Early classification of spatio-temporal events using partial information

Sevvandi Kandanaarachchi[1]
and
Rob J Hyndman[1]
and
Kate Smith-Miles[2]

[1]Faculty of Business and Economics, Monash University, Clayton VIC 3800, Australia.
[2]School of Mathematics and Statistics, The University of Melbourne, Parkville, VIC 3010, Australia

7 March 2019

Abstract

This paper investigates early event classification in spatio-temporal data streams, where events need to be classified using partial information, i.e. while the event is still ongoing. The framework incorporates two early event classification algorithms with different strengths as well as an event extraction algorithm. We apply this framework to synthetic and real world problems and demonstrate its reliability and broad applicability. The algorithms and data are available in the R package *eventstream*, and other code in the supplementary material.

1 Introduction

Early detection and classification of emerging events in data streams is an important challenge in our data-rich world. Data streams may arise from many different applications including social media, Internet of Things, video surveillance, epidemiology and wireless sensors, to name a few. In each of these diverse applications, there are typically events that occur and are of interest because of their disruptive behaviour to the system.

In particular, we are interested in events that start, develop for some time, and stop at a certain time. Such events can be characterised by measurable properties or features, including the “age” of the event. It is a challenge to classify these events while they are still developing because only partial information is available at this stage. Once the events have stopped developing – when the events are finished, it is easier to classify them as the complete event features are now available. For example, it is easier to differentiate a daffodil from a tulip when both are in full bloom, but more difficult to differentiate a daffodil bud from a tulip bud without resorting to other information such as characteristics of leaves. Another example is identifying a network intrusion attack in its early stages. While it may be easier to identify a breach after it has happened, it is more difficult to identify which bits of network traffic is causing the breach while it is happening (Suthaharan 2014).

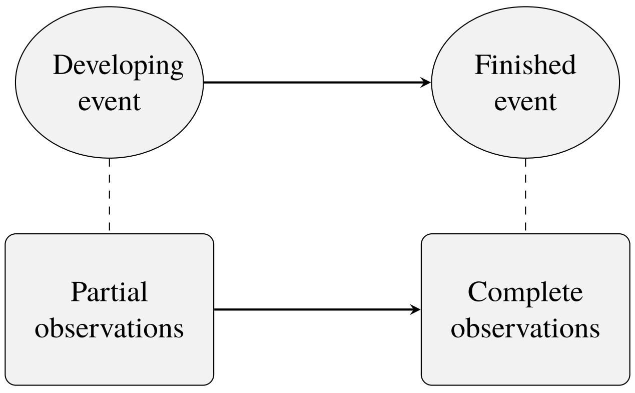


Figure 1: *Event states and partial observations.*

In this regard, we can think of these events as having two states; developing and finished (Figure 1). The partial information contained in the developing events give rise to partial or premature observations, while the finished events give rise to complete observations. As the event develops, it gives rise to a series of partial observations – each partial observation encapsulating more information than its predecessor – culminating with the complete observation (Figure 2). Thus partial observations vary with the age of the event: the difference between the current time and the start time of the event . If early classification is important, one needs to take partial observations into account in the classification process. While event detection in data streams has received much attention from different disciplines ranging from video surveillance to social media (Adam et al. 2008, Ke et al. 2005, Medioni et al. 2001, Weng & Lee 2011, Li et al. 2012, Abdelhaq et al. 2013, Allan et al. 1998, Yin et al. 2009, Mao et al. 2015), they have not been explored for developing/premature event classification to the best of our knowledge.

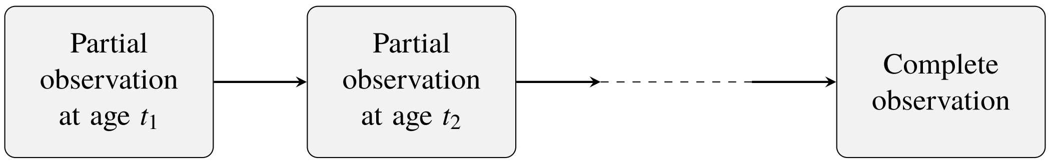

Partial observation at age t_1
Partial observation at age t_2
Complete observation

Figure 2: *Partial observations growing with event-age.*

A general framework for event classification in data streams comprises different stages : 1. data pre-processing 2. event extraction 3. feature computation and 4. event classification . This framework, augmented with partial observations structure gives the additional functionality of early event classification as depicted in Figure 3. In our framework we do not consider data pre-processing as a separate stage as this is highly dependent on the application.

Fundamentally, early event classification can be tackled by embedding age-varying coefficients in a learned model (Hastie & Tibshirani 1993). A linear model with age-varying coefficients is given by

$$y_t = a_0(t) + a_1(t)x_1(t) + \cdots + a_b(t)x_b(t) + \varepsilon_t\,, \tag{1}$$

where y_t is the output at age t, $a_i(t)$ are the age-varying coefficients, and $x_i(t)$ are the attributes of the event at age t, i.e. the partial/premature observation. A logistic model with age-varying coefficients is given by Equations (1) and (2):

$$z_t = e^{y_t}/(1 + e^{y_t}), \tag{2}$$

where z_t is the probability of the event being of a given class. As an event develops, the features $x_i(t)$ change with the age of the event, while keeping the class label constant. Thus, it is clear that the coefficients $a_i(t)$ need to change with the age of the event.

At this point, we note that concept drift (Widmer & Kubat 1996, Tsymbal 2004, Klinkenberg & Joachims 2000, Gama et al. 2014) or non-stationarity of data streams (Hulten et al. 2001, Gama 2010, Gaber et al. 2005) is different from age-varying events. For non-stationary data streams the distribution of data changes with time. For example consider a fixed part of a river, which is monitored for fluctuations in water volume and animals in the river. In months of heavy rains the water volume increases changing the distribution compared to previous months. This is an example of non-stationarity. In contrast age-varying events are about the extracted events and not the data-stream. To continue with the same example consider a log appearing on this portion of the river. When the log comes closer and the image becomes clearer, suppose it becomes apparent that it is not a log, but a crocodile. This is an example of an age-varying event. Clearly, from the time when the log appeared to the time when it was detected that it was a crocodile, no significant changes in water volume or the animal distributions took place. The volume of water and the average number of crocodiles in the river does not need to change when the perception of the log changed to that of a crocodile as a result of more information. Thus age-varying events comprise change within the event as a result of maturing partial observations, while non-stationarity is on change within the data stream.

1.1 Real world example

We turn to a real application where age-varying events occur. Figure 4a shows the heatmap of a dataset produced from a fibre optic cable. A pulse is periodically sent through the cable and this results in a data matrix where each horizontal row gives the strength of the signal at a fixed location x_0, and each vertical column gives the strength of the signal along the cable at a fixed time t_0. In this dataset the yellow parts represent high intensity values and the blue parts represent low intensity values.

Fibre optic sensor cables are used in many applications including optical communications, detecting undersea cable faults (Jiang & Sui 2009), detecting oil leakages (Niklès et al. 2004), detecting intruders on secured

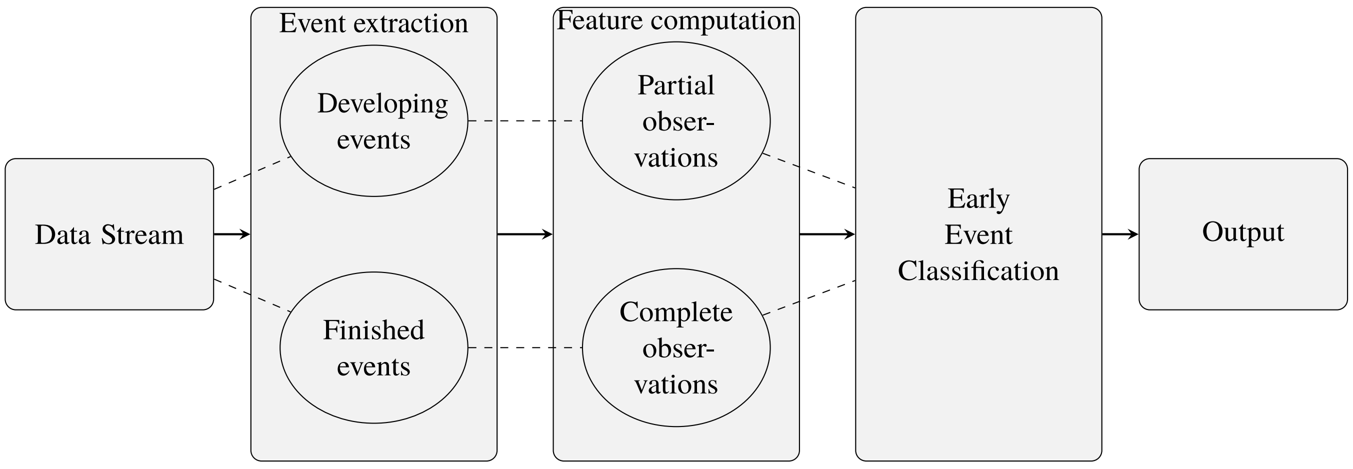
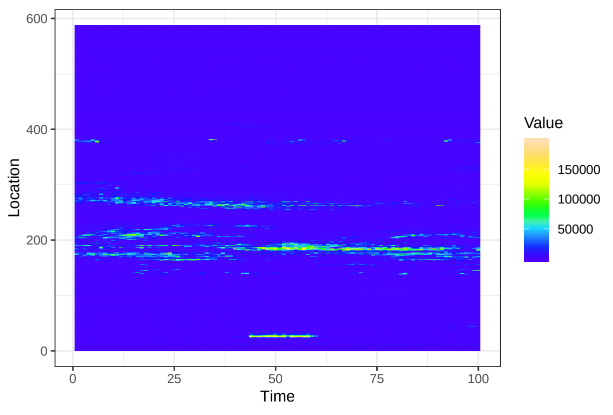
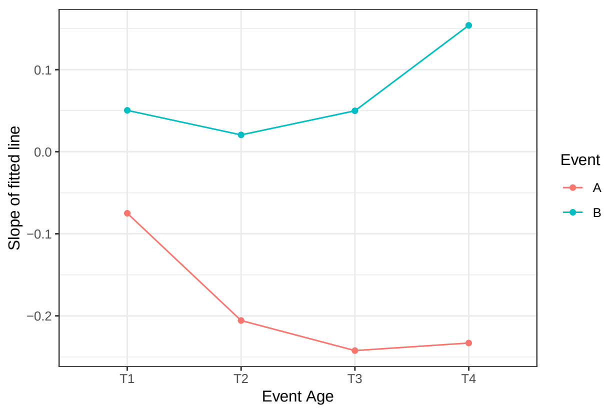


