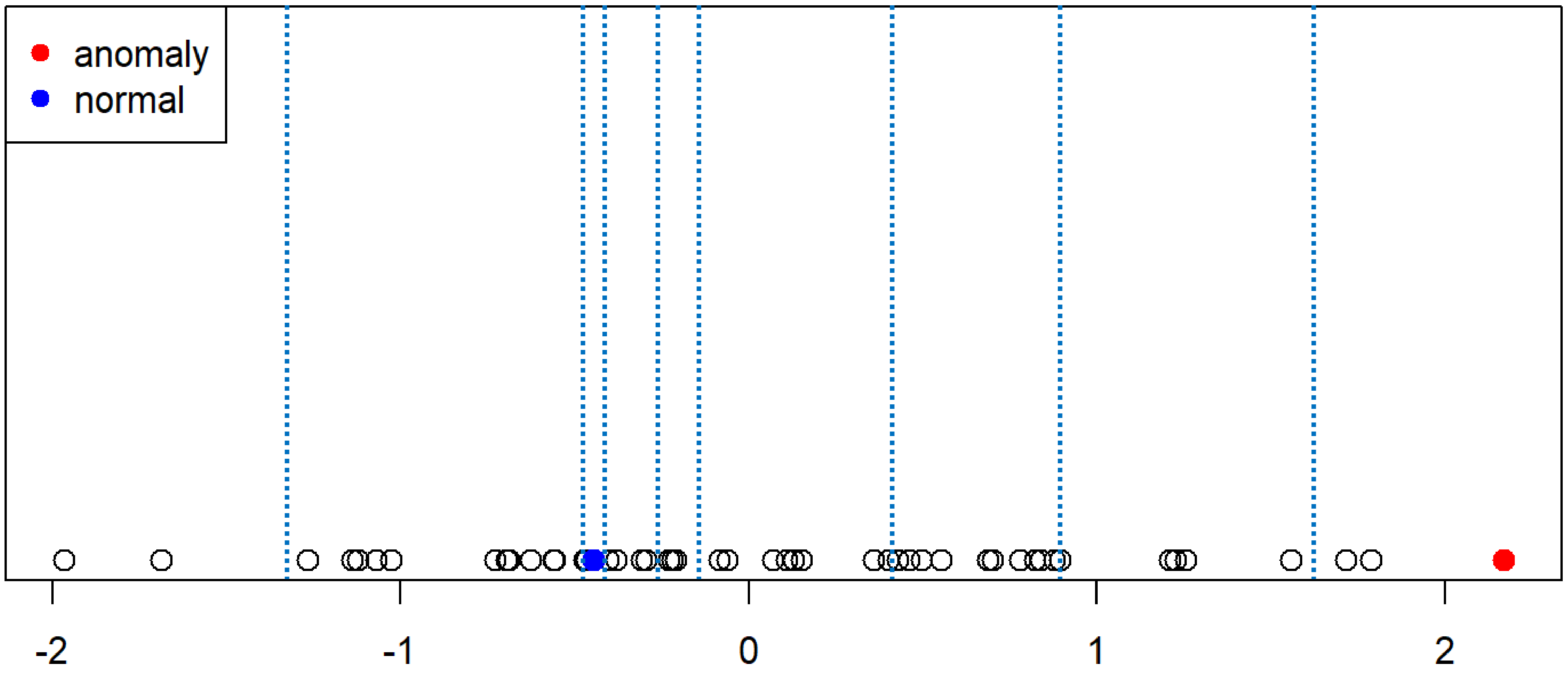
It took 4 tries to isolate the anomalous point.



ISOLATION FORESTS

Key idea: Try to isolate every data point by drawing a line at a random position.

It took 8 tries to isolate the normal point.



ISOLATION FORESTS

- A tree is generated recursively by randomly selecting a feature (variable) and then selecting a partition point between the maximum and minimum value of that feature.

