Learning data streams with changing distributions and temporal dependency

Yiliao Song *Member, IEEE*, Jie Lu *Fellow, IEEE*, Haiyan Lu *Senior Member, IEEE*, Guangquan Zhang

***Abstract*—In a data stream, concept drift refers to unpredictable distribution changes over time, which violates the identical-distribution assumption required by conventional machine learning methods. Current concept drift adaptation techniques mostly focus on a data stream with changing distributions. However, since each variable of a data stream is a time series, these variables normally have temporal dependency problems in the real world. How to solve concept drift and temporal dependency problems at the same time is rarely discussed in the concept-drift literature. To solve this situation, this paper proves and validates that, the testing error decreases faster if a predictor is trained on a temporally reconstructed space when drift occurs. Based on this theory, a novel drift adaptation regression (DAR) framework is designed to predict the label variable for data streams with concept drift and temporal dependency. A new statistic called LDD$^+$ is proposed and used as a drift adaptation technique in the DAR framework to discard outdated instances in a timely way, thereby guaranteeing that the most relevant instances will be selected during the training process. The performance of DAR is demonstrated by a set of experimental evaluations on both synthetic data and real-world data streams.**

***Index Terms*—concept drift, drift adaptation, non-stationary environment, data stream**

I. Introduction

A data stream is potentially infinite amounts of data that arrive in a sequential way from a variety of sources such as economics, industrial monitoring, ecosystems, and so on [1], [2]. New challenges have appeared in data stream mining, one of which relates to data distribution changes [3]. Standard machine learning approaches are built on a static assumption of independent and identically distributed (i.i.d) data and therefore are not suitable for learning data streams once the data distribution has experienced unpredictable changes [4], [5].

Concept drift refers to these unpredictable distribution changes over time, and concept drift adaptation aims to solve the concept drift problem by continuously updating the trained predictors [6]. However, existing adaptation methods assume that the data stream only contains the concept drift problem. In fact, data streams may also have other uncertain characteristics [7]. In a data stream, each variable is now a time series process that is probably autocorrelated, which leads to the temporal dependency problem. Dealing with the problems of temporal dependency and concept drift simultaneously is a big challenge [8].

Y. Song, J. Lu, H. Lu and G. Zhang are with the Decision Systems and e-Service Intelligence Laboratory, Australian Artificial Intelligence Institute, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia (e-mail: Yiliao.Song@student.uts.edu.au; Jie.Lu@uts.edu.au; Haiyan.Lu@uts.edu.au; Guangquan.Zhang@uts.edu.au.) All the data and codes are available via https://github.com/songyiliao/DAR.

TABLE I
DIFFERENCE AMONG INCREMENTAL LEARNING, CONCEPT DRIFT ADAPTATION, AND OUR SETTING.

	Stationary data stream	Data stream with changing distributions	Data stream with changing distributions and temporal dependency
Incremental learning	✓	✗	✗
Drift adaptation	✓	✓	✗
Our setting	✓	✓	✓

This paper discusses the concept drift problem in a more realistic scenario where data streams have concept drift and temporal dependency problems at the same time. Classic incremental learning discusses the problem of how to learn a predictor when the training data is available one by one or batch by batch, and thus it is widely used for predicting data streams. Concept drift adaptation discusses how to continuously learn predictors when future data follow different data distributions. This paper discusses how to continuously learn predictors when the data instances are temporally dependent and the data distribution may change at the same time. The difference between incremental learning, drift adaptation, and our setting is presented in Table I.

To handle this kind of data stream, a novel drift-adapted framework named DAR is proposed in regression cases. In DAR, we propose to train the predictor on a reconstructed feature space to solve the temporal dependency problem. In addition, we formally propose a new way to embed the drift detection techniques into an informed adaption method. Compared to a general informed drift adaption method which only uses detection information when drift occurs, our drift adaptation method is able to extract more statistical information from the drift detection process and achieves an accurate and robust online prediction performance.

The novelty and main contributions of this paper are as follows:

1) It fills the research gap of handling a data stream with concept drift and the temporal dependency problem from the aspect of time series processes;
2) It proves that testing errors decrease faster in a linear predictor trained on the reconstructed space for data streams with concept drift and temporal dependency. We compare the error decreasing speed of linear predictors trained on the original and reconstructed feature spaces;
3) The conclusion in 2) is generalized to non-linear cases by introducing locally weighted regression so that we

can use this conclusion for a general regression task for predicting data streams with concept drift and temporal dependency;

4) We develop a new statistic called LDD$^+$ to measure the distance from a data instance to a high-dimensional data distribution. Instead of using LDD$^+$ to detect whether drift occurs, we implement drift adaptation based on the value of LDD$^+$ when every new instance arrives;
5) Combining the above aspects, a drift-adapted regression framework (DAR) is proposed for data streams with concept drift and temporal dependency.

The paper is organized as follows. Section II discusses related works in the concept drift area. Section IV explains the proposed DAR framework in detail. Section V outlines the results of the synthetic and real experimental evaluation. Section VI concludes the study and discusses future work.

II. Related Work

Concept drift adaptation aims to design a blind or an informed strategy to update predictors for obtaining accurate real-time prediction results [9]. A *blind* adaptation refers to drift adaptation without drift detection techniques, which is to passively update the predictor [10], [11], [12]. Many ensemble learning techniques in the concept drift area use a blind adaptation strategy [13], [7]. An *informed* adaptation denotes that drift adaptation is based on drift detection analysis [14], which is to actively update the predictor [15], [16]. So far, most informed adaptation updates the predictor when the drift detection method identifies the occurrence of drift [17]. For example, in [18], the predictor is updated when an alarm is triggered—the designed estimator is larger than a threshold. Similar studies are presented in [19], [20]. Instead of only using information on *when* drift occurs to update predictors, a recent survey pointed out that an understanding of *where* and *how* drift occurs is also important for informed adaptation[17], especially for data-driven decision support [3]. A few methods in recent research discuss where drift occurs, such as [21], [22].