Figure 3: *Framework for event extraction and classification for spatio-temporal data.*



(a)



(b)

Figure 4: *Figure 4a shows data from a fibre optic cable. We extract events from this window and compute event features. We consider two events belonging to two different classes and an event feature that changes with event-age. Figure 4b shows this event feature, which is the slope of a line fitted to the event signal values, and how it changes with event-age.*

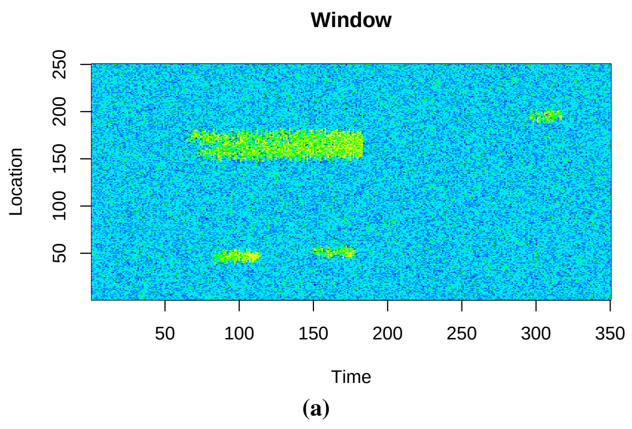
premises (Griffiths 1995), monitoring health of infra-structure such as bridges and pipe-lines (Li et al. 2004), to name a few. Events in these applications can often be grouped into two classes. For example, a cable lying on the sea bed can produce spatio-temporal events that are either cable faults (A), or non-fault events due to the activity in the ocean (B). Due to its sensitivity, fibre optic cables are also prone to noise. In a setting where early classification is important, we need to classify these events quickly, preferably while they are still ongoing.

In the dataset in Figure 4a, events are seen in the lighter-coloured parts. The event approximately at location 30 between the time interval 45 to 60 is of class A while other events that appear between locations 150 and 400 are of class B. Figure 4b shows an event feature, which is the slope of a fitted line to the event signal values, computed on two events belonging to each class. As the event matures, we see this feature change with event-age. Due to the commercially sensitive nature of the dataset, we refrain from giving details about the actual application.

1.2 Contributions

We propose the framework depicted in Figure 3 for early event classification in spatio-temporal data streams using the partial observation structure. Specifically, our contributions in this paper are:

1. We introduce two algorithms for early event classification - a static algorithm suited for stable, stationary distributions and a dynamic algorithm for non-stationary data distributions. This is our main contribution.
   (a) The static algorithm is based on multiple layers of base classifiers, which are bound together by penalty terms to work as a single classifier. This model works on partial observations and classifies developing events. In our implementation, we use logistic regression as the base classifier. However, other base classifiers can be used. We call this model Static Age-Varying Events Classifier (SAVEC).
   (b) The dynamic algorithm is based on a series of non-Gaussian state space models, which works on partial observations. Due to the functionality of the Kalman filter, the model can update easily, which is advantageous for non-stationary data distributions We call this model Dynamic Age-Varying Events Classifier (DAVEC).
2. We introduce a simple algorithm for event extraction from spatio-temporal data, as events need to be extracted before the classification process.
3. We demonstrate the validity of these algorithms on synthetic and real world data.
4. We make the algorithms and data available in the R package *eventstream* (Kandanaarachchi 2018).



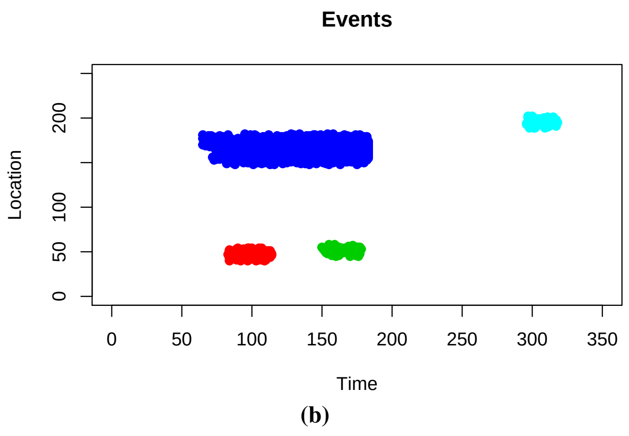


Figure 5: *A synthetic dataset from eventstream and the events extracted from it using* $\alpha = 0.95$, $\epsilon = 5$ *and* $minPts = 10$.

The remainder of the paper is organised as follows. Section 2 discusses the preliminaries. We introduce our event extraction algorithm and discuss the event features. We also introduce the notation for partial observations, which is essential for classifying age-varying events. In Sections 3 and 4 we introduce the static and dynamic age-varying events classifiers, SAVEC and DAVEC respectively. We discuss the implementation of age-varying coefficients in SAVEC and DAVEC and how it gives a growing prediction on developing events. In Section 5 we discuss results of our framework using synthetic data and two real applications. The first application focuses on early event classification and is associated with the dataset in Figure 4. The second application demonstrates the relevance of our event extraction algorithm to Nitrogen dioxide (NO_2) data from NASA's NEO (*NASA Earth Observations* 2004) website. Finally, in Section 6, we present our conclusions and discuss future work.

2 Events, features and partial observations

2.1 Event extraction

We extract events from data streams of 2 or 3 dimensions; i.e. 1 time dimension and either 1 or 2 spatial dimensions. We employ a simple method for event extraction using DBSCAN (Ester et al. 1996), which is a density based clustering algorithm. Our event extraction algorithm is explained in Algorithm 1.

```
input  : a 2 or 3-dimensional array $A_{n\times m}$ or $A_{n\times m\times k}$ , and parameters $\alpha$, $\epsilon$ and $minPts$.
output : events $A|_S \subset A$ and the event ids for each $s_{ij} \in A|_S$ (or $s_{ijk}$ if $A$ is 3-dimensional).
1 Let $a_{ij}$ be the signal value at $(i, j)$ position of $A$, if $A$ is 2-dimensional and $a_{ijk}$ be the signal value at $(i, j, k)$ position of $A$ if $A$ is 3–dimensional.
2 Let $q$ denote the $\alpha$-percentile of the signal values of $A$.
3 $S = \{(i, j) \mid a_{ij} > q\}$ for 2-dimensional $A$ and $S = \{(i, j, k) \mid a_{ijk} > q\}$ for 3-dimensional $A$.
4 $B = A|_S$, i.e., signal values of $A$ in $S$ locations.
5 Using DBSCAN cluster B using $\epsilon$ and $minPts$.
6 This clustering gives each $s \in A|_S$ a cluster id.
7 Consider each cluster as an event.
```

Algorithm 1: *Extract events from a dataset or window.*

Figure 5 shows a synthetic dataset generated from the package *eventstream*, along with the events extracted using Algorithm 1.

2.2 Features

As we work with a data stream, we use a moving window model in our experiments. We extract events from data in the current window and compute features for these events. The feature set comprises some basic features such as length and width of each event, and some other features that compute the intensity of each event relative to the background. The "Relative to the background" features are equivalent to a family of signal to noise ratio features and are motivated from the fibre optic application (see Figure 4a).

To compute the SNR family of features we use smoothing splines and thus they are only computed for two-dimensional data streams due to ease of computation. Using a small portion from the beginning of each window, which correspond to the recent past, we compute the mean, median, IQR and standard deviation for each location. Using these values at each location, we compute four smoothing splines. The objective is to have the background mean, median, IQR and standard deviation pixel value for each location. The median and IQR splines from a small window in Figure 6a are shown in Figures 6b and 6c.

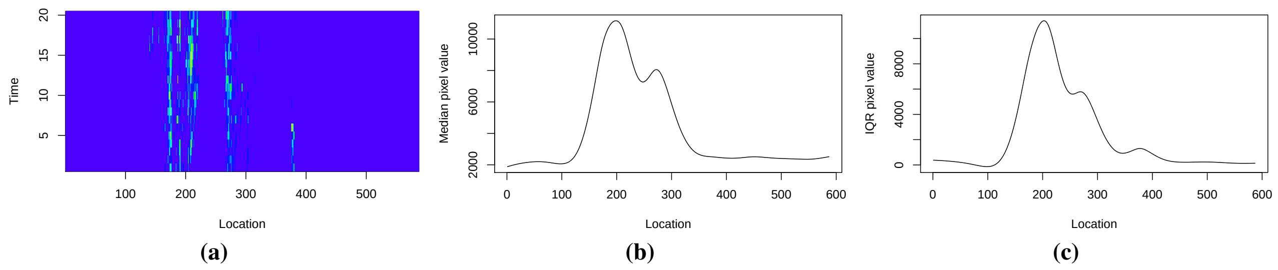


Figure 6: *The initial portion of a window and the resulting median and IQR splines.*

For two-dimensional events we compute the following features:

1. Number of cells/pixels in event
2. Length of event
3. Width of event
4. Length to width ratio of event
5. Centroid
   The centroid is used to compute other features which are relative to the event. It is not used in event classification.
6. Sum of signal-values of cells in event
7. Mean signal-value of event
8. Standard deviation of signal-values of event
9. Slope of the fitted line ζ
   The average signal value at each time of the event is computed and a line ζ is fitted to the average values. The slope of the fitted line ζ is a feature of interest .
10. Linear and quadratic coefficients of a fitted parabola p
    The average signal value at each time of the event is computed and a parabola p is fitted to the average values. The linear and quadratic coefficients of the fitted parabola p are features of interest.
11. n standard deviations from the mean
    The proportion of event cells/pixels that has signal-values greater than n global standard deviations from the global mean for $n \in \{2, 3, 4\}$.
12. n local IQR from local median
    The value of the median smoothing spline at each event centroid is used as the local median for that event. Similarly, the value of the IQR smoothing spline at each event centroid is used as the local IQR for that event. This feature gives the proportion of event pixels/cells that has signal-values greater than n local IQRs from the local median for $n \in \{5, \ldots, 8\}$
13. Local IQRs from local median
    Let us denote the 75th percentile of the event signal value by x. This feature gives the number of local IQRs for which x is greater than the local median. Both local IQR and local median are computed using splines described above.
14. Local standard deviation from local mean
    Similar to the previous feature, our x is the 80th percentile of the event signal value. Here we compute the number of local standard deviations for which x is greater than the local mean.

