Data Stream Mining- Lecture 3

Concept Drift

Chandresh Kumar Maurya , Assistant Professor

Eötvös Loránd University, Budapest, Hungary

October 8, 2019

Introduction

In data mining, machine learning, we trying to find function f that maps input x to output y.

$$y = f(x) \tag{1}$$

Such a function is assumed to be *stationary*, i.e., distribution generating the data is fixed (but unknown). But, real-life data is

- Non-stationary
- Evolving

Need methods which can detect and adapt to changes in the underlying function. Underlying function is called *concept*. Also known as: **Change detection, co-variate shift, dataset shift etc.** Note: cf. Noise vs concept drift

An example

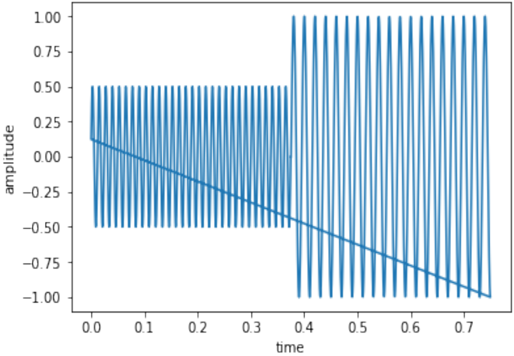


Figure: Concept drift in signals

Causes of Concept drift

A drift can occur in many ways. Let us look at the Bayesian decision formula used in naive Bayes.

$$p(c/D) = \frac{P(D/c)p(c)}{p(D)}$$

$$posterior \propto likelihood \times prior$$

Where D and c denote the data and class labels. A change can then be attributed due to:

- Class prior $p(c)$ may change over time
- likelihood p(D/c) might change
- posterior $p(c/D)$ can change.

Three causes of concept drift

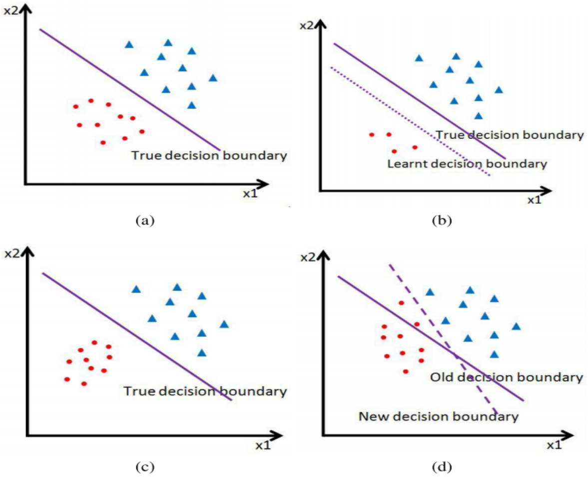


Fig. 2. Illustration of three concept drift types. (a) Original distribution. (b) $P(y)$ drift. (c) $p(\mathbf{x}|y)$ drift. (d) $P(y|\mathbf{x})$ drift.

Figure: Causes of concept drift [Wang et al., 2018]

Types of concept drifts

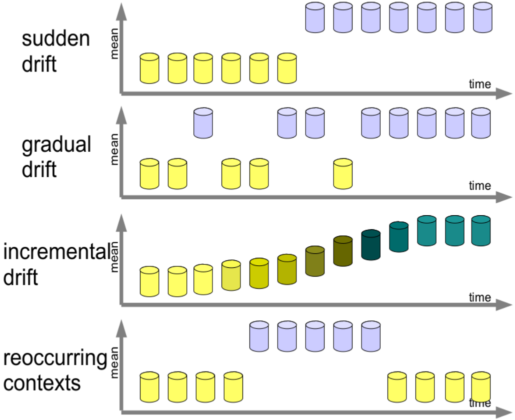


Figure: Types of concept drifts

An example could be changing gear.

Handling concept drift

Concept drift can be handled in many ways. Mostly, they are characterized in the following ways:

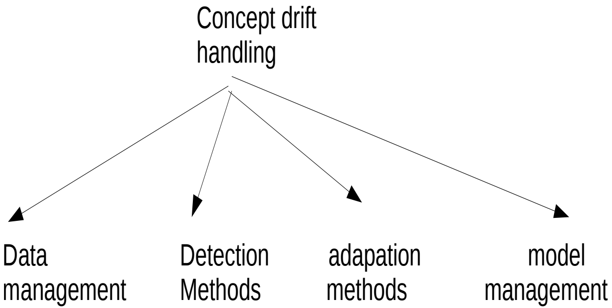


Figure: Methods of handling concept drift

Data Management

These technique characterize the data stored in memory for handling concept drift. They can be further classified as :