Both the blind and informed drift adaptation methods have been validated to effectively handle the concept drift problem in a data stream [23], [24]. As blind adaptation methods do not discuss whether drift truly exists[25], these methods are very similar to incremental/online learning from a technical aspect [26]. The learner error may accumulate when a data stream contains mixed types of drift, as discussed in [27]. Compared to blind adaptation, informed adaptation is more robust because the detection process is able to avoid the old/bad information presented in the predictor [22].

In recent years, research on concept drift has focused on more realistic problems. For example, concept drift with imbalanced data involves the situation where the learning boundary changes for classification data that at least one class only has limited samples of data [28], [29], [30]. Another situation is that concept drift appears in noisy data [31], [32]. In such situations, it is very difficult to identify whether the decrease in learner accuracy is caused by concept drift or noise. In particular, as the data streams are obtained from different sources at the same time, concept drift in multiple streams [33], [34], [35] has also gained increasing attention in academic research and practical applications.

As the concept drift problem is always discussed in a data stream, one of the important characteristics of data streams has been neglected in existing solutions: temporal dependency. In fact, each variable of a data stream is a time series process that is temporal dependent. Therefore, a good prediction method for real-time data streams should be able to handle the concept drift problem and temporal dependency at the same time. This paper provides a comprehensive study for such a situation from both theoretical and practical aspects.

III. Preliminaries

In this paper, we consider the data stream from the aspect of time series analysis. We refer to [36] for the definitions and characteristics of a time series and list them in this section. All the required preliminaries are presented in this section.

Definition 1 (Time Series [36])**.** *A time series process S_t is a sequence taken at successive equally spaced points in time.*

Definition 2 (Autocovariance [36])**.** *Autocovariance is an important property of a time series process S_t, which is*

$$\gamma_\tau = E\left[(S_t - \mu_t)(S_{t+\tau} - \mu_{t+\tau})\right]. \tag{1}$$

where μ_t and $\mu_{t+\tau}$ are the expectations of S_t and $S_{t+\tau}$. The autocovariance is the covariance if S_t and $S_{t+\tau}$ are two random variables.

If the time series process is autoregressive, γ depends on τ rather than being a fixed constant [36]. This means that another basic assumption in conventional machine learning, *independence, is also invalid.*

Definition 3 (Covariance-stationary and Ergodic for the Mean [36])**.** *A time series process S_t is covariance-stationary and ergodic for the mean if*

$$\begin{cases} E(S_t) = \mu & \text{for all } t \\ E\left[(S_t - \mu)(S_{t-\tau} - \mu)\right] = \gamma_\tau & \text{for all } t \text{ and } \tau \\ (1/T)\sum_{t=1}^{T} S_t \xrightarrow{p} E(S_t) & \text{as } T \to \infty. \end{cases} \tag{2}$$

Remark 1. *The first two equations denote that neither the mean μ_t nor the autocovariance γ_τ depends on time t*[1]*. The third equation guarantees that the time average will eventually converge to the expectation $E(S_t)$.*

Definition 4 (pth-order Autoregressive Process [36])**.** *A pth-order autoregressive process S_t satisfies*

$$S_t = c + \phi_1 S_{t-1} + \phi_2 S_{t-2} + ... + \phi_p S_{t-p} + \epsilon_t, \tag{3}$$

where S_{t-1},..., S_{t-p} are lag orders (earlier observations) of S_t and ϵ_t is the white noise sequence (Definition 5).

Definition 5 (White Noise Sequence [36])**.** *A white noise sequence is a sequence $\{\epsilon_t\}_{t=-\infty}^{\infty}$ satisfying:*

$$E(\epsilon_t) = 0, E(\epsilon_t^2) = \sigma^2 \text{ and } E(\epsilon_t, \epsilon_\tau) = 0 \text{ for } t \neq \tau. \tag{4}$$

[1]This does not conflict with (2), γ_τ depends on τ but is independent on t

Remark 2. *Definition 4 can be simply rewritten by* $\sum_{i=0}^{p} \mathcal{L}^{(i)} S_t$ *by introducing the Lag operator* $\mathcal{L}$ *that for any integer* k, $\mathcal{L}^k S_t = S_{t-k}$ *[36].*

IV. The drift-adapted regression framework for temporal dependent data streams

This paper discusses the concept drift problem in a more realistic scenario where the data stream also contains temporal discrepancy. In this scenario, the i.i.d assumption is invalid as the data instances are neither independent nor identically distributed. A drift-adapted regression (DAR) framework is proposed to predict the data stream with both concept drift and temporal dependency problems. We propose a new statistic LDD^+ to implement an informed drift adaptation procedure in DAR. Instead of using LDD^+ to identify whether drift occurs, LDD^+ is designed to rank the importance of each instance. The information of instance importance can help to update the training data when every new instance arrives and thus solve the adaptation delay problem in most informed adaptation methods.

In this section, details of the proposed DAR framework are introduced. Section IV-A gives the assumptions and definitions of a data stream that has concept drift and temporal dependency problems; based on these assumptions and definitions, Section IV-B describes the learning task in data streams with concept drift and temporal dependency. In the end of Section IV-B, we give a brief summary of the proposed definitions in Section IV-A and Section IV-B and their relationship. Section IV-C gives the theoretical foundations of the basic idea of our proposed DAR framework, and Section IV-D explains the adaptation procedure and how to implement the DAR framework.

A. Assumptions and definitions

We consider the data stream from an aspect of time series analysis. A data stream consists of $d+1$ time series processes. This section starts with the definition of a *data stream*, and then the *concept drift* definition is introduced and expanded. Based on these definitions and definitions of time series, the definition of a *data stream with concept drift and temporal dependency* is presented and explained.