For three-dimensional data streams we compute a subset of the above features. In particular, we compute features 1–8 from the above list and an equivalent of feature 14 using the global standard deviation and the global mean. These features now provide a compact way to represent a data stream and the embedded events, summarising salient properties of the time window in terms of event signal strength and shape. This summary becomes input to a classifier to identify types of events.

2.3 Partial/incomplete observations

In the classical setting, a classification problem comprises observations $(\boldsymbol{x}_i, y_i)$ for $i \in 1, \ldots, N$ where $\boldsymbol{x}_i \in \mathbb{R}^b$ is the attribute vector of the ith observation and y_i is its class label. The task of the classifier is to learn the class boundary by using the given set of observations. Then for any new observation $\boldsymbol{x}_j$, the classifier can predict its class label using the learned class boundaries. Let us call this a **standard classifier**.

Standard classifiers have been widely popular in diverse fields of study and practice. However, they are not without limitations. One of the limitations is that once a classifier is trained, it has fixed class boundaries. If the new data is different from the data learned by the classifier, the output of the classifier is of little use. This is particularly the case in data-streaming scenarios, where data distributions are non-stationary (sometimes also referred to as concept drift). It is necessary for a classifier to re-adjust its class boundaries when faced with non-stationarity. The literature on adapting or evolving classifiers is significant (Duchi et al. 2011, Frey & Slate 1991, Nishida et al. 2005, Alippi & Roveri 2008*a*,*b*). Let us call these classifiers **evolving classifiers**.

Now, consider the case when a new observation is not made available at once but gradually, where we get partial information about the new observation and the amount of partial information increases with time. This is the case for events described in Section 1.1. Let $\boldsymbol{x}_j$ be a new observation which becomes available partially via the following finite sequence of partial observations $\{\boldsymbol{p}^j_{t_1}, \boldsymbol{p}^j_{t_2}, \boldsymbol{p}^j_{t_3}, \ldots, \boldsymbol{p}^j_{t_n}\}$. Here the partial observation of $\boldsymbol{x}_j$ at age t_k is denoted by $\boldsymbol{p}^j_{t_k}$ and $\boldsymbol{p}^j_{t_n} = \boldsymbol{x}_j$ with $t_1 < t_2 < \cdots < t_n$. We differentiate between the time and the age of a partial observation. A partial observation that begins at time $t = t_1$ has age 0 at time t_1, and at time $t = t_2$ it has age $t_2 - t_1$.

We consider the question "how can we classify partial observations?" If one trains a single standard classifier on all partial observations, it may be optimal for a certain set of partial observations p_{t_k} at a given age t_k, but not all partial observations, because partial observations change with time. If one waits until the partial observation has formed into a full observation $\boldsymbol{x}_j$, then a standard classifier can be used. However, for some applications such as intrusion detection it might be too late to wait until the full observation has formed. One option is to have a series of standard classifiers $\{C_{t_i}\}_{i=1}^n$ each trained on partial observations p_{t_i}. When a new observation gradually arrives in the form of a sequence of partial observations $\{\boldsymbol{p}^k_{t_1}, \boldsymbol{p}^k_{t_2}, \boldsymbol{p}^k_{t_3}, \ldots, \boldsymbol{p}^k_{t_n}\}$, the classifier C_{t_i} can be used on $\boldsymbol{p}^k_{t_i}$. Thus, as the partial observation grows, we have a growing prediction $\{y_{t_1}, y_{t_2}, \ldots, y_{t_n}\}$ of the class label. More importantly, we do not need to wait until the partial observation matures to a full observation before making a prediction.

However, having a series of classifiers independent of each other is sub-optimal because each classifier is only trained on a portion of the data, i.e. it is trained on individual snapshots of events at different ages. By linking event snapshots of different event-ages in an appropriate way, better predictions can be achieved. The classifiers SAVEC and DAVEC described in the next sections address this limitation, while giving a growing prediction.

3 SAVEC: Static Age-Varying Events Classifier

Let the standard classifier minimize the loss function given by $\mathscr{L}$, i.e.

$$\arg\min_{\beta} \frac{1}{N} \sum_{i=1}^{N} \mathscr{L}(\boldsymbol{x}_i, y_i; \beta),$$

where $\beta = (\beta_0, \beta_1, \ldots, \beta_l)$ and $(\boldsymbol{x}_i, y_i)$ are observations for $i \in \{1, \ldots, N\}$. By extending a classifier to take in partial observations, we initially minimize the following loss function,

$$\underset{\tilde{\beta}}{\arg\min} \frac{1}{nN} \sum_{i=1}^{N} \sum_{j=1}^{n} \mathscr{L}(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}),$$

where $\tilde{\beta} = \{\tilde{\beta}_{jk}\}$ for $j \in \{1, \ldots, n\}$, $k \in \{0, 1, \ldots, l\}$ and $\boldsymbol{x}_j$ gradually becomes available as $\{\boldsymbol{p}^j_{t_1}, \boldsymbol{p}^j_{t_2}, \boldsymbol{p}^j_{t_3}, \ldots, \boldsymbol{p}^j_{t_n}\}$ with $\boldsymbol{x}_j = \boldsymbol{p}^j_{t_n}$. For a given j, y_j is the class label of $\boldsymbol{x}_j$, and so for all k, $\boldsymbol{p}^j_{t_k}$ have the same y_j. Let us call this initial extended classifier $\mathscr{E}$.

We note that $\mathscr{E}$ is equivalent to a series of n classifiers $\{C_{t_j}\}_{j=1}^n$. To show this, first note that C_{t_j} minimizes the loss function

$$\underset{\tilde{\beta}_j}{\arg\min} \frac{1}{N} \sum_{i=1}^{N} \mathscr{L}(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}_j),$$

with $\tilde{\beta}_j = (\tilde{\beta}_{j0}, \tilde{\beta}_{j1}, \tilde{\beta}_{j2}, \ldots, \tilde{\beta}_{jl})$. Also,

$$\underset{\tilde{\beta}}{\arg\min} \sum_{i=1}^{N} \sum_{j=1}^{n} \mathscr{L}(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}) = \underset{\tilde{\beta}}{\arg\min} \sum_{j=1}^{n} \sum_{i=1}^{N} \mathscr{L}(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}) . \tag{3}$$

In addition $\{\boldsymbol{p}^i_{t_j}\}_{i=1}^N$ for a fixed age t_j only affects $\tilde{\beta}_j$. Therefore the matrix $\tilde{\beta}$ can be computed row by row by minimizing $\sum_{i=1}^N \mathscr{L}\left(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}_j\right)$ for each j. Thus, we can write

$$\underset{\tilde{\beta}}{\arg\min} \sum_{j=1}^{n} \sum_{i=1}^{N} \mathscr{L}\left(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}\right) = \left[\underset{\tilde{\beta}_1}{\arg\min} \sum_{i=1}^{N} \mathscr{L}\left(\boldsymbol{p}^i_{t_1}, y_i; \tilde{\beta}_1\right) \cdots \underset{\tilde{\beta}_n}{\arg\min} \sum_{i=1}^{N} \mathscr{L}\left(\boldsymbol{p}^i_{t_n}, y_i; \tilde{\beta}_n\right) \right]^T . \tag{4}$$

From equations (3) and (4) we see that the initial extended classifier $\mathscr{E}$ is equivalent to a series of n classifiers $\{C_{t_j}\}_{j=1}^n$.

Having n independent classifiers $\left\{C_{t_j}\right\}_{j=1}^n$ will make the class boundaries for each classifier independent of each other. As $\boldsymbol{x}_i \in \mathbb{R}^b$, the class boundary of C_{t_j} lives in the ambient space $\mathbb{R}^b$. Let us denote the class boundary of C_{t_j} by B_j and the class boundary of $\mathscr{E}$ by $B_{\mathscr{E}}$. As n independent classifiers $\left\{C_{t_j}\right\}_{j=1}^n$ in totality are equivalent to $\mathscr{E}$, it follows that the class boundary $B_{\mathscr{E}}$ comprises the set $\left\{B_j\right\}_{j=1}^n$. By stacking B_j in the order of increasing j, i.e. by considering the set $\left\{\left(t_j, B_j\right)\right\}_{j=1}^n$, we can visualize a continuous class boundary that connects the set $\left\{B_j\right\}_{j=1}^n$ with increasing t_j. This continuous class boundary lives in $\mathbb{R}^{b+1}$. Thus, the set $\left\{\left(t_j, B_j\right)\right\}_{j=1}^n$ can be viewed as a discrete approximation of a continuous class boundary. Therefore, $B_{\mathscr{E}}$ can be viewed as living in $\mathbb{R}^{b+1}$.