- Each partition corresponds to a split point in a tree (the isolation tree).

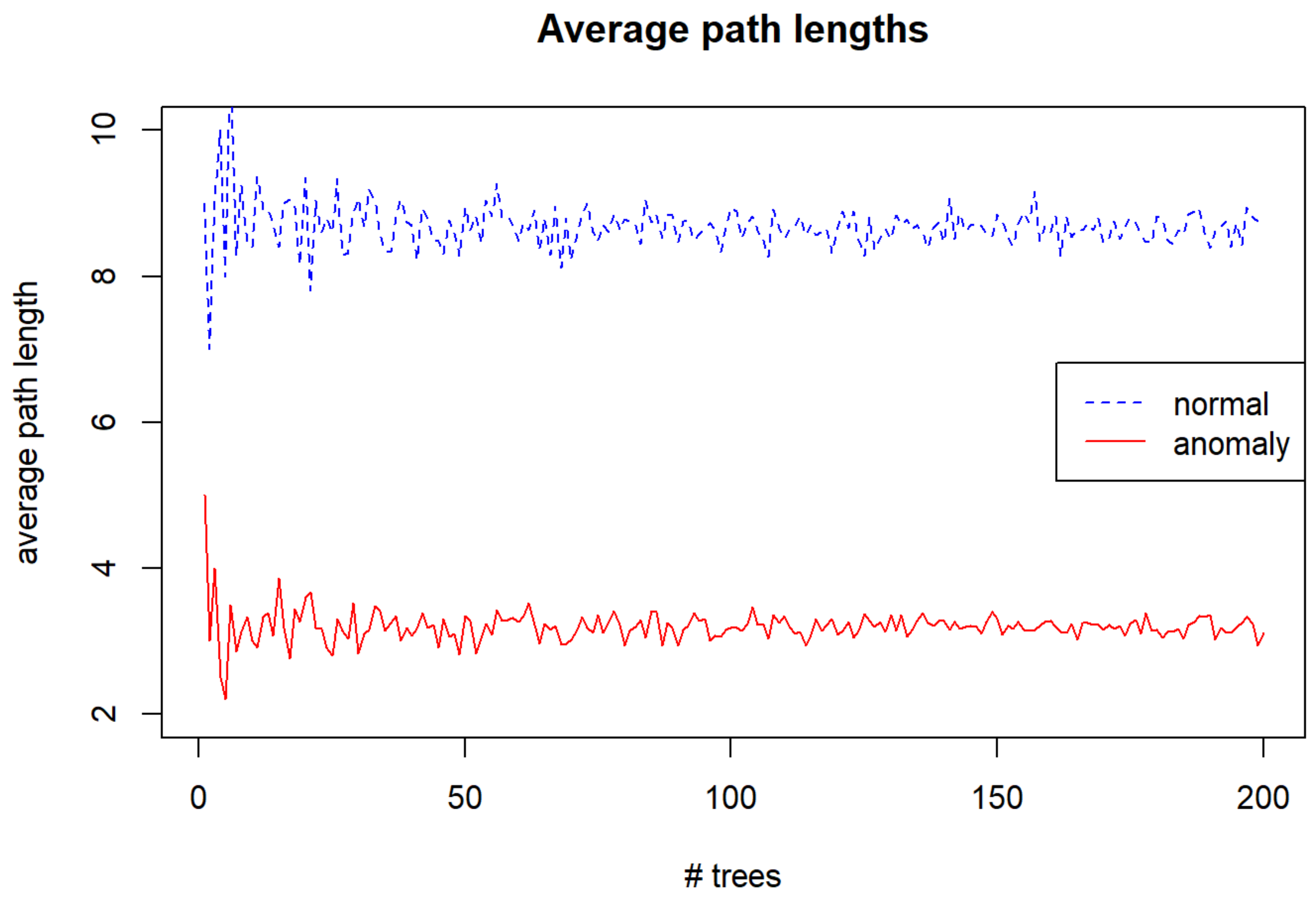
- The procedure terminates when all instances are isolated.