Definition 6 (Data Stream)**.** *A data stream* $D_t = \{(\boldsymbol{X}_t, y_t) \,|t = 1, ...\infty\}$*, is generated from distribution* $\mathcal{P}_t$ *with* $p_t(\boldsymbol{X}, y)$ *its probability function or probability density function (pdf), where* $\left\{\boldsymbol{X}_t \in \mathbb{R}^d\right\}$ *is the attribute variable (or the input) consisting of* d *time series, for some* d*, and* $\left\{y_t \in \mathbb{R}^1\right\}$ *is the label variable (or the scalar output).*

So far, concept drift is defined in Definition. 7. This widely accepted definition of concept drift highlights the characteristics of *drift*, but it does not explain the meaning of "*concept*". When studying the problem of concept drift, the term "concept" is used to represent hidden data patterns such as the probability distributions and relationships between $\boldsymbol{X}_t$ and y_t. Concept drift is caused by the hidden context [37], rather than stochastic disturbances. Unlike outliers, a concept will last for a period after it shows, rather than existing momentarily. To present this characteristics of concept, a constraint is added to the current definition of concept drift as presented in Definition 8.

Definition 7 (Concept Drift (short version))**.** *Concept drift is defined if the underlying distribution changes, i.e.,* $\exists t$ *that* $p_{t+1}(\boldsymbol{X}, y) \neq p_t(\boldsymbol{X}, y)$.

Definition 8 (Concept Drift (full version))**.** *Concept drift occurs in a data stream if* $\exists t_{d^{(i)}}$ *that*

$$\begin{cases} p_{t+1}(\boldsymbol{X}, y) \neq p_t(\boldsymbol{X}, y), \text{ for } t = t_{d^{(i)}} \\ p_{t+1}(\boldsymbol{X}, y) = p_t(\boldsymbol{X}, y), \text{ for } t \in \left[t_{d^{(i)}+\tau_i}, t_{d^{(i+1)}}\right) \end{cases} \tag{5}$$

where $\forall i,\ t_{d^{(i+1)}} - t_{d^{(i)}} > 1$, $t \in \mathbb{Z}^+$ *presents the time step,* $d^{(i)}$ *is an order statistics denoting the* i^{th} *drifted time point, and* $\tau_i = 1$ *is for the sudden drift while* $1 < \tau_i < t_{d^{(i+1)}} - t_{d^{(i)}}$ *is specifically for the occurrence of incremental drift.*

Remark 3. *In the full version definition, a data stream contains concept drift if the data pattern changes at least once, namely* $\{t_{d^{(i)}}\} \neq \emptyset$ *that* $p_{t+1}(\boldsymbol{X}, y) \neq p_t(\boldsymbol{X}, y)$*, for* $t = t_{d^{(i)}}$*; in addition, the changed pattern is not ephemeral, but will last for a period (at least last for two time steps), which is manifested by* $\forall i,\ t_{d^{(i+1)}} - t_{d^{(i)}} > 1$*. The pattern stays the same in this period that* $p_{t+1}(\boldsymbol{X}, y) = p_t(\boldsymbol{X}, y)$*, for* $t \in \left[t_{d^{(i)}+\tau_i}, t_{d^{(i+1)}}\right]$*; here* $\tau_i = 1$ *when the drift occurs suddenly while* $\tau_i > 1$ *when the drift occurs incrementally in the period of* $\left[t_{d^{(i)}+1}, t_{d^{(i)}+\tau_i}\right]$*.* ***All drift adaptation methods are at least one-instance delayed. Without the constraint that a new pattern will be retained for a period, any adaptation is invalid in principle.***

The definition of a data stream contains $(\boldsymbol{X}_t, y_t)$, but does not give the temporal details of $\boldsymbol{X}$ and y. Next, $(\boldsymbol{X}_t, y_t)$ is explained from a time series aspect to present the temporal dependency. We assume that a data stream consists of time series processes, and these time series contain certain characteristics as well as unknown characteristics. We separate the certain characteristics from the unknown characteristics so that the effectiveness of these certain characteristics on modelling can be analyzed.

In this paper, we use decomposing and mixing to separate the certain characteristics from the unknown characteristics for each variable in a data stream. Next, we first define the decomposed time series and mixed time series. Based on these two definitions, a data stream with drift and temporal-dependency is defined.

Definition 9 (Decomposed Time Series)**.** *Each time series process in the data stream,* T_t*, is decomposed as the weighted sum of two time series process,* V_t *and* S_t*.*

$$T_t = (1 - s)V_t + sS_t, s \in [0, 1] \tag{6}$$

where V_t *is a time series process with unknown characteristics which represents the uncertainty aspect of this variable, and* S_t *process is covariance-stationary and ergodic for the mean (Definition 3) which represents the certain temporal dependency.* V_t *and* S_t *are assumed to be independent.*

Definition 10 (Mixed Time Series)**.** *Given that* $T_t^1, T_t^2, ..., T_t^n$ *are* n *time series processes (Definition 9), and* $\boldsymbol{\omega}_t$ *is the*

weighting vector satisfying $\sum_{i=1}^{n}\omega_t^{(i)}=1, |\omega_t|\leq 1$*, a mixed time series* MT_t *is defined as*

$$\begin{aligned} MT_t &= \sum_{i=1}^{n}\omega_t^{(i)}\times T_t^{(i)} \\ &= (1-s)\sum_{i=1}^{n}\omega_t^{(i)}V_t^{(i)} + s\sum_{i=1}^{n}\omega_t^{(i)}S_t^{(i)} \\ &= (1-s)MV_t + sMS_t. \end{aligned} \tag{7}$$

To describe data streams with drift and temporal-dependence, the data stream is considered to consist of $d+1$ time series processes, and each variable's values between two consecutive drift points (Definition 9) are considered to be a realization of special case of MT_t.