- Full Memory- maintain sufficient statistics via weighting examples and learn model on that.
- Partial Memory-uses recent examples to adapt the learner
  1. Fixed window (aka gradual forgetting)
  2. Adaptive window (abrupt forgetting)

Detection Methods

These methods characterize the techniques and mechanism of drift detection. Advantages:

- Can identify the location of the change
- Quantification of change

Two approaches:

1. Monitoring evolution of performance measures over time
2. Monitoring distributions over 2 different time-windows-A reference window and most recent example window

An example of the first approach is by Joachims et al. [Klinkenberg and Joachims, 2000] and second approach is given in [Kifer et al., 2004]

Example of detection method: CUSUM Algorithm

CUSUM (CUmulative SUM) algorithm is a change detection algorithm. It monitors the cumulative sum of log-likelihood ratio to detect a change.

- Consider a sequence of independent random variables x_t with pdf $p_\theta(x)$.
- Our parameter is θ which before change is θ_0 and after change is θ_1.
- Assume θ_0 is known.
- The log-likelihood ratio is defined by

$$s_t = \log \frac{p_{\theta_1}(x)}{p_{\theta_0}(x)} \tag{2}$$

Example of detection method: CUSUM Algorithm

Formally, let S_t be the current cumsum of log-likelihood ratio and m_t the current min value of S_t, the CUSUM compares this difference with a threshold.

$$g_t = S_t - m_t \geq \delta$$

where

$$S_t = \sum_{i=0}^{t} s_i$$

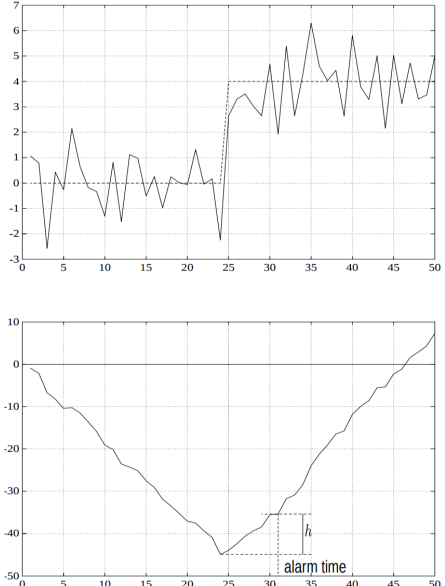
$$s_i = \log \frac{p_{\theta_1}(x)}{p_{\theta_0}(x)}$$

$$m_t = \min_{0 \leq j \leq t} S_j$$

So the change point is:

$$t_a = \min\{t : S_t \geq m_t + \delta\}$$

Typical behavior of the log-likelihood ratio S_k corresponding to a change in the mean of a Gaussian sequence with constant variance.



Contd...

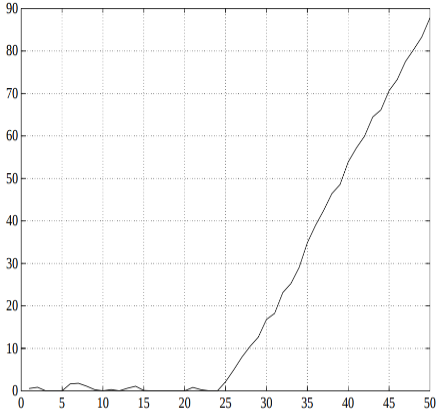


Figure: Behavior of CUSUM g_t

Adaptive methods

These methods characterise the adeptness of the decision model. Two approaches:

- Blind methods:Change decision model at regular intervals irrespective of whether a change has occurred or not. Examples includes weighting and time windows of fixed size.
- Informed methods: It includes methods which alter the model only after a change has been detected.

Decision model management

These techniques characterizes the number of decision model that needs to be kept in-memory. Key assumption:

Data comes from multiple distributions

Intuitively, we maintain a separate model each time a change is detected. What happens when the number of changes is high? Dynamic weighted majority (DWM) algorithm.

DWM

Key Idea: learn a separate model each time a change is detected and keep doing this until memory limit is hit. After that, delete a model based on its performance on the unseen data. In short, you dynamically create and delete model based on their performance. Decision about a new test point is given by majority voting.