- The number of partitions (split lines) that were required to isolate a point corresponds to the path length of that point to the root of the tree.

- Since anomalies are easier to isolate they will be closer to the root of the tree. An instance is marked as an anomaly if it has a short average path length to the root across many Isolation Trees.

ISOLATION FORESTS

Average path length (number of splits required to isolate the point) for the normal and anomalous point.



ISOLATION FORESTS

Considerations:

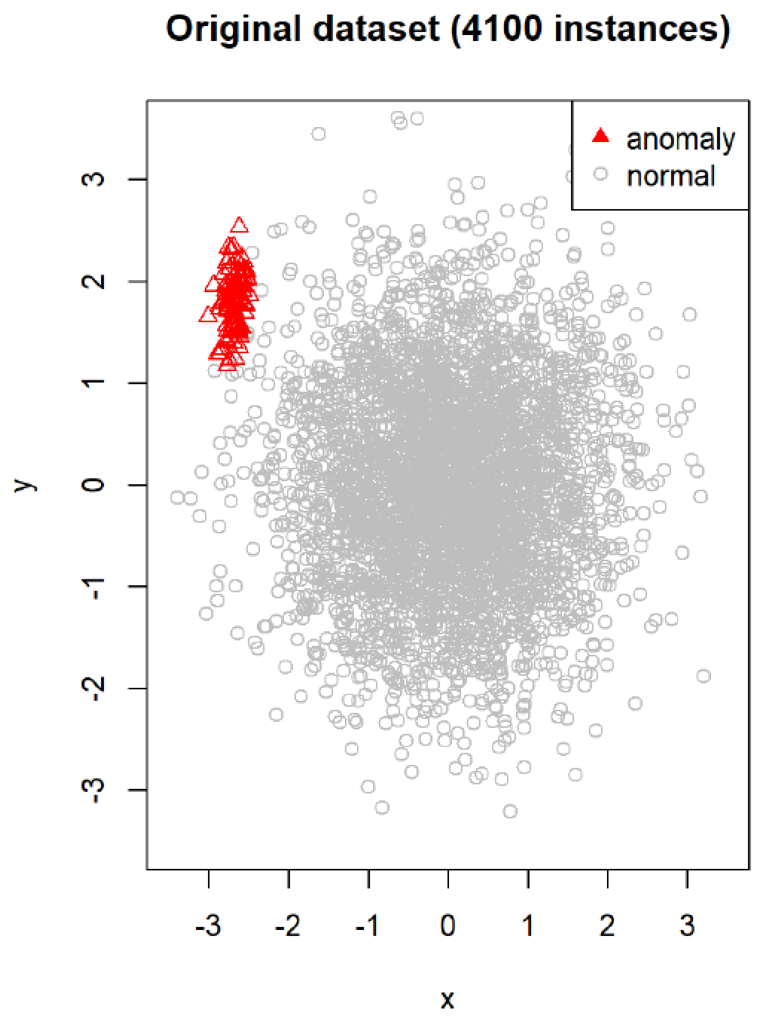
- Instead of directly using the average path lengths for deciding whether or not an instance is an anomaly, the authors of the method proposed an anomaly score that is between 0 and 1.