Definition 11 (Drift Point)**.** *Drift point is defined as the time point when a new concept starts. Namely the* $t_{d^{(i)}}$ *in Definition 8.*

Definition 12 (Data Stream with Drift and Temporal-dependence)**.** *The data stream with drift and temporal-dependence is defined as* $m+1$ *mixed time series.*

$$DS_t^{(\boldsymbol{X},y)} = \left\{MT_t^1, ..., MT_t^m, MT_t^y | s, \boldsymbol{\omega}, V, S\right\} \tag{8}$$

For example, a data stream contains two different patterns in which $y_t=\beta_1 X_t$ before time point t_d and $y_t=\beta_2 X_t$ after t_d, where X_t is a time series. Given $T_t^{x,1}=T_t^{x,2}=X_t$, $T_t^{y,1}=\beta_1 T_t^{x,1}$, $T_t^{y,2}=\beta_2 T_t^{x,1}$ four time series, $\omega_t^{x,1}$, $\omega_t^{y,1}$ equals 1 when $t<t_d$ while 0 when $t\geq t_d$, and $\omega_t^{x,2}$, $\omega_t^{y,2}$ equals 0 when $t<t_d$ while 1 when $t\geq t_d$. Given $MT_t^x=\omega_t^{x,1}T_t^{x,1}+\omega_t^{x,2}T_t^{x,2}$ and $MT_t^y=\omega_t^{y,1}T_t^{y,1}+\omega_t^{y,2}T_t^{y,2}$, this data stream is $\{MT_t^x, MT_t^y\}$.

Remark 4. *Clearly, in the above example,* X_t *and* y_t *themselves are time series but here we use a linear combination of time series—i.e.,*$(1-s)MV_t+sMS_t$*—to represent them because if they are considered as a single time series, their statistical properties are completely unknown. The definition of a mixed time series helps to abstract the regular components in* X_t *and* y_t *as* MS_t *and leave the chaos components as* MV_t*.*

So far, we have given the definition of a data stream with temporal dependency and concept drift. The drift is presented in a data stream $DS_t^{(\boldsymbol{X},y)}$ if $\omega_t\neq\omega_{t+1}$ for some t, and the temporal dependency is presented by the time series T_t. In this paper, we provide a solution to find the unbiased estimation for MS_t. However, the estimation on MV_t will not be discussed, because the characteristics of MV_t are unknown in our assumption. Therefore, for an arbitrary data stream $DS_t^{(\boldsymbol{X},y)}(s,\cdot)$, a bigger s means a better estimation by our method. This claim will also be validated in the experiments on synthetic data (Section V-B3). In the following discussion, the problem will be simplified to a special case of $s=1$, namely $MT_t^{(\boldsymbol{X},y)}=MS_t^{(\boldsymbol{X},y)}$.

For a data stream with drift and temporal dependency, each time series process $MS_t^{(i)}$ is assumed to be a pth-order autoregressive process denoted by AR(p).

B. Problem description

According to Definition 8, a concept will exist for at least a time period of τ once it appears, and the occurrence of concept drift at t_d means the end of one concept. During the time period of one specific concept, the learning aim is to obtain a predictor h_t for $p_t(\boldsymbol{X},y)$.

Definition 13 (Learning Objective at t-step)**.** *To predict the value of the label variable for a data stream at time step t, the learning aim is to obtain a predictor* h_t *for* $p_t(\boldsymbol{X},y)$*, which can be denoted as*

$$h_t = \underset{h\in\mathcal{H}}{\arg\min}\ \ell\left(h,\boldsymbol{X},y|\left(\boldsymbol{X},y\right)\in p_t\left(\boldsymbol{X},y\right)\right), \tag{9}$$

where $\mathcal{H}$ *is the hypothesis set,* $\ell:\mathbb{R}^1\times\mathbb{R}^1\to\mathbb{R}_+$ *is the loss function used to measure the magnitude of error.*

In the regression task, the loss function is normally the squared loss, i.e., $\ell\left(h,\boldsymbol{X},y\right)=\left\|h\left(\boldsymbol{X}\right)-y\right\|_2^2$.

Definition 14 (Learning Objective for a Data Stream $(\boldsymbol{X}_t,y_t)$)**.** *The aim of the whole learning process for a data stream* $(\boldsymbol{X}_t,y_t)$ *is to find* h_t *for each concept.*

$$\underset{h_1,h_2,\ldots,h_t,\ldots}{\arg\min}\sum_t \ell\left(h_t,\boldsymbol{X}_t,y_t|\left(\boldsymbol{X}_t,y_t\right)\in p_t\left(\boldsymbol{X},y\right)\right). \tag{10}$$

Referring to Definition 14, the learning process of the data stream $DS_t^{(\boldsymbol{X},y)}$ will be as in Definition 15. The lag operator $\mathcal{L}$ (Remark 2) is used for a concise format.

Definition 15 (Learning Objective for Data Streams with Drift and Temporal-dependence $DS_t^{(\boldsymbol{X},y)}$)**.**

$$\underset{h_1,h_2,\ldots,h_t,\ldots}{\arg\min}\sum_t \ell\left(h_t,\boldsymbol{X},y,\mathcal{L}|\left(\boldsymbol{X},y\right)\sim p_t\left(\boldsymbol{X},y\right)\right). \tag{11}$$

Remark 5. *Compared to the previous learning task for data streams, the new loss function in* (11) *contains* $\mathcal{L}$*, which denotes the temporal dependency.*

Brief summary of Section IV-A and IV-B. Although it is common sense that the data stream consists of time series, the characteristics of these time series are unknown. The idea of decomposed time series (Definition 9) is to separate the certain temporal dependency from other uncertain characteristics so that we can further study data streams from the time series aspect. If the data stream does not have the problem of concept drift, Definition 9 is enough for further study. The occurrence of concept drift in a data stream breaches the first condition in Definition 3, which makes Definition 9 useless. Therefore, mixed time series (Definition 10) is proposed. The definition of mixed time series converts the problem of concept drift into the problem of changing weights so that the occurrence of concept drift does not breach the conditions in Definition 3 and the lag operator $\mathcal{L}$ could be added in Definition 15. Although Definitions 1-14 are not directly presented in the final algorithm, they are the theoretical foundations to guarantee the feasibility of the designed algorithm as well as the possible foundations for future studies on non-i.i.d problems.