In addition, it is desirable if $B_{\mathscr{E}}$ has a certain level of smoothness, because it aids convergence of early classification based on partial observations to full classification based on full observations. To incorporate smoothness in $B_{\mathscr{E}}$, we modify the original loss function by including an L_2 penalty term as follows:

$$\varphi\left(\tilde{\beta}, \lambda\right) = \frac{1}{nN} \sum_{i=1}^{N} \sum_{j=1}^{n} \mathscr{L}\left(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}\right) + \lambda \sum_{k=1}^{l} \sum_{j=1}^{n-1} \left(\tilde{\beta}_{j+1,k} - \tilde{\beta}_{j,k}\right)^2 , \tag{5}$$

for some $\lambda > 0$. The constant λ is a parameter that can be specified. Recall that $\tilde{\beta}_j = \left(\tilde{\beta}_{j1}, \tilde{\beta}_{j2}, \ldots, \tilde{\beta}_{jl}\right)$ relates to partial observations $\{\boldsymbol{p}^i_{t_j}\}_{i=1}^N$ for a fixed t_j, i.e. $\tilde{\beta}_{j1}$ is the coefficient of the first covariate at age t_j and $\tilde{\beta}_{j2}$ the coefficient of the second covariate at age t_j. Thus the penalty term $(\tilde{\beta}_{j+1,k} - \tilde{\beta}_{j,k})^2$ takes coefficients for the kth covariate at ages t_j and t_{j+1} and penalizes the difference, enforcing a certain smoothness in event-age. This is the loss function minimized by SAVEC (*static age-varying events classifier*). Thus SAVEC finds $\tilde{\beta}$ such that,

$$\underset{\tilde{\beta}}{\arg\min}\, \varphi\left(\tilde{\beta}, \lambda\right) = \underset{\tilde{\beta}}{\arg\min} \left(\frac{1}{nN} \sum_{i=1}^{N} \sum_{j=1}^{n} \mathscr{L}(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}) + \lambda \sum_{k=1}^{l} \sum_{j=1}^{n-1} (\tilde{\beta}_{j+1,k} - \tilde{\beta}_{j,k})^2 \right) . \tag{6}$$

As any loss function $\mathscr{L}$ can be used in equation (6), SAVEC is a general formulation. Our implementation of SAVEC in *eventstream* (Kandanaarachchi 2018) uses logistic regression as the base classifier and the associated loss function. However, SAVEC can be implemented with any loss function. For logistic regression (Friedman et al. 2001) the loss function $\mathscr{L}$ is given by

$$\mathscr{L}(\boldsymbol{p}^i_{t_j}, y_i; \tilde{\beta}) = -y_i \left([1 \ (\boldsymbol{p}^i_{t_j})^T] \tilde{\beta}_j \right) + \log \left(1 + \exp \left\{ [1 \ (\boldsymbol{p}^i_{t_j})^T] \tilde{\beta}_j \right\} \right) . \tag{7}$$

Here the vector $[1 \ (\boldsymbol{p}^i_{t_j})^T]$ denotes the concatenation of the vector $\boldsymbol{p}^i_{t_j}$ with the constant 1 to account for the intercept.

Next we explore non-Gaussian state space models for partial observations.

4 DAVEC: Dynamic Age-Varying Events Classifier

We use non-Gaussian state space models as described in (Durbin et al. 1998) and (McCormick et al. 2012) to model partial observations, and the implementation in R package *dma* (McCormick et al. 2018). We start with the linear Gaussian state space models which are given by the following equations :

$$y_t = Z_t \alpha_t + \varepsilon_t \, , \tag{8}$$

$$\alpha_{t+1} = T_t \alpha_t + R_t \eta_t \, , \tag{9}$$

where $\varepsilon_t \in \mathcal{N}(0, H_t)$ and $\eta_t \in \mathcal{N}(0, Q_t)$. Here y_t is an N-dimensional vector of observations and α_t is a $(l+1)$-dimensional state vector. The matrices Z_t, T_t, R_t, H_t and Q_t are known at the beginning. Equation (8) is called the "observation equation" and equation (9) the "state equation". When using state space models for dynamic regression, α_t denotes the coefficients which update dynamically.

If y_t is a binary response such that

$$y_t \sim \text{Bernoulli}(p_t) \, ,$$

equation (8) changes as

$$\text{logit}(p_t) = Z_t \alpha_t \, , \tag{10}$$

with logit function $f(t) = \log(t/(1-t))$, while keeping equation (9) unchanged.

4.1 Modelling partial observations with DAVEC

We are interested in a growing prediction $\{\theta^j_{t_1}, \theta^j_{t_2}, \ldots, \theta^j_{t_n}\}$ for partial observations $\{\boldsymbol{p}^j_{t_1}, \boldsymbol{p}^j_{t_2}, \boldsymbol{p}^j_{t_3}, \ldots \boldsymbol{p}^j_{t_n}\}$ belonging to the same complete observation. Again, we note that for partial observations $\{\boldsymbol{p}^j_{t_1}, \boldsymbol{p}^j_{t_2}, \boldsymbol{p}^j_{t_3}, \ldots \boldsymbol{p}^j_{t_n}\}$, the time stamps $(t_1, t_2, \ldots, t_n)$ denote the ages of the partial observations, not the time of occurrence.

As state space models generally deal with time, for each age t_k we use a separate state space model. However, for some cases it may benefit if there is communication between the models regarding the same observation. To add communication between the models, we bundle the partial observation $\boldsymbol{p}^j_{t_k}$ with the prediction output $\theta^j_{t_{k-1}}$ of the lower age model and use it as input for the age t_k model.

To explain this further, let us define $\boldsymbol{P}_{t_k} = \left[\boldsymbol{p}^1_{t_k} \ \boldsymbol{p}^2_{t_k} \ \cdots \ \boldsymbol{p}^N_{t_k} \right]^T$ - an $l \times N$ matrix containing all partial observations of age t_k, $\mathbf{1} = [1, 1, \ldots, 1]^T$ and $\theta_{t_k} = \left[\theta^1_{t_k}, \theta^2_{t_k}, \ldots, \theta^N_{t_k} \right]^T$ - the logit predictions of the partial observations in $\boldsymbol{P}_{t_k}$, where $\boldsymbol{p}^j_{t_k}$ is an l-vector and N denotes the number of complete observations. Then the equation corresponding to equation (10) for the initial model for age t_1 can be written as

$$\theta_{t_1} = Z_{t_1} \alpha_{t_1} \, . \tag{11}$$

$$\text{with} \qquad Z_{t_1} = [\mathbf{1} \ \boldsymbol{P}_{t_1}] \, , \tag{12}$$

Using the output θ_{t_1}, we write the model for age t_2 as

$$Z_{t_2} = [\mathbf{1}\ \boldsymbol{P}_{t_2}\ \theta_{t_1}], \tag{13}$$
$$\theta_{t_2} = Z_{t_2}\alpha_{t_2}\,. \tag{14}$$

Building in this way we obtain

$$Z_{t_k} = [\mathbf{1}\ \boldsymbol{P}_{t_k}\ \theta_{k_{t-1}}], \tag{15}$$
$$\theta_{t_k} = Z_{t_k}\alpha_{t_k}\,. \tag{16}$$

The equations (15), (16) and (9) describe the model for the partial observations at age t_k. Let us denote the model which describes the partial observations of age t_k by Λ_k and the predicted output using the inverse link function by $\hat{\boldsymbol{y}}_{t_k}$. Our Dynamic Age-Varying Events Classifier (DAVEC) incorporates a series of sub-models $\{\Lambda_k\}_{k=1}^{n}$ in its implementation. Figure 7 shows DAVEC with internal communications between sub-models Λ_k and Λ_{k+1}.

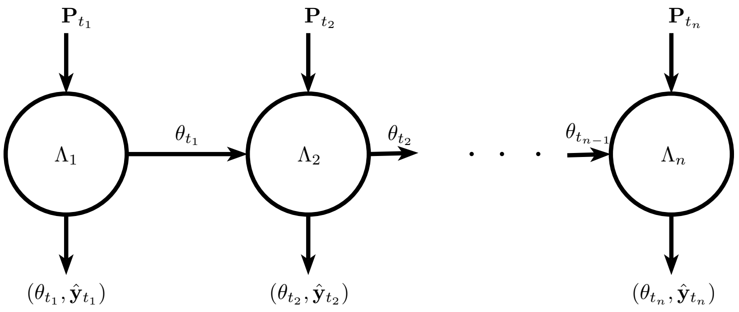


Figure 7: *Structure of DAVEC with communication.*

However, communication between models Λ_k and Λ_{k+1} may not improve the overall accuracy for every partial observations problem. For example, for a class imbalanced problem, a small number of incorrect predictions in θ_{t_k} can have an adverse effect on the model Λ_{k+1}. Therefore, for some instances such as class imbalanced datasets, it is desirable for DAVEC to have no communications between the sub-models Λ_k and Λ_{k+1} as shown in Figure 8.

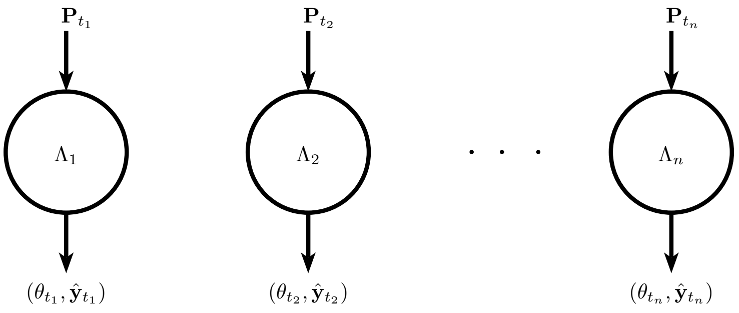

$\mathbf{P}_{t_1}$
$\mathbf{P}_{t_2}$
$\mathbf{P}_{t_n}$
Λ_1
Λ_2
Λ_n
$(\theta_{t_1}, \hat{\mathbf{y}}_{t_1})$
$(\theta_{t_2}, \hat{\mathbf{y}}_{t_2})$
$(\theta_{t_n}, \hat{\mathbf{y}}_{t_n})$

Figure 8: *Structure of DAVEC without communication.*

5 Applications

We use SAVEC and DAVEC to classify events in synthetic and real data streams. The R code applicable to this section is available in the supplementary material.

5.1 Synthetic data

The synthetic data used in this section can be generated using the R package *eventstream*. The synthetic data contains events of two classes: A and B. All events belonging to class A look similar, that is they have one single non-standard shape or visual pattern. In contrast, events belonging to class B can have one of three different non-standard shapes, including the shape of events of class A. The use of three different shapes for class B events and one shape for class A events (with that shape being similar to some of those of class B) was motivated from the fibre optic application.

