Random Forest classification anomaly detection: Supervised anomaly detection with a Random Forest classifier trained on labeled events; predicted probabilities above a threshold are flagged.

This tutorial shows supervised anomaly detection with a Random Forest classifier on a labeled train/test split.

Steps: - Load and visualize the dataset - Normalize, train RF, evaluate on train and test - Plot detections and residual magnitudes

# Install Harbinger (if needed)  
#install.packages("harbinger")

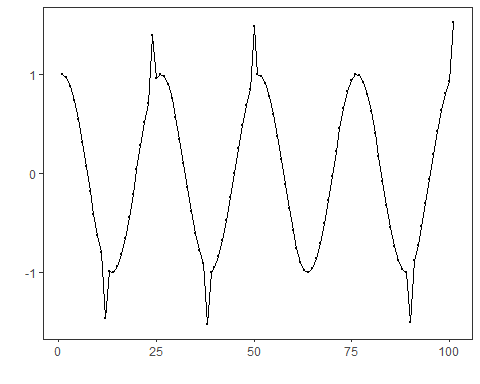
# Load required packages  
library(daltoolbox)  
library(harbinger)

# Load example anomaly datasets  
data(examples\_anomalies)

# Select the train/test dataset  
dataset <- examples\_anomalies$tt  
  
head(dataset)

## serie event  
## 1 1.0000000 FALSE  
## 2 0.9689124 FALSE  
## 3 0.8775826 FALSE  
## 4 0.7316889 FALSE  
## 5 0.5403023 FALSE  
## 6 0.3153224 FALSE

# Plot the raw time series  
har\_plot(harbinger(), dataset$serie)



# Split into train/test and normalize features  
train <- dataset[1:80,]  
test <- dataset[-(1:80),]  
  
norm <- minmax()  
norm <- fit(norm, train)  
train\_n <- transform(norm, train)  
summary(train\_n)

## serie event   
## Min. :0.0000 Mode :logical   
## 1st Qu.:0.2859 FALSE:76   
## Median :0.5348 TRUE :4   
## Mean :0.5221   
## 3rd Qu.:0.7587   
## Max. :1.0000

# Configure RF classifier  
model <- hanc\_ml(cla\_rf("event", c("FALSE", "TRUE"), mtry = 1, ntree = 5))

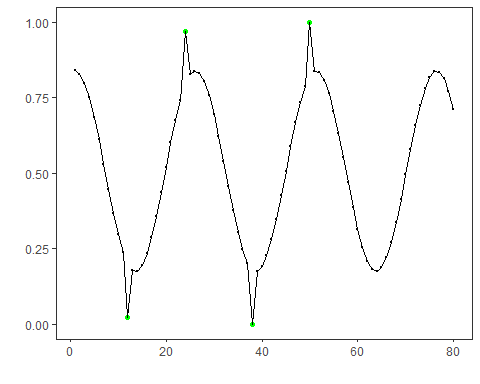
# Fit on training data and evaluate on train  
model <- fit(model, train\_n)  
detection <- detect(model, train\_n)  
print(detection |> dplyr::filter(event == TRUE))

## idx event type  
## 1 12 TRUE anomaly  
## 2 24 TRUE anomaly  
## 3 38 TRUE anomaly  
## 4 50 TRUE anomaly

evaluation <- evaluate(model, detection$event, as.logical(train\_n$event))  
print(evaluation$confMatrix)

## event   
## detection TRUE FALSE  
## TRUE 4 0   
## FALSE 0 76

# Plot training detections  
har\_plot(model, train\_n$serie, detection, as.logical(train\_n$event))



# Prepare normalized test set  
test\_n <- transform(norm, test)

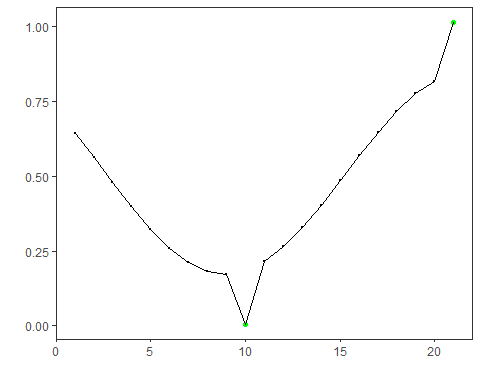
# Detect and evaluate on test  
detection <- detect(model, test\_n)  
print(detection |> dplyr::filter(event == TRUE))

## idx event type  
## 1 10 TRUE anomaly  
## 2 21 TRUE anomaly

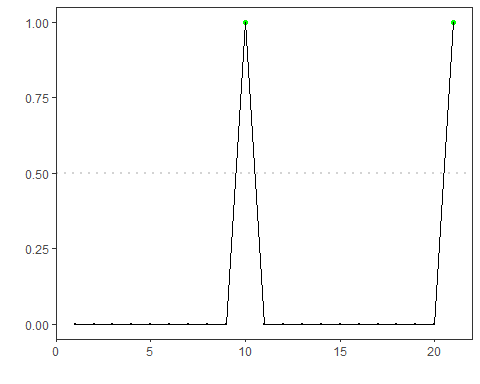
evaluation <- evaluate(model, detection$event, as.logical(test\_n$event))  
print(evaluation$confMatrix)

## event   
## detection TRUE FALSE  
## TRUE 2 0   
## FALSE 0 19

# Plot test detections  
har\_plot(model, test\_n$serie, detection, as.logical(test\_n$event))



# Plot residual magnitude and decision threshold  
har\_plot(model, attr(detection, "res"), detection, test\_n$event, yline = attr(detection, "threshold"))



References - Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.