C. Analysis of testing error when real drift exists

In this section, we discuss how to find a better solution to each h_t in (11) step by step. We start from the linear case

(Section IV-C1) and then expand the conclusion of the linear case into a general case (Section IV-C2).

In this section, virtual drift is not considered because when the learning objective is to obtain less error and a squared loss is considered in the regression task, the concept drift problem focuses on real drift. To prove this, we introduce Theorem 1.

Theorem 1. *The estimation with smallest squared loss is the expectation of y conditional on $\boldsymbol{X}$: $\hat{y} = h(\boldsymbol{X}) = E(y|\boldsymbol{X})$ where $h = \underset{h \in \mathcal{H}_{reg}}{\arg\min}\, \ell(h, \boldsymbol{X}, y)$.*

Proof. The proof is given in Appendix A. □

Assuming the hypothesis set contains the optimal predictor $E(y|\boldsymbol{X})$, if $p(\boldsymbol{X})$ (virtual drift) changes but $p(y|\boldsymbol{X})$ (real drift) stays the same, the current predictor can be the same as the previous predictor because $E(y|\boldsymbol{X})$ has not changed. Therefore, virtual drift is omitted in the following discussion. Next, we discuss how to effectively build and update the predictor when real drift ($E(y|\boldsymbol{X})$ changes) occurs.

1) The linear case: We first consider the simplest case where there is only one attribute variable in the feature space, denoted by X_t, and y_t is the label variable; y_t and X_t are linearly correlated, and X_t is a first-order autoregressive process. The sudden drift occurs at time point t_d.

In this linear case, $X_t = \beta_0 + \beta_1 X_{t-1} + \epsilon_t, |\beta_1| < 1$, and $y_t = \theta_0 + \theta_1 X_t + \epsilon_t$ (concept 1) before time point t_d, $y_t = \theta_0' + \theta_1' X_t + \epsilon_t$ (concept 2) after t_d where $\theta_0' \neq \theta_0$ and $\theta_1' \neq \theta_1$. Clearly, $\{X_t, y_t\}$ is a data stream with drift and temporal dependency in which a real drift occurring at t_d, and X_t is a first-order autoregressive process. According to Definition 12, this data stream can be written as $DS_t^{(X,y)} = \left\{MT_t^X, MT_t^y\right\}$, where MT_t^x is:

$$\begin{aligned} MT_t^x &= S_t^x \\ S_t^x &: X_t = \beta_0 + \beta_1 X_{t-1} + \epsilon_t \end{aligned} \tag{12}$$

and MT_t^y is:

$$\begin{aligned} MT_t^y &= \omega_t^1 S_t^{y,1} + \omega_t^2 S_t^{y,2} + \omega_t^3 S_t^{y,3} \\ S_t^{y,1} &: y_t = \theta_0 + \theta_1\beta_0 - \theta_0\beta_1 + \beta_1 y_{t-1} + \epsilon_t \\ S_t^{y,2} &: y_t = \theta_0' + \theta_1'\beta_0 - \theta_0'\beta_1 - \frac{\theta_0'\theta_1'\beta_1}{\theta_1} y_{t-1} + \epsilon_t \\ S_t^{y,3} &: y_t = \theta_0' + \theta_1'\beta_0 - \theta_0'\beta_1 + \beta_1 y_{t-1} + \epsilon_t \\ \boldsymbol{\omega}_t^{(1,2,3)} &= \begin{cases} (1,0,0) & t < t_d \\ (0,1,0) & t = t_d \\ (0,0,1) & t > t_d \end{cases} \end{aligned} \tag{13}$$

For concise notation, we use X, y to present the data stream. The component in X, y at different t is the same as in (12) and (13).

There are two ways to estimate y_t: a) The traditional approach is to find the optimal predictor in the hypothesis set $\mathcal{H} : \mathcal{X} \to \mathcal{Y}$; b) the optimal hypothesis can also be found in the hypothesis set $\mathcal{H} : (\mathcal{L}, \mathcal{Y}) \to \mathcal{Y}$ to estimate y_t. Next, we will discuss the difference between these two approaches when drift occurs.

The estimations of $\boldsymbol{\theta}$ or $\boldsymbol{\beta}$ and y_t computed by a) and b) before t_d are unbiased, consistent, and efficient estimations if the ordinary least squares (OLS) method is used. Similarly, the observations after viewing enough instances of the new pattern comprise the new training set, and the estimations of $\boldsymbol{\theta'}$ or $\boldsymbol{\beta}$ and y_t by a) or b) are also unbiased, consistent, and efficient.

The difference between a) and b) is represented when the training set ***mixes data from the old pattern (old concept) and the new pattern (new concept)***. Assume that the training set contains n observations of (X_t, y_t), where $n_1 = n - n_2$ of them are from the old concept, and n_2 of them are from the new concept. A predictor is trained with this training set to estimate future y_t which follows the new pattern.