Figure 9a contains two events of class A, and Figure 9b contains 3 events of class B. The shapes are labelled as 1, 2 or 3 in both Figures 9a and 9b, with shape 1 being the common shape.

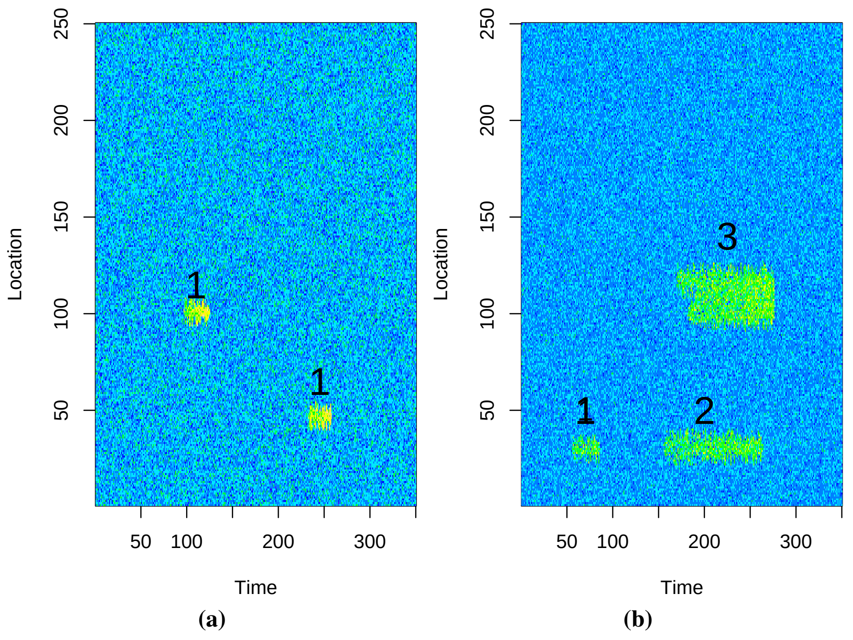


Figure 9: *Class A events in Figure 9a and Class B events in Figure 9b.*

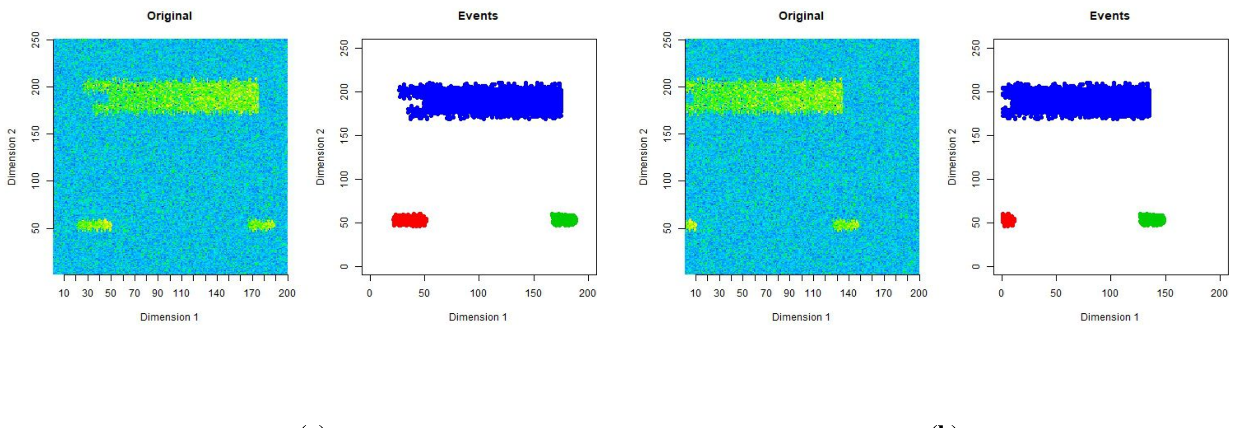
The number of events of class A and B, and their positions, are randomly generated. The other difference between the events of class A and B, apart from the shape, is that values of the pixels belonging to events of class A and B come from different probability distributions. For both classes the intensity of pixel values increase linearly with the age of the event. We list the differences between class A and B events in Table 1.

Feature	Class A value distribution	Class B value distribution
Starting cell/pixel values	$\mathcal{N}(4, 3)$	$\mathcal{N}(2, 3)$
Ending cell/pixel values	$\mathcal{N}(8, 3)$	$\mathcal{N}(4, 3)$
Maximum age of event - shape 1	$\mathcal{U}(20, 30)$	$\mathcal{U}(20, 30)$
Maximum age of event - shape 2	-	$\mathcal{U}(100, 150)$
Maximum age of event - shape 3	-	$\mathcal{U}(100, 150)$
Maximum location width of event - shape 1	$\mathcal{U}(20, 26)$	$\mathcal{U}(20, 26)$
Maximum location width of event - shape 2	-	$\mathcal{U}(30, 38)$
Maximum location width of event - shape 3	-	$\mathcal{U}(50, 58)$

Table 1: *Differences in class A and class B events.*

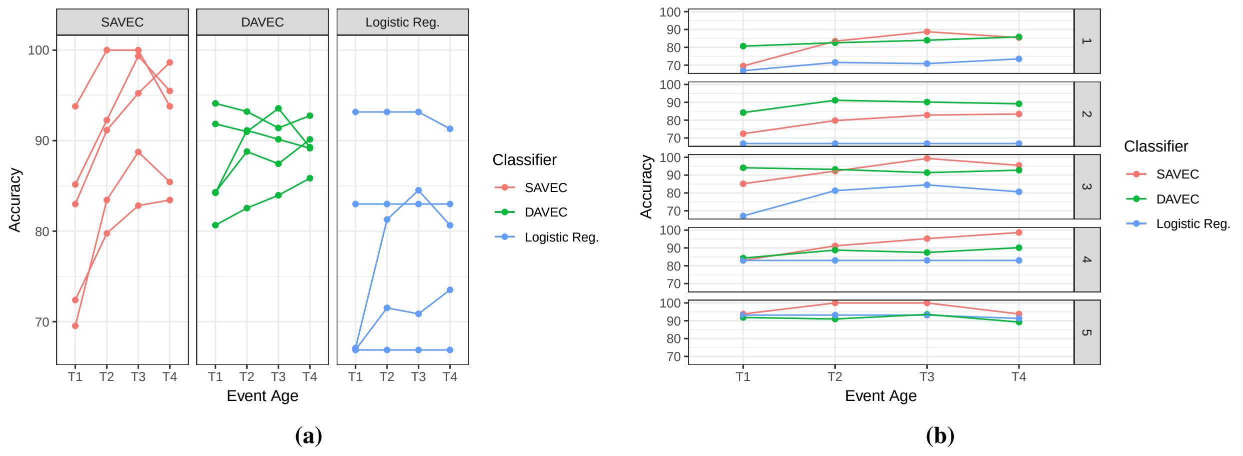
We classify the extracted events from synthetic data using SAVEC and DAVEC. As a benchmark comparison we use logistic regression. We use the same feature vector to summarise the extracted events for all three classifiers. Figure 10 shows snapshots of the data and the extracted events for two time windows.

We repeat each experiment 5 times with data streams generated with different seeds. For each experiment we generate a data stream of dimension 3500×250 of which 80% (2800×250) is used for training and the remaining 20% for testing. For synthetic data classification, we do not use features which were motivated from the real example, i.e. we do not use features 11 and 12 from the list in Section 2.2, for computational efficiency. In addition, the centroid is not used in any classification task. As class A events can have a maximum age of 30, we



(a) **(b)**

Figure 10: *Two windows of data and extracted events.*



(a) **(b)**

Figure 11: *Accuracy of the 3 classifiers over 5 repeats grouped by classifier in Figure 11a and by repetition in Figure 11b.*

use 4 event ages for the classification tasks at $t = 8, 16, 24$ and 32 time units. That is, event features are calculated at these ages. We use a moving window model of dimension 200×250 which moves by a step of 8×250.

Figure 11 shows the accuracy of SAVEC, DAVEC and the logistic regression classifier over 5 repetitions. From Figure 11b we see that for each repetition either SAVEC or DAVEC surpasses the logistic regression classifier. From Figure 11a we see that the SAVEC performs better on average than the other two classifiers for synthetic data. Table 2 gives the average test set accuracy and standard deviation results for the 3 classifiers, which confirm these observations. Also we see that all 3 classifiers improve their average accuracy levels with the age of the events.

Classifier	Accuracy				Standard deviation			
	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
SAVEC	80	89	93	91	9	7	7	6
DAVEC	87	89	89	89	5	4	3	2
Logistic regression	75	79	79	79	12	10	10	9

Table 2: *Average test set accuracy and standard deviation* (%) *over* 5 *repetitions.*

5.2 Fibre optic cable data

The data for the first real application is from a fibre optic cable, and is shown in Figure 12. The data set is available in the R package *eventstream*. Again, for commercially sensitive reasons, we cannot provide more information about the application. The data set has dimensions 379×587, with class A events labelled with letter **A**. The extracted events have a maximum age of 40 time units, so we use a window model with a window size 40×587 and a step size 10×587 to extract events and compute features.

Similar to the synthetic data stream, we classify the extracted events using SAVEC and DAVEC. Again as a benchmark we use logistic regression. The main difference compared to the synthetic data is that the real data stream has a much smaller number of class A events compared to class B events. Due to this class imbalance, we configure DAVEC for real data differently from the synthetic data model. First, we use DAVEC without internal communications, as depicted in Figure 8. Second, as there are only 4 class A events, there is not enough class A data to update DAVEC regularly. Consequently, we use DAVEC for training and prediction without the updating step given in equation (9).

We use 4-fold cross validation because there are 4 class A events, with event ages $t = 10, 20, 30$ and 40, because the maximum event-age is 40 time units. Each classifier, SAVEC, DAVEC and logistic regression is trained on a data stream comprising 3 training folds, and tested on the remaining fold.

