Stream mining. Lab 3: Drift detection and triggering model updates

Maciej Grzenda, PhD, DSc
Maciej.Grzenda@pw.edu.pl

Faculty of Mathematics and Information Science
Warsaw University of Technology, Poland

Warszawa, 2025

1. Introduction
2. Investigating different drift detectors
3. Detecting drifts to trigger model updates
4. Preliminary discussion of project ideas

Introduction

Lab objectives

The objective of this lab is to gain hands-on experience in the use of different drift detectors.

We will use both synthetic data streams and real data streams

We will look for the way drift detection can be used to trigger model updates

This will rely mostly on `river`, i.e. Python-based library

While we are going to use the `river` library, please note that further details on many stream mining algorithms implemented in the library can be found in [1].

Investigating different drift detectors

Task 1: drift detection with different drift detectors

Task objectives

What happens when the distribution of our data changes (or does not)? Will the drift detector detect it? If so, will there be a difference in drift detection by different detectors?

To investigate such a setting:

1. Please implement a Python script using `river` library to use ADWIN, but also KSWIN, HDDM_A and HDDM_W for detecting changes in real or synthetic data provided by you
2. Next, please use one of the numeric features from the data set you like, ideally based on a time series. As an example, you may use the London travel mode choice data set.
3. Investigate different settings of the detectors above to observe their impact on drift detection(s)

Are all drift detectors applicable to all input streams?

Task 2: drift detection in multiple features

Task objectives

What happens when we separately monitor changes in the distribution of different features? How many detections will we get? Which features exhibit drift most frequently?

To investigate such a setting:

1. Please implement a Python script using `river` library to use one of drift detectors e.g. ADWIN for detecting changes in real multivariate data provided by you
2. As an example, you may use the London travel mode choice data set.
3. Investigate different detector settings to observe their impact on drift detection

Do we need one instance of a drift detector or many instances in this setting?

Detecting drifts to trigger model updates

Task 3: triggering immediate model updates

Task objectives

Can we use drift detection in errors made by, e.g. a classifier to trigger updates of the classifier? If we replace a classifier with a new one, will we get a better performance?

To investigate such a setting:

1. Please implement a Python script to use ADWIN detector for detecting changes in errors made by a model
2. apply a method such as Naive Bayes (NB) to non-stationary hyperplane data and display drift detections
3. when a drift in the stream of model errors is detected, initialise a new model and replace the old model with the new model
4. experiment with other than NB methods

What happened with the performance of a model?

Task 4: triggering model updates relying on shadow models

Task objectives

Is it reasonable to replace a model, e.g. a classifier, with a newly initialised one and use it immediately for predictions?

To answer this question, please modify the script from the previous task to produce predictions with both:

- the scenario of immediate use of a new model developed after drift detection for predictions, i.e. the one developed in the previous task
- the scenario in which
  - when a drift warning is received from a detector, we initialise the new *shadow model* and update it with new instances
  - when a drift confirmation is received from the detector, we replace the model with the already pre-trained *shadow model*

Are the predictions from the immediate replacement and shadow model scenarios the same? Which drift detector is well-suited for this setting?

Preliminary discussion of project ideas

References I

[1] Albert Bifet, Ricard Gavaldà, Geoff Holmes, and Bernhard Pfahringer. *Machine Learning for Data Streams with Practical Examples in MOA*. MIT Press, 2018. https://moa.cms.waikato.ac.nz/book/.