Theorem 2. *Given a data stream $DS_t^{(\boldsymbol{X},y)} = \left\{MT_t^X, MT_t^y\right\}$ where MT_t^X and MT_t^y are as presented in* (12) *and* (13) *separately, the testing error linearly decreases to 0 as n_2/n increases (the training data contains more instances of new concept) if using $\mathcal{H} : \mathcal{X} \to \mathcal{Y}$ as the hypothesis set.*

Proof. The proof is given in Appendix B. □

Theorem 3. *Given a data stream $DS_t^{(\boldsymbol{X},y)} = MS_t^{(\boldsymbol{X},y)}$, the testing error exponentially decreases to 0 as n_2/n increases if using $\mathcal{H} : (\mathcal{L}, \mathcal{Y}) \to \mathcal{Y}$ as the hypothesis set.*

Proof. The proof is given in Appendix D. □

As the new instances arrive and are included in the training set, the n_2/n finally increases to 1. Clearly, the error decreases faster under the condition of Theorem 3 (exponentially decreases to 0) than that under the condition of Theorem 2 (linearly decreases to 0).

The conclusion of the unary case above is also suitable in the multiple linear case. If no collinearity exists in the multiple case, θ_i for $X^{(i)}$ (ith dimension of $\boldsymbol{X}$) is similarly computed to (23) but Xy needs to subtract $\boldsymbol{X}y$ of other dimensions. The y_t process still converges to a constant by constraint (2). Therefore, after drift occurs, as the training set contains more instances of the new pattern, the testing error of the predictor trained with the training set on the reconstructed space $(\mathcal{L}, \mathcal{Y})$ decreases faster than that of the predictor trained on the original space $(\mathcal{X}, \mathcal{Y})$.

2) A general case: The conclusion of the linear case can be applied in a general case by introducing locally weighted regression that the predictor trained on the reconstructed space is better than the predictor trained on the original space even when $\boldsymbol{X}$ and y have a non-linear correlation. *The locally weighted regression is an existing method, which is not the contribution in this paper. However, it helps us to widen the conclusion in the linear case to a general case.*

The learning objective of locally weighted regression at each target point $\boldsymbol{X}_u$ is represented as follows:

$$\underset{\boldsymbol{\theta}(\boldsymbol{X}_u)}{\arg\min} \sum_{t=1}^{N} K_k(\boldsymbol{X}_u, \boldsymbol{X}_t)(y_t - \boldsymbol{\theta}^T \boldsymbol{X}_u)^2, \tag{14}$$

where

$$K_k(\boldsymbol{X}_u, \boldsymbol{X}_t) = D\left(\frac{|\boldsymbol{X}_t - \boldsymbol{X}_u|}{b_k(\boldsymbol{X}_u)}\right). \tag{15}$$

$K_k(\boldsymbol{X}_u, \boldsymbol{X}_t)$ is the weight determined by the distance from $\boldsymbol{X}_u$ to the k-nearest neighborhoods of $\boldsymbol{X}_u$, $b_k(\boldsymbol{X}_u)$, and in this paper, D is defined as:

$$D(d) = \begin{cases} 1 & d \leq b_k(\boldsymbol{X}_0) \\ 0 & d < b_k(\boldsymbol{X}_0) \end{cases} \tag{16}$$

To build a predictor based on the current training set $(\boldsymbol{X}_t, y_t)$ to estimate y_u, we first find the k-nearest neighborhoods of $(\boldsymbol{X}_u, y_u)$, and learn a linear predictor on these neighborhoods. y_u is estimated by this local predictor. This converts the general case to a sum of linear cases which is discussed in Section IV-C1.

Remark 6. *For a data stream with concept drift and temporal dependency $DS_t^{(\boldsymbol{X},y)}$, this subsection gives a theoretical conclusion that when the training set mixes data from two patterns, the error of a predictor built on the reconstructed space $(\mathcal{L}, \mathcal{Y})$ decreases faster than that of a predictor built on the original space $(\mathcal{X}, \mathcal{Y})$ as the training set contains more instances from the new pattern.*

D. Drift adaptation procedure in DAR

Section IV-C concludes that the predictor built on the reconstructed space is better than the predictor built on the original space for a data stream with concept drift and temporal dependency. Once the space is reconstructed, the next problem is how to build and update the predictor on the reconstructed space to adapt to the newest pattern. This section discusses this drift adaptation procedure used in DAR. The drift adaptation is based on a proposed statistic LDD$^+$. In this section, we first introduce LDD$^+$ and then the adaptation procedure.

1) The local drift degree (LDD+): When the training set mixes with instances of different patterns, we consider the pattern of the upcoming future instances to be the new concept. According to Theorem 3, the more new concept instances are included, the more accurate it is to use a predictor built on this training set to estimate future instances. Therefore, updating the training set to include more new concept instances is also an important process for data streams with concept drift. If none of the instances in the training set follows the new pattern, it is impossible for any method to train an effective predictor to estimate future values. In this paper, we propose to update the training set based on the **local drift degree (LDD$^+$)**.

The original edition of LDD, proposed by Liu et al., is a statistic to quantify regional discrepancies between two different sample sets, and this discrepancy is compared to two thresholds to determine the drift area [38]. In this paper, LDD is improved from three aspects: 1) we redesign this statistic so that the limitation of sample size can be removed; 2) LDD is applicable to classification tasks while LDD$^+$ is to regression tasks; 3) we do not need thresholds to implement adaptation.

Given Δ_1 and Δ_2 two $m+1$-dimension populations from space $\mathbb{R}^{m+1}$, two samples of Δ_1 and Δ_2, δ_1 and δ_2, consist of instances from Δ_1 and Δ_2 respectively. As it is impossible to acquire all instances in Δ_1 and Δ_2, δ_1 and δ_2 are used to infer by statistical theories whether Δ_1 and Δ_2 have the same distribution. If Δ_1 and Δ_2 have the same distribution, the number of instances belonging to δ_1 and δ_2 in any arbitrary subspace $\varpi \subset \mathbb{R}^{m+1}$ are theoretically the same, which leads to an insignificant discrepancy between the number of instances belonging to δ_1 and δ_2. If Δ_1 and Δ_2 have different distributions, uneven density exists in at least one subspace. Based on the above idea, the original version of LDD is defined in (17) [38]:

$$d_\varpi = \frac{|\delta_{\varpi_2}|/n_{\delta_2}}{|\delta_{\varpi_1}|/n_{\delta_1}} - 1 \tag{17}$$

where $|\delta_{\varpi_1}|$ and $|\delta_{\varpi_2}|$ represent the number of instances in ϖ belonging to δ_1 and δ_2, and n_{δ_1} and n_{δ_2} are the sample size of δ_1 and δ_2 respectively.