We report additional accuracy measures that are designed for imbalanced datasets. These metrics are positive predictive value (PPV), negative predictive value (NPV) and area under the receiver operator characteristic curve (AUC). We give the definitions of these metrics below:

$$\text{Positive predictive value (PPV)} = \frac{\text{Number of true positives}}{\text{Number of predicted positives}},$$

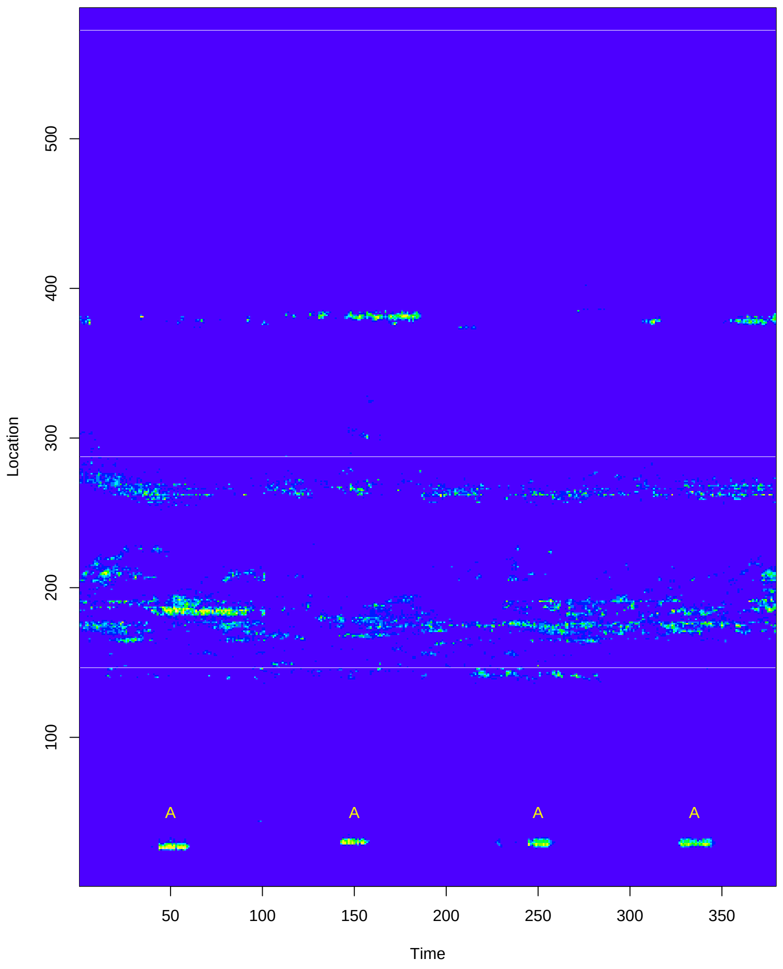


Figure 12: *Data stream from a fibre optic cable.*

$$\text{Negative predictive value (NPV)} = \frac{\text{Number of true negatives}}{\text{Number of predicted negatives}},$$

The number of predicted positives in PPV is the sum of true positives and false positives, and the number of predicted negatives in NPV is the sum of true negatives and the false negatives. Considering PPV and NPV together gives a two-sided accuracy measures. For example, a classifier that predicts all observations as negative except for one correct positive observation achieves a PPV of 100% but a small NPV. The combination of PPV and NPV gives the overall accuracy of the model.

In contrast, the area under the receiver operator characteristic (ROC) curve is a single measure that captures the effectiveness of a classifier. The ROC curve is a plot of the true positive rate against the false positive rate for different classification thresholds. The area under the curve (AUC) provides a measure of discrimination between positive and negative classes. The AUC can be interpreted as the probability that a positive observation is ranked higher than a negative observation. The AUC does not depend on the classification threshold as it is an aggregate measure. An AUC closer to 1 is reflective of a good model, while a random predictor will give an AUC closer to 0.5.

Accuracy Measure	Classifier	Mean				Standard deviation			
		t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
PPV	SAVEC	95	95	100	95	8	8	0	8
	DAVEC	74	69	68	68	18	15	12	23
	Logistic regression	91	75	75	70	16	31	31	34
NPV	SAVEC	91	92	93	92	6	3	4	5
	DAVEC	98	100	99	100	2	0	1	0
	Logistic regression	95	95	95	92	1	1	3	5
AUC	SAVEC	93	94	96	94	5	5	2	5
	DAVEC	94	97	94	97	7	0	5	1
	Logistic regression	93	85	85	81	7	15	15	14

Table 3: *Mean and standard deviation of PPV, NPV and AUC* (%) *over* 4 *folds.*

Table 3 gives the average PPV, NPV and AUC values with their standard deviations over the 4-folds for the fibre-optic data stream. For PPV and NPV, we use a probability threshold of 0.5, i.e. if the output probability is greater than 0.5, it is deemed class A, and class B otherwise. A high PPV signifies a low false positive rate and a high NPV, a low false negative rate. From Table 3, we see that SAVEC has the highest average PPV and DAVEC has the highest average NPV for all event ages. Of the two classifiers, SAVEC is more conservative in predicting class A events compared with the DAVEC. That is, if SAVEC predicts an event as belonging to class A, it is more likely that the actual event is of class A compared with DAVEC. This is supported by the high mean PPV values and the associated low standard deviations of SAVEC. On the other hand, DAVEC is much more likely to catch all class A events compared with SAVEC as seen by the higher NPV values, i.e. there may be more false positives but a low number of false negatives indicate that class A events are unlikely to be labelled as class B. Whether it is better to be conservative and miss some class A events while being very accurate in terms of the predicted class A events, or to identify all class A events while giving some false positives, depends on the application. Often the cost of false positives and false negatives are not the same; for example consider a medical test for cancer. Thus, we see that our proposed two models have different strengths. In addition, both these models outperform the logistic regression classifier as seen by the AUC values.

5.3 Nitrogen dioxide monitoring

The second application focuses on event extraction from Nitrogen Dioxide (NO_2) data from NASA's NEO website (*NASA Earth Observations* 2004). Our aim is to show the applicability of our event extraction algorithm in diverse applications. The Ozone Monitoring Instrument (OMI) (Levelt et al. 2006) aboard the Aura satellite records a variety of air quality measures including NO_2 concentrations around the world. We use some of this data to demonstrate the event extraction and feature computation process for a 3-dimensional data stream.

We use OMI NO_2 monthly data from March to June in the years 2008 and 2018 for a comparison study. For each month the data comes in a matrix of 1799×3600 dimensions. For ease of calculation we reduce the dimension of this matrix to 180×360, by computing the average of each 10×10 block of data in the original matrix. The OMI NO_2 data for two months (March 2008 and March 2018) are shown in Figure 13.

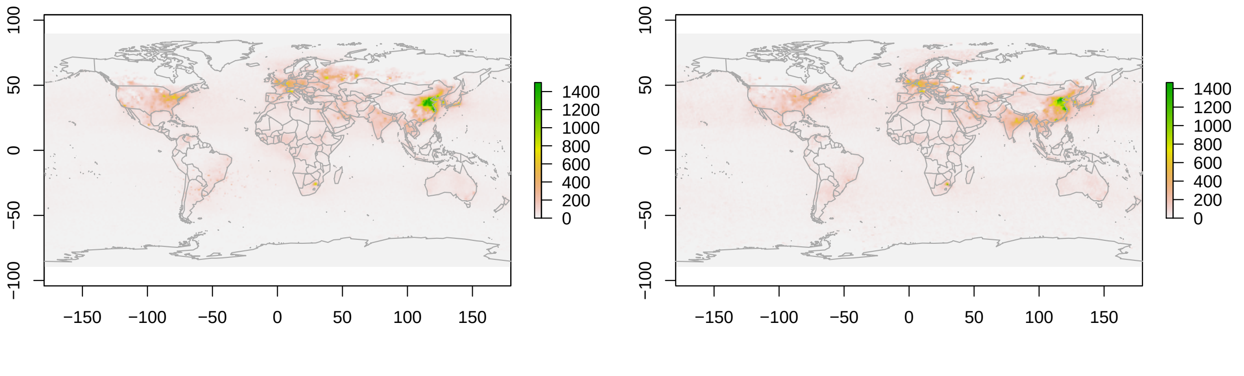


Figure 13: *NO_2 data for March 2008 (Figure 13a) and March 2018 (Figure 13b).*

Using Algorithm 1, we extract events from this data set, which are clusters of high NO_2 levels spanning space and time. Events are extracted from a 3-dimensional data stream of $4 \times 180 \times 360$ array, where each 180×360 matrix corresponds to the NO_2 levels of a given month. The parameters used in Algorithm 1 are $\alpha = 0.97$, $\epsilon = 2$ and minPts = 20. There are 23 events/clusters in the 2008 data and 13 events/clusters in 2018 data. Figure 14 shows these clusters for March 2008 and March 2018 (corresponding to the data in Figure 13) with each cluster/event depicted by a single colour.