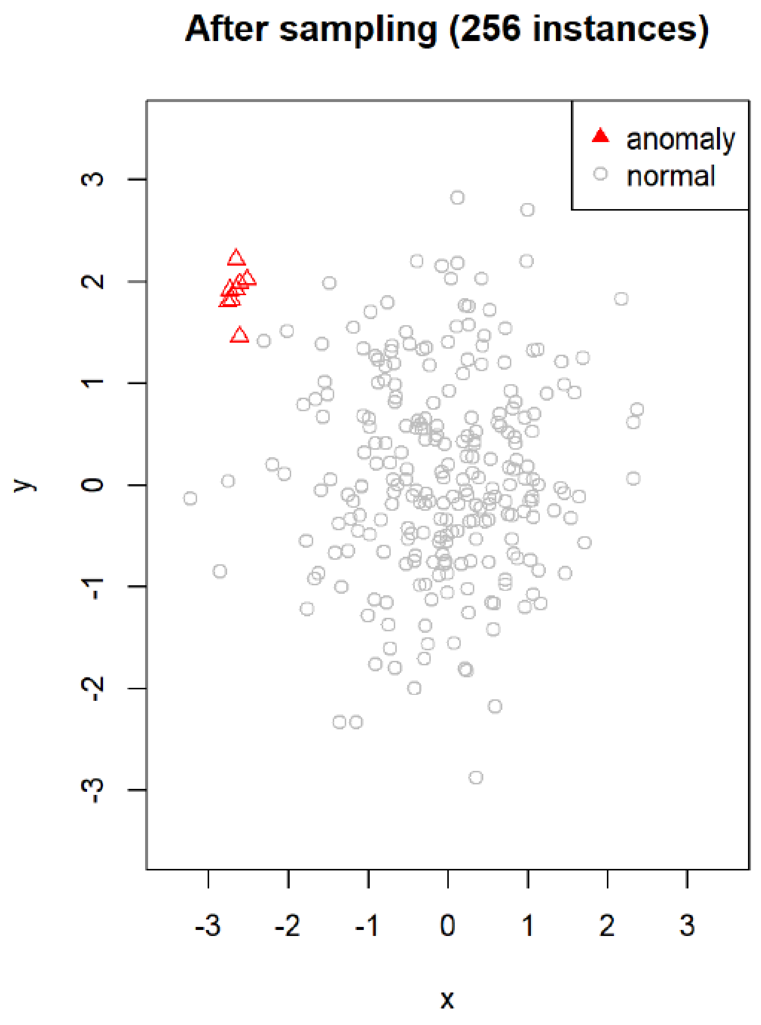
- The closer the anomaly score is to 1 the more likely the instance is an anomaly.

SWAMPING AND MASKING

- **Swamping:** normal instances are too close to anomalies and thus, marked as anomalies.

- **Masking:** the presence of too many anomalies close together.





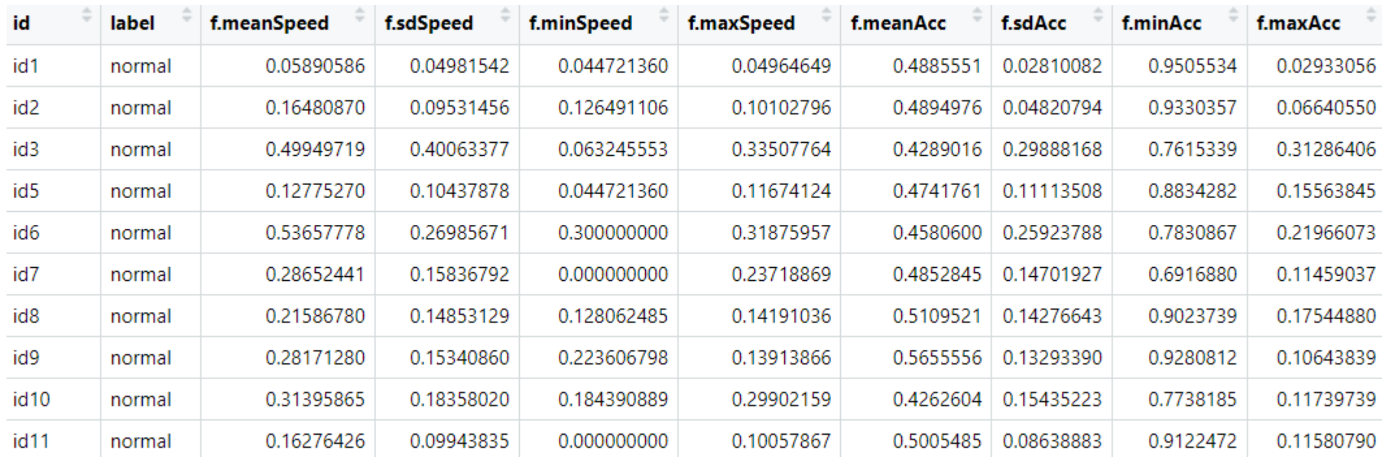
The only two parameters of the algorithm are the number of trees and the sampling size.