When δ_1 is the current training set, and δ_2 is the newly arrived data instances, LDD can be used to measure the distribution difference of the training set and the newly arrived batch. If LDD is larger than a statistical threshold, the newly arrived instances are considered to have a different distribution from the training set, which denotes a drift. The original LDD assumes that $|\delta_{\varpi_1}|/n_{\delta_1}$ is a constant. However, this is not always true. To overcome this drawback, we propose LDD$^+$ as follows:

$$d_\varpi^+ = \frac{|\delta_{\varpi_1}|}{n_{\delta_1}} - \frac{|\delta_{\varpi_2}|}{n_{\delta_2}} \tag{18}$$

where $|\delta_{\varpi_1}|$, $|\delta_{\varpi_2}|$, n_{δ_1} and n_{δ_2} have the same meaning as in (17).

Theorem 4. *Given δ_1 and δ_2 have the same distribution, $d_\varpi^+ \sim N(0, S_{\delta_1}^2/n_{\delta_1} + S_{\delta_2}^2/n_{\delta_2})$, where $S_{\delta_1}^2$ are $S_{\delta_2}^2$ are the sample variances, and n_{δ_1} and n_{δ_2} are the sample size.*

Proof. The proof is given in Appendix E. □

Remark 7. *It is true that the value of* (18) *equals the value of* (17) *times a scalar, if we neglect assumptions required by* (17)*. However, they are completely different if we consider their distributions. In* (17)*, the denominator is assumed to be a constant. Only if this denominator ($|\delta_{\varpi_1}|/n_{\delta_1}$) is a constant, the statistic d_ϖ could be normal by the central limit theorem. However, this assumption is not true in real applications. As a result, LDD is sensitive to different n_{δ_1}.* (18) *considers $|\delta_{\varpi_1}|/n_{\delta_1}$ as a random variable, allowing the variation of n_{δ_1} and thus is more robust. In principle,* (17) *is a statistic for one-sample test, but* (18) *is for two-sample test.*

We conduct three groups of experiments to validate the impact of sample size on LDD and LDD$^+$ when they are applied to test distribution changes. The comparison results are shown in Table II. S_0 is a **200**-length-sample of data from $N(0,1)$. S_1 is a **2000**-length-sample and S_2 is a **200**-length-sample. In the experiments of *no-drift*, S_1 and S_2 are from $N(0,1)$. In the experiments of *mean drift*, S_1 and S_2 are from $N(1,1)$. In the experiments of *variance drift*, S_1 and S_2 are from $N(0,2)$.

In the no-drift case, LDD and LDD$^+$ achieves consistent results of not rejecting the null hypothesis of $H_0 : S_0 = S_1$ and $H_0 : S_0 = S_2$. In the drift cases, LDD$^+$ shows consistent testing results of rejecting the null hypothesis. However, LDD obtains inconsistent testing results given different sample sizes. This validates our claim that LDD$^+$ is more robust to the sample size.

TABLE II
COMPARISON EXPERIMENTS BETWEEN LDD AND LDD$^+$: AVERAGE P-VALUE OVER 100-TRIALS

Avg.p-value	LDD		LDD$^+$	
	$S_0 = S_1$	$S_0 = S_2$	$S_0 = S_1$	$S_0 = S_2$
no-drift	0.2138	0.1343	0.1388	0.1185
mean drift	0.0893$^+$	0.1483	0.000*	0.000*
variance drift	0.4323	0.0002*	0.0012*	0.0229*

Algorithm 1: Computation of LDD$^+$

Input : DS: the current training set
DS_w: the newly arrived batch of data
k: the number of nearest neighbors
Output: ldd$^+$

for $t = 1$ **to** $|DS_w|$ **do**
  $B = [DS, DS_w]$;
  $(knn_1, ..., knn_{k+1}) = knnsearch(DS_w, B, k+1)$; %find $k+1$ nearest neighbors of DS_w, $|\cdot|$ computes the cardinality
  $n_0 = |n_{i\in(1,k+1)} \in DS|$; %the number of neighbors in DS
  $n_1 = |n_{i\in(1,k+1)} \notin DS|$; %the number of neighbors in DS_w
  ldd$^+(t) = n_0/|DS| - n_1/|DS_w|$
end
return ldd$^+$

The original version of LDD is applied to a classification task where $|\delta_\omega|$ is computed based on the L2 norm of feature vectors given the same label that is $d(k) = ||\boldsymbol{X}_k - \boldsymbol{X}_0||_2^2$ when $y_k = y_0$ and $d(k) = \infty$ when $y_k \neq y_0$. In the regression task, $|\delta_\omega|$ is computed based on the distance of $||\boldsymbol{Z}_k - \boldsymbol{Z}_0||_2^2$ where $\boldsymbol{Z} = (\boldsymbol{X}, y)$ in this paper[2]. The computation process of LDD$^+$ is given in Algorithm 1. The returned LDD$^+$ values represent the relevance of instances in DS to the distribution of DS_w. A larger LDD$^+$ denotes that this instance is less similar/important to the new concept.

Clearly, LDD$^+$ could be used to detect whether drift occurs if we compare LDD$^+$ with a pre-assigned threshold. However, in this paper, we use LDD$^+$ to select the most important instances in the updated training set, and update the predictor on this training set. In this way, we do not need to involve the threshold parameter and guarantee the adaptation is informed at the same time.