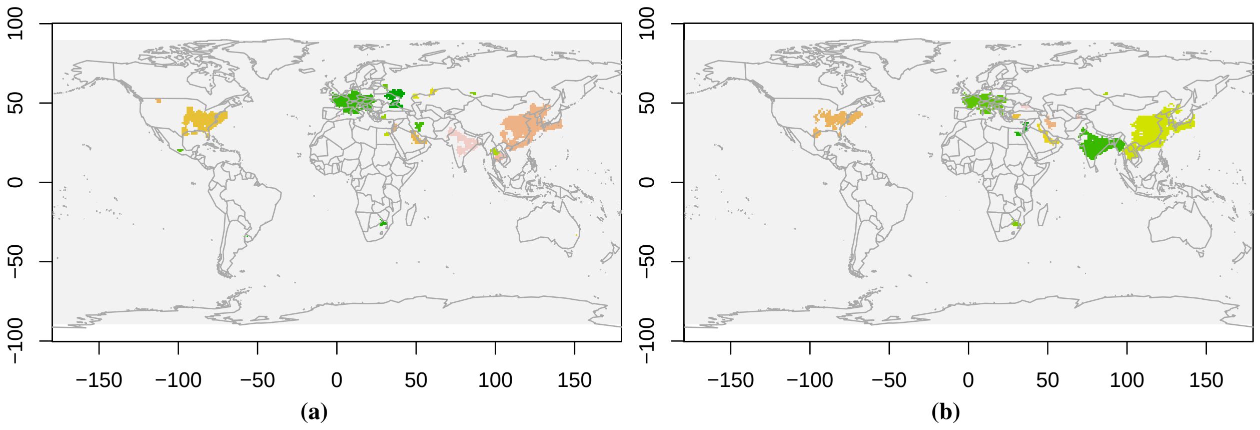


Figure 14: *Events extracted from NO_2 data for March 2008 (Figure 14a) and March 2018 (Figure 14b). Colours do not reflect NO_2 levels. Each event is depicted by a single colour.*

For each event we compute features 1–8 and 14 from the list of features in Section 2.2. These features are chosen for ease of computation. Specially as we are not focusing on event classification in this application, we do not compute the other features which involve fitting splines for 3-dimensional data.

We analyse the two events that have the highest average NO_2 levels for each of 2008 and 2018. The maps in Figure 15 show the spatial locations and the NO_2 levels of these two events for 2008 and 2018. As these two events are geographically at the same location, we compare the average NO_2 levels for these two events for the months from March to June in 2008 with those of 2018. The graph in Figure 16 shows that the average NO_2 levels have decreased in 2018 compared to 2008 for this geographical location.

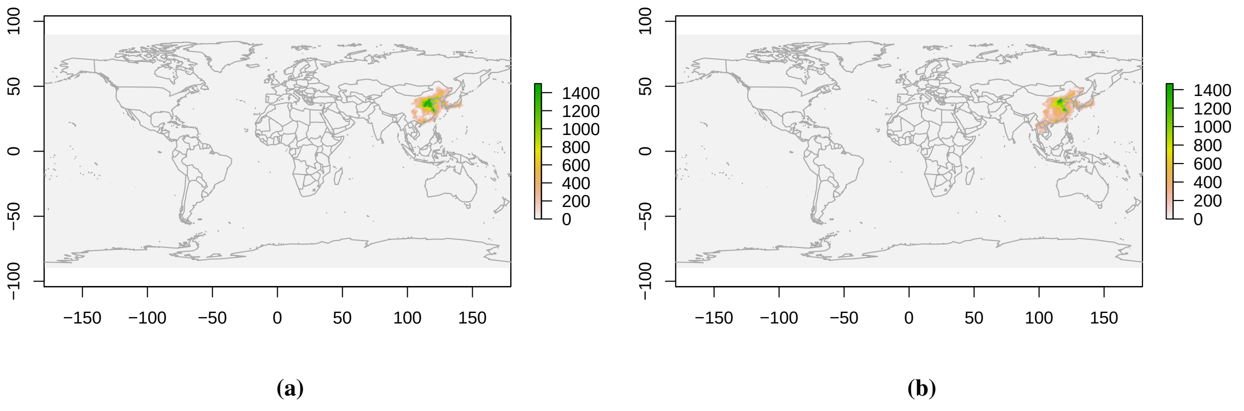


Figure 15: *The two events with the highest average NO_2 levels for March 2008 (Figure 15a) and March 2018 (Figure 15b).*

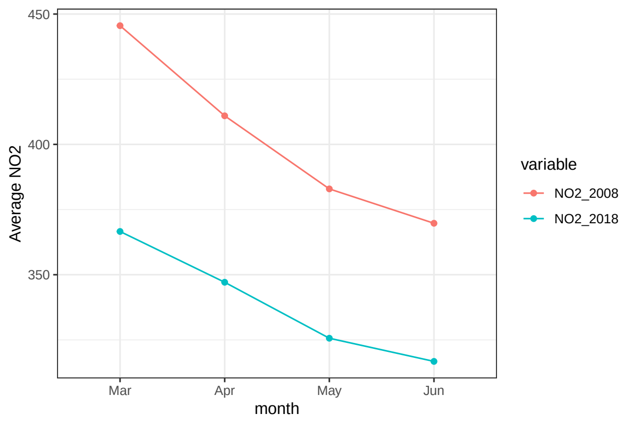


Figure 16: *The average NO_2 levels for the two events in Figure 15.*

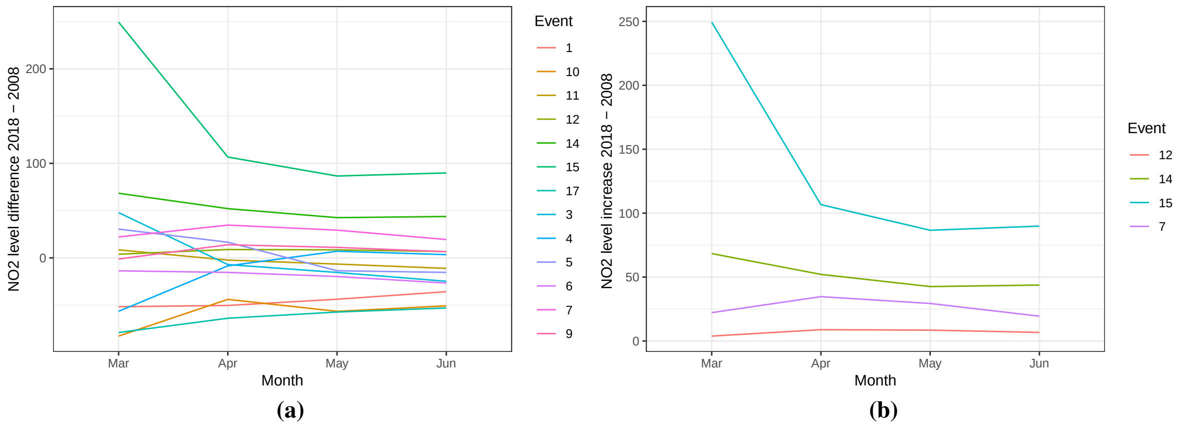


Figure 17: *(a) Difference in average NO_2 levels (2018 – 2008) for all matched events. (b) The events for which average NO_2 level difference is positive (average NO_2 levels have increased from 2008 to 2018).*

Next we match the events in 2008 and 2018 by their spatial location and perform a simple analysis. We want to know if these NO_2 events have increased or decreased in severity during this 10 year time gap. For each of these matched events we find the average NO_2 level difference between 2018 and 2008. Figure 17a shows the graph of these differences and from Figure 17b we see that 4 events have positive NO_2 differences for each month, i.e., the average NO_2 levels have increased from 2008 to 2018 for these 4 events. In addition, event 15 is different from

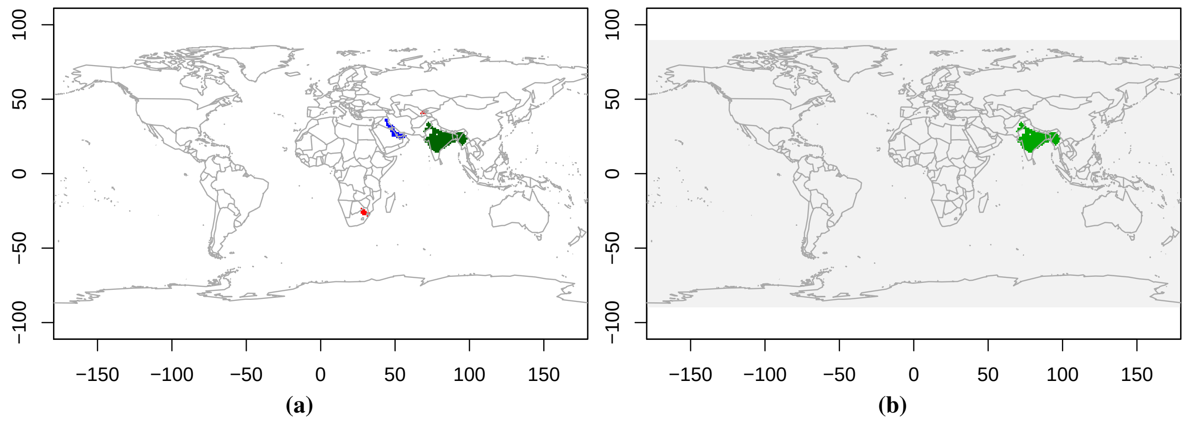


Figure 18: *The location in March 2018 of events with increased average NO_2 levels in Figure 18a, and the outlying NO_2 event in Figure 18b.*

other events. Figure 18 shows the locations of the increased NO_2 events in March 2018 along with the location of the outlying event 15.

While the increased industrialisation in India and the ongoing war in the Middle-East may be contributors, an investigation of reasons behind the increased NO_2 levels is beyond the scope of this study. Our aim is to demonstrate the applicability of our event extraction and event classification algorithms for different applications.

6 Conclusions

This paper has proposed a framework for event extraction and event classification in data streams with a focus on early event classification. We proposed two early event classifiers, SAVEC and DAVEC. SAVEC is a static classifier suited for stable data distributions and DAVEC is a dynamic classifier suited for dynamic or non-stationary data distributions. We have tested our early classification framework using 2 applications, one synthetic and one real. We have shown that we obtain better accuracy results compared to logistic regression for these applications.

In addition, we have proposed an algorithm for event extraction, which can be used on two or three dimensional data streams. The applicability of our event extraction algorithm has been demonstrated using the NO_2 data from NASA's NEO website.

Future directions for this research include extending SAVEC to use other base classifiers and including a greater variety of event extraction processes.