FEATURE EXTRACTION

script: extract_featrues.R

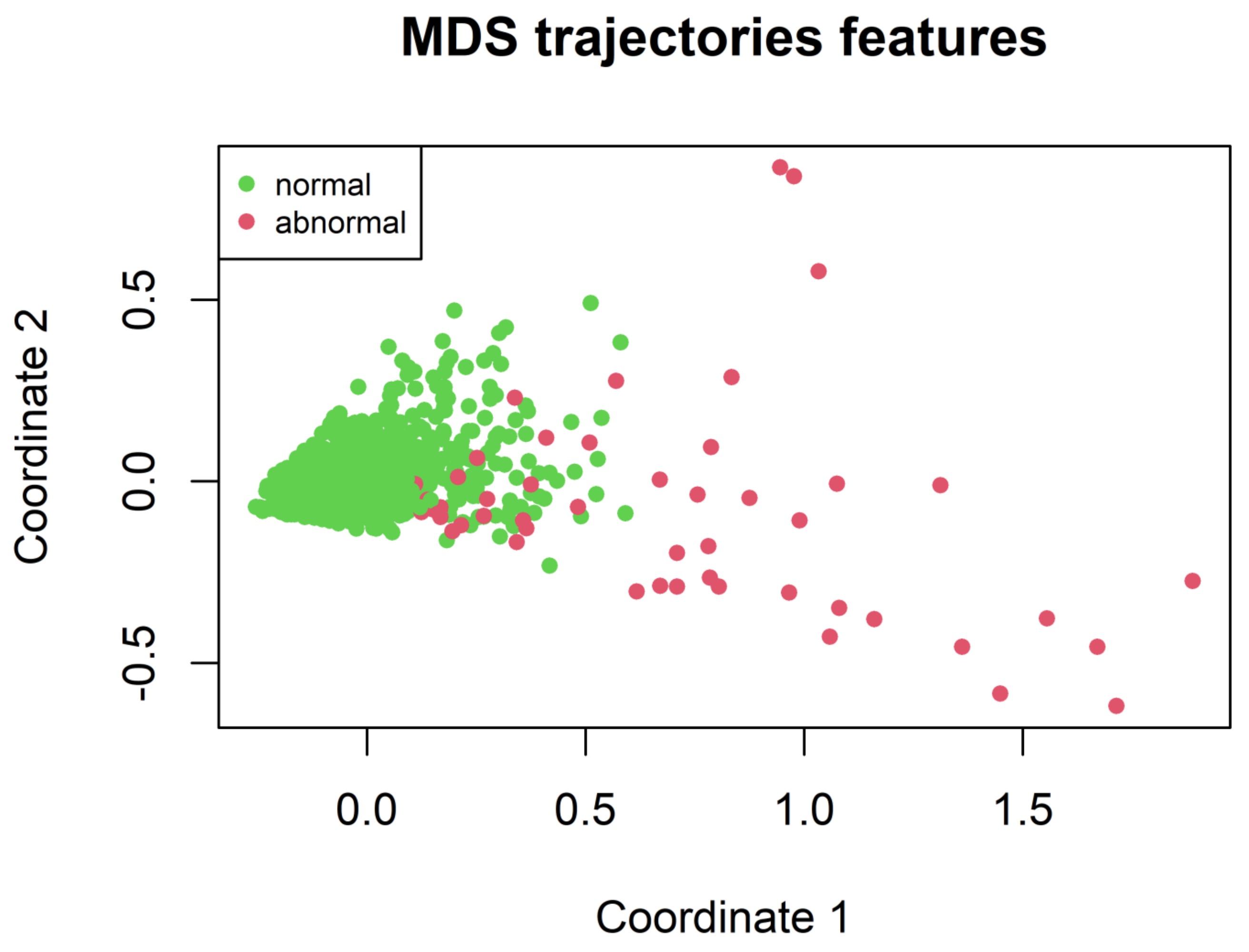
Isolation forests work with features, thus, we need to extract features from the trajectories.

```
derivs <- TrajDerivatives(resampled)
f.meanSpeed <- mean(derivs$speed)
f.sdSpeed <- sd(derivs$speed)
f.minSpeed <- min(derivs$speed)
f.maxSpeed <- max(derivs$speed)
f.meanAcc <- mean(derivs$acceleration)
f.sdAcc <- sd(derivs$acceleration)
f.minAcc <- min(derivs$acceleration)
f.maxAcc <- max(derivs$acceleration)
```