Exercise: DWM paper reading and implementation [Kolter and Maloof, 2007]

Illustrative examples: Drift detection Statistical Process Control Algorithm (SPC)

- Performance of drift detection models is evaluated by error rate.
- Error is binomial r.v. with probability p_i and s.d. $s_i = \sqrt{p_i(1-p_i)/i}$.
- Binomial r.v. can be approximated by normal distribution since streams are infinite.
- $(1-\alpha/2)$ confidence interval for p for large enough samples can be approximated by $p_i \pm z * s_i$.
- Thus, SPC maintain two variables p_i and s_i and compares them with $p_{\min}$ and $s_{\min}$.

Contd...

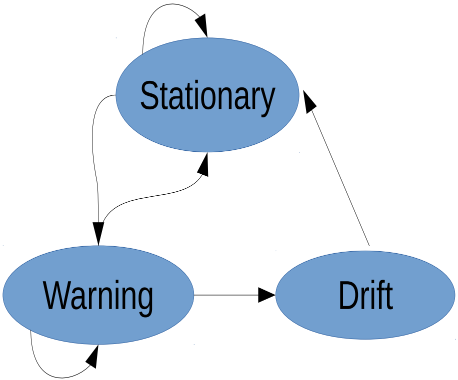


Figure: State transition diagram during concept drift

Contd...

Algorithm 8: The SPC Algorithm

input: Φ: Current decision model
Sequence of examples: $\{\vec{x}_j, y_j\}^n$

begin
  Let $\vec{x}_j, y_j$ be the current example;
  Let $\hat{y}_j \leftarrow \Phi(\vec{x}_j)$;
  Let $error_j \leftarrow L(\hat{y}_j, y_j)$;
  Compute error's mean p_j and variance s_j;
  if $p_j + s_j < p_{min} + s_{min}$ **then**
    $p_{min} \leftarrow p_j$;
    $s_{min} \leftarrow s_j$;
  if $p_j + s_j < p_{min} + \beta \times s_{min}$ **then**
    `/* In-Control */`
    $Warning? \leftarrow False$;
    Update the current decision model with the example $\vec{x}_j, y_j$;
  else
    if $p_j + s_j < p_{min} + \alpha \times s_{min}$ **then**
      `/* Warning Zone */`
      if *NOT Warning?* **then**
        $buffer \leftarrow \{\vec{x}_j, y_j\}$;
        $Warning? \leftarrow TRUE$;
      else
        $buffer \leftarrow buffer \cup \{\vec{x}_j, y_j\}$;
    else
      `/* Out-Control */`
      Re-learn a new decision model using the examples in the buffer;
      $Warning? \leftarrow False$;
      Re-start p_{min} and s_{min};

Contd...

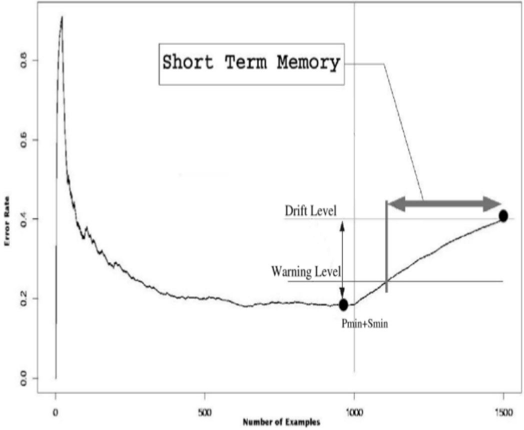


Figure: Dynamically constructed time-window. The vertical line marks the change of concept.

Implementation of SPC

Exercise: implement SPC algorithm and plot error rate (no. of errors divided by no of examples processed)

- SEA
- Intrusion detection
- spam detection
- Usenet

dataset link:
http://www.liaad.up.pt/kdus/products/datasets-for-concept-drift

Evaluation of drift detection methods

Drift detection methods are evaluated on the following metrics:

- **Error rate** (Number of mistakes made so far)
- **Probability of true detection or TPR**
- **Probability of false alarm or FPR**
- **Delay in detection**
- **precision/recall/AUC etc.**

Bibliography I

Kifer, D., Ben-David, S., and Gehrke, J. (2004).
Detecting change in data streams.
In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB '04, pages 180–191. VLDB Endowment.

Klinkenberg, R. and Joachims, T. (2000).
Detecting concept drift with support vector machines.
In *In Proceedings of the Seventeenth International Conference on Machine Learning (ICML*, pages 487–494. Morgan Kaufmann.

Kolter, J. Z. and Maloof, M. A. (2007).
Dynamic weighted majority: An ensemble method for drifting concepts.
J. Mach. Learn. Res., 8:2755–2790.

Wang, S., Minku, L. L., and Yao, X. (2018).
A systematic study of online class imbalance learning with concept drift.
IEEE transactions on neural networks and learning systems, 29(10):4802–4821.