7 Supplementary material

R-package eventstream: This package contains Algorithm 1 for event extraction, SAVEC for early event classification, functions for synthetic data generation, the fibre-optic data stream and NO_2 data for 2008 and 2018. It is available from Github at `https://github.com/sevvandi/eventstream`. DAVEC is built using existing functionality of the R package *dma* (McCormick et al. 2018).
Scripts: There are three files containing the R code used in Section 5. The file `Supp_Mat_1.R`, `Supp_Mat_2.R` and `Supp_Mat_3.R` contain the code applicable for synthetic data in Section 5.1, fibre optic data in Section 5.2 and NO_2 data in Section 5.3 respectively. The file `Supp_Mat_4.R` contains the code used to produce some other graphs and images in the paper.
Other R-packages: We have used the following R-packages either in this paper or within the package *eventstream* : *abind* (Plate & Heiberger 2016), *AtmRay* (Anderson 2013), *pROC* (Robin et al. 2011), *reshape2* (Wickham 2007), *ggplot2* (Wickham 2016), *raster* (Hijmans 2018), *maps* (Becker et al. 2018), *tensorA* (van den

Boogaart 2018), *glmnet* (Friedman et al. 2010), *dbscan* (Hahsler & Piekenbrock 2018) and *MASS* (Venables & Ripley 2002).

Acknowledgements

Funding was provided by the Australian Research Council through the Linkage Project LP160101885. The first author would like to thank Dr Andres Ramirez Hassan for useful discussions.

References

Abdelhaq, H., Sengstock, C. & Gertz, M. (2013), 'Eventweet: Online localized event detection from twitter', *Proceedings of the VLDB Endowment* **6**(12), 1326–1329.

Adam, A., Rivlin, E., Shimshoni, I. & Reinitz, D. (2008), 'Robust real-time unusual event detection using multiple fixed-location monitors', *IEEE transactions on pattern analysis and machine intelligence* **30**(3), 555–560.

Alippi, C. & Roveri, M. (2008*a*), 'Just-in-time adaptive classifiers—part i: Detecting nonstationary changes', *IEEE Transactions on Neural Networks* **19**(7), 1145–1153.

Alippi, C. & Roveri, M. (2008*b*), 'Just-in-time adaptive classifiers—part ii: Designing the classifier', *IEEE Transactions on Neural Networks* **19**(12), 2053–2064.

Allan, J., Papka, R. & Lavrenko, V. (1998), On-line new event detection and tracking, *in* 'Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval', ACM, pp. 37–45.

Anderson, J. (2013), *AtmRay: Acoustic Traveltime Calculations for 1-D Atmospheric Models*. R package version 1.31.
URL: *https://CRAN.R-project.org/package=AtmRay*

Becker, R. A., Wilks, A. R., Brownrigg, R., Minka, T. P. & Deckmyn., A. (2018), *maps: Draw Geographical Maps*. R package version 3.3.0.
URL: *https://CRAN.R-project.org/package=maps*

Duchi, J., Hazan, E. & Singer, Y. (2011), 'Adaptive subgradient methods for online learning and stochastic optimization', *Journal of Machine Learning Research* **12**(Jul), 2121–2159.

Durbin, J., Koopman, S.-J. et al. (1998), *Time series analysis of non-Gaussian observations based on state space models from both classical and Bayesian perspectives*, Tilburg University.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X. et al. (1996), A density-based algorithm for discovering clusters in large spatial databases with noise., *in* 'Kdd', Vol. 96, pp. 226–231.

Frey, P. W. & Slate, D. J. (1991), 'Letter recognition using holland-style adaptive classifiers', *Machine learning* **6**(2), 161–182.

Friedman, J., Hastie, T. & Tibshirani, R. (2001), *The elements of statistical learning*, Vol. 1, Springer series in statistics New York, NY, USA:.

Friedman, J., Hastie, T. & Tibshirani, R. (2010), 'Regularization paths for generalized linear models via coordinate descent', *Journal of Statistical Software* **33**(1), 1–22.
URL: *http://www.jstatsoft.org/v33/i01/*

Gaber, M. M., Zaslavsky, A. & Krishnaswamy, S. (2005), 'Mining data streams: a review', *ACM Sigmod Record* **34**(2), 18–26.

Gama, J. (2010), *Knowledge discovery from data streams*, Chapman and Hall/CRC.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M. & Bouchachia, A. (2014), 'A survey on concept drift adaptation', *ACM computing surveys (CSUR)* **46**(4), 44.

Griffiths, B. (1995), Developments in and applications of fibre optic intrusion detection sensors, *in* 'Security Technology, 1995. Proceedings. Institute of Electrical and Electronics Engineers 29th Annual 1995 International Carnahan Conference on', IEEE, pp. 325–330.

Hahsler, M. & Piekenbrock, M. (2018), *dbscan: Density Based Clustering of Applications with Noise (DBSCAN) and Related Algorithms*. R package version 1.1-2.
URL: *https://CRAN.R-project.org/package=dbscan*

Hastie, T. & Tibshirani, R. (1993), 'Varying-coefficient models', *Journal of the Royal Statistical Society. Series B (Methodological)* pp. 757–796.

Hijmans, R. J. (2018), *raster: Geographic Data Analysis and Modeling*. R package version 2.8-4.
URL: *https://CRAN.R-project.org/package=raster*

Hulten, G., Spencer, L. & Domingos, P. (2001), Mining time-changing data streams, *in* 'Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 97–106.

Jiang, Q. & Sui, Q. (2009), Technological study on distributed fiber sensor monitoring of high voltage power cable in seafloor, *in* 'Automation and Logistics, 2009. ICAL'09. IEEE International Conference on', IEEE, pp. 1154–1157.

Kandanaarachchi, S. (2018), *eventstream: An implementation of streaming events and their classification*. R package version 0.0.9000.
URL: *https://github.com/sevvandi/eventstream*

Ke, Y., Sukthankar, R. & Hebert, M. (2005), Efficient visual event detection using volumetric features, *in* 'Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on', Vol. 1, IEEE, pp. 166–173.

Klinkenberg, R. & Joachims, T. (2000), Detecting concept drift with support vector machines., *in* 'ICML', pp. 487–494.

Levelt, P. F., van den Oord, G. H., Dobber, M. R., Malkki, A., Visser, H., de Vries, J., Stammes, P., Lundell, J. O. & Saari, H. (2006), 'The ozone monitoring instrument', *IEEE Transactions on geoscience and remote sensing* **44**(5), 1093–1101.

Li, H.-N., Li, D.-S. & Song, G.-B. (2004), 'Recent applications of fiber optic sensors to health monitoring in civil engineering', *Engineering structures* **26**(11), 1647–1657.

Li, R., Lei, K. H., Khadiwala, R. & Chang, K. C.-C. (2012), Tedas: A twitter-based event detection and analysis system, *in* 'Data engineering (icde), 2012 ieee 28th international conference on', IEEE, pp. 1273–1276.

Mao, Y., Jie, Q., Jia, B., Ping, P. & Li, X. (2015), Online event detection based on the spatio-temporal analysis in the river sensor networks, *in* 'Information and Automation, 2015 IEEE International Conference on', IEEE, pp. 2320–2325.

McCormick, T. H., Raftery, A. E., Madigan, D. & Burd, R. S. (2012), 'Dynamic logistic regression and dynamic model averaging for binary classification', *Biometrics* **68**(1), 23–30.

McCormick, T. H., Raftery, A., Madigan, D., Kandanaarachchi, S. & Sevcikova, H. (2018), *dma: Dynamic Model Averaging*. R package version 1.4-0.
URL: *https://CRAN.R-project.org/package=dma*

Medioni, G., Cohen, I., Brémond, F., Hongeng, S. & Nevatia, R. (2001), 'Event detection and analysis from video streams', *IEEE Transactions on pattern analysis and machine intelligence* **23**(8), 873–889.

NASA Earth Observations (2004), `https://neo.sci.gsfc.nasa.gov/view.php?datasetId=AURA_NO2_M`.

Niklès, M., Vogel, B. H., Briffod, F., Grosswig, S., Sauser, F., Luebbecke, S., Bals, A. & Pfeiffer, T. (2004), Leakage detection using fiber optics distributed temperature monitoring, *in* 'Smart Structures and Materials 2004: Smart Sensor Technology and Measurement Systems', Vol. 5384, International Society for Optics and Photonics, pp. 18–26.

Nishida, K., Yamauchi, K. & Omori, T. (2005), Ace: Adaptive classifiers-ensemble system for concept-drifting environments, *in* 'International Workshop on Multiple Classifier Systems', Springer, pp. 176–185.

Plate, T. & Heiberger, R. (2016), *abind: Combine Multidimensional Arrays*. R package version 1.4-5.
URL: *https://CRAN.R-project.org/package=abind*

Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.-C. & Müller, M. (2011), 'proc: an open-source package for r and s+ to analyze and compare roc curves', *BMC Bioinformatics* **12**, 77.

Suthaharan, S. (2014), 'Big data classification: Problems and challenges in network intrusion prediction with machine learning', *ACM SIGMETRICS Performance Evaluation Review* **41**(4), 70–73.

Tsymbal, A. (2004), 'The problem of concept drift: definitions and related work', *Computer Science Department, Trinity College Dublin* **106**(2).

van den Boogaart, K. G. (2018), *tensorA: Advanced Tensor Arithmetic with Named Indices*. R package version 0.36.1.
URL: *https://CRAN.R-project.org/package=tensorA*

Venables, W. N. & Ripley, B. D. (2002), *Modern Applied Statistics with S*, fourth edn, Springer, New York. ISBN 0-387-95457-0.
URL: *http://www.stats.ox.ac.uk/pub/MASS4*

Weng, J. & Lee, B.-S. (2011), Event detection in twitter, *in* 'Fifth international AAAI conference on weblogs and social media'.

Wickham, H. (2007), 'Reshaping data with the reshape package', *Journal of Statistical Software* **21**(12), 1–20.
URL: *http://www.jstatsoft.org/v21/i12/*

Wickham, H. (2016), *ggplot2: Elegant Graphics for Data Analysis*, Springer-Verlag New York.
URL: *http://ggplot2.org*

Widmer, G. & Kubat, M. (1996), 'Learning in the presence of concept drift and hidden contexts', *Machine learning* **23**(1), 69–101.

Yin, J., Hu, D. H. & Yang, Q. (2009), Spatio-temporal event detection using dynamic conditional random fields., *in* ‘Ijcai’, Vol. 9, pp. 1321–1327.