id	label	f.meanSpeed	f.sdSpeed	f.minSpeed	f.maxSpeed	f.meanAcc	f.sdAcc	f.minAcc	f.maxAcc
id1	normal	0.05890586	0.04981542	0.044721360	0.04964649	0.4885551	0.02810082	0.9505534	0.02933056
id2	normal	0.16480870	0.09531456	0.126491106	0.10102796	0.4894976	0.04820794	0.9330357	0.06640550
id3	normal	0.49949719	0.40063377	0.063245553	0.33507764	0.4289016	0.29888168	0.7615339	0.31286406
id5	normal	0.12775270	0.10437878	0.044721360	0.11674124	0.4741761	0.11113508	0.8834282	0.15563845
id6	normal	0.53657778	0.26985671	0.300000000	0.31875957	0.4580600	0.25923788	0.7830867	0.21966073
id7	normal	0.28652441	0.15836792	0.000000000	0.23718869	0.4852845	0.14701927	0.6916880	0.11459037
id8	normal	0.21586780	0.14853129	0.128062485	0.14191036	0.5109521	0.14276643	0.9023739	0.17544880
id9	normal	0.28171280	0.15340860	0.223606798	0.13913866	0.5655556	0.13293390	0.9280812	0.10643839
id10	normal	0.31395865	0.18358020	0.184390889	0.29902159	0.4262604	0.15435223	0.7738185	0.11739739
id11	normal	0.16276426	0.09943835	0.000000000	0.10057867	0.5005485	0.08638883	0.9122472	0.11580790

VISUALIZE FEATURES



Normal: 1093
Abnormal: 54

DATA PARTITION

script: isolation_forest_fish.R

Since abnormal cases are limited, they all will be used for the test set.

The training set will consist of 80% of the data.

ISOLATION FOREST

script: isolation_forest_fish.R

```
library(solitude)

m.iforest <- isolationForest$new(sample_size = 256,
                                 num_trees = 100, nproc = 1)

# Fit the model with the train data and without ids and labels.
m.iforest$fit(train.normal[,-c(1:2)])
```

ISOLATION FOREST

script: isolation_forest_fish.R

```r
# Predict anomaly scores on train set.
train.scores <- m.iforest$predict(train.normal[,-c(1:2)])
# Sort and display instances with the highest anomaly scores.
head(train.scores[order(anomaly_score, decreasing = TRUE)])
```

Output:

```
 id average_depth anomaly_score
1:  75         4.08     0.7587771
2: 618          4.56     0.7345308
3: 147          4.63     0.7310602
4: 661          4.81     0.7222110
5: 756          4.81     0.7222110
6:  54         5.40     0.6939492
```

ISOLATION FOREST

script: isolation_forest_fish.R

```
threshold <- 0.76
# Predict anomaly scores on test set.
test.scores <- m.iforest$predict(test.all[,-c(1:2)])
```

Output:

```
          Reference
Prediction   0   1
         0 218  42
         1   0  12

Sensitivity (12/54)
  0.2222222
```

ISOLATION FOREST

script: isolation_forest_fish.R

```
threshold <- 0.6
```

Output:

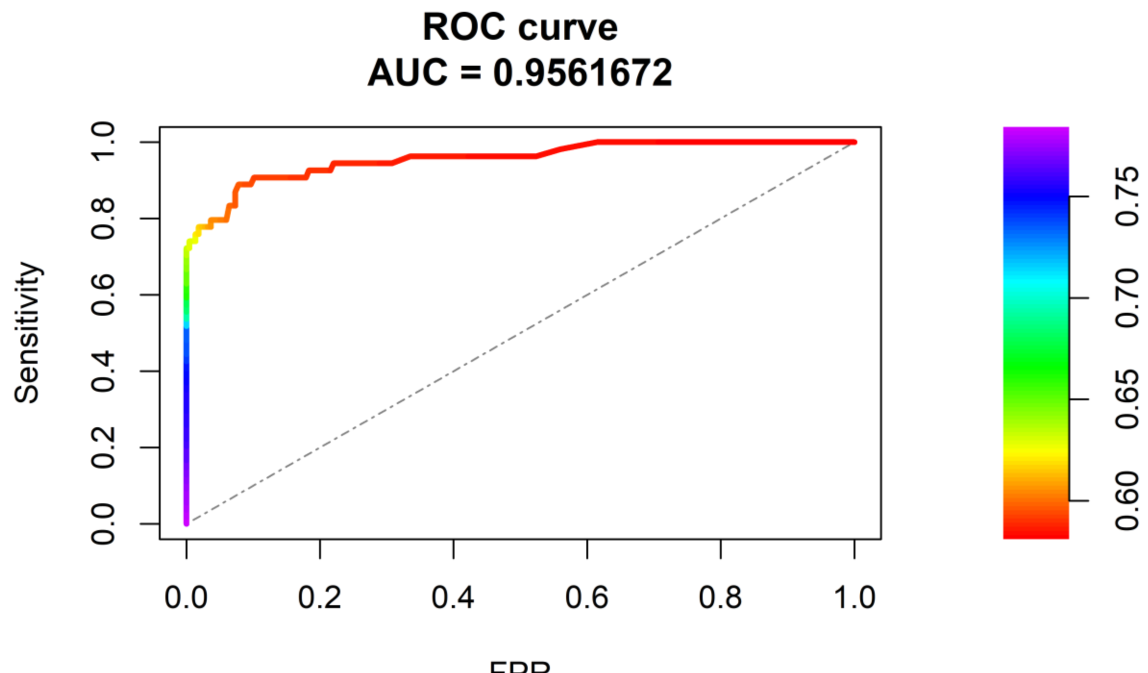
```
            Reference
Prediction    0    1
         0  204    9
         1   14   45

Sensitivity (45/54)
  0.8333333
```

ISOLATION FOREST

script: isolation_forest_fish.R

We can visualize how the false positive rate and sensitivity varies with different thresholds ROC curve.

```r
library(PRROC)

roc_obj <- roc.curve(scores.class0 = test.scores$anomaly_score,
                     weights.class0 = gt.all, curve = TRUE, rand.compute = T)

plot(roc_obj, rand.plot = T)
```



SUMMARY

- Exploratory analysis of fish trajectories
- Plot trajectories
- Feature extraction
- Train an Isolation forest
- Evaluate performance

THANK YOU!

Enrique Garcia Ceja (enriquegc.com)

Tecnologico de Monterrey University, Mexico.