2) The general procedure and pseudocode of DAR: According to the conclusion in Section IV-C, training a predictor on the reconstructed space is more effective than training it on the original space. One key problem is how to reconstruct the feature space, namely how to identify the p in (4). A larger p means more lag orders are involved in the predictor. If a large p is applied, the predictor will be complex and may induce the sparsity problem. However, valuable information is ignored if p is too small. The principle of p identification is to include orders that strongly affect the current state under the condition that the predictor will not be too complex.

p could be a pre-assigned parameter if there is prior information. In this paper, we do not assume there is available prior information. We use the Akaike information criterion (AIC) [39] to identify the lag order p. For each data stream, the identification of p is conducted on the historical data at the beginning. After this, the determined p will be used for this data stream without change.

[2] $\boldsymbol{Z}$ can also be in other forms of combinations of $\boldsymbol{X}$ and y such as a kernel function, which may improve the prediction accuracy.

Algorithm 2: The drift-adapted regression framework (DAR framework)

Input : DS_0: the historical data, the initial training set
DS_w: the newly arrived batch of data
k: the number of nearest neighbors
$\boldsymbol{\theta}$: required parameters in LWR
w: $w = |DS_w|$
Output: $\hat{y}_t, t = T+1, ...T+w$ %the estimated value

Process 1 **begin**
  for $i = 1 : d+1$ **do**
    for *j=1:3 % the max lag order is 3* **do**
      compute AIC(j, DS_0)
    end
    $p(i) = \arg\min_j \text{AIC}(j, DS_0)$ % determine the best lag order
  end
end
% now the input contains the lag orders
Process 3 **begin**
  $\boldsymbol{X} = []; y = []$
  for $i = T : T+w$ **do**
    $learnset = DS_0$
    if $isempty(ldd^+) \neq True$ **then**
      $DeleteIndex =$ ldd$^+$(ldd$^+(1:w) > w)$
      $learnset(DeleteIndex) = []$
    end
    Process 2 **begin**
      $\hat{y}_i = LWR(learnset, \boldsymbol{X}_i, \boldsymbol{\theta}, k)$
      $\boldsymbol{X} = [\boldsymbol{X}; \boldsymbol{X}_i]; y = [y; y_i]$
    end
  end
  $driftinsts = [\boldsymbol{X}, y]$
  $driftbase = DS_0$
  ldd$^+ =$ LDD$^+(driftinsts, driftbase)$ %Here, LDD$^+$ is for the function of this statistic, and ldd$^+$ is for its computed value.
  $DS_0 = [DS_0(w:end); (\boldsymbol{X}, y)]$
end
return $\hat{y}_t, t = T+1, ...T+w$

The AIC of a p-th autoregressive predictor trained on a N-size data sample is computed as:

$$AIC(p) = 2k - 2ln(\hat{\ell}(p)), \tag{19}$$

where k is the number of free parameters to be estimated (for example, here it is p+1), and ℓ is the likelihood function to estimate the parameter vector $\boldsymbol{\phi} = (c, \phi_1, ..., \phi_p)$ in (4).

$$\ell(p) = f_{Y_p,...,Y_1}(y_p, ...y_1; \boldsymbol{\phi}) \times \prod_{t=p+1}^{N} f(y_t|y_{t-1}, ..., y_{t-p}; \boldsymbol{\phi}). \tag{20}$$

For an autoregressive process as in (4) where p is unknown but assumed to be less than a pre-assigned value P, using AIC to determine p is to find $p \in (1, P)$ with minimum AIC from a collection of predictor AR(1),...,AR(P).

$$\widetilde{p} = \arg\min_{p\in(1,P)} \text{AIC}(p) \tag{21}$$

The time series identification of each feature is conducted on the historical data. Once the value of p is determined, it will not change. The maximum p is 3 in this paper. We reconstruct the original space to $(\mathcal{X}, \mathcal{Y}, \mathcal{L})$ using this process.

Combining the process of reconstructing space and LDD$^+$-based adaptation, we have the DAR framework as is shown in Figure 1. The pseudo code of the DAR framework is given in Algorithm 2. Before a new batch of instances arrives, the

historical data $\{DS_t^{(\boldsymbol{X},y)}|t=1,...,T\}$ is the initial training set. During Process 1, the lag orders, X_{t-p}, y_{t-p} are added to the system as new attributes. Process 2 conducts the locally weighted regression algorithm as given in Algorithm 3 in Appendix F. The rules for adaptation are as follows: 1. The newly arrived batch of w instances represents the newest pattern, so they will be added to the input set during updating and the oldest w instances in the input set will be deleted; 2. The relevance of old instances to the new pattern is measured by ldd$^+$. 3. If the remaining $(T-w)$ old instances in the input set have larger ldd$^+$ than the wth largest ldd$^+$, they will also be deleted from the training set. This process is realized in Process 3 in the DAR algorithm.

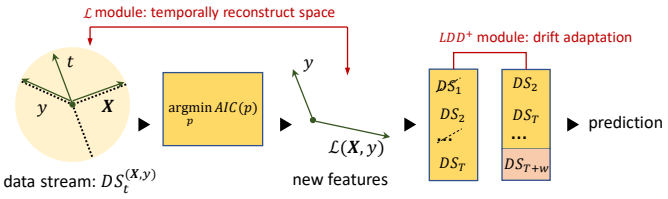


Fig. 1. The flowchart of the DAR framework. DAR have two modules: the $\mathcal{L}$ module aims to handle the temporal dependency, and the LDD$^+$ module handles concept drift. The data stream is mapped into a reconstructed space. After that, we conduct LDD$^+$-based adaptation on this space, and obtain real-time prediction results.

V. Experiment Evaluation

In this section, the effectiveness of the proposed DAR will be proved on both synthetic data and real data. In Section V-A, we explained the experimental design. The experiments for synthetic data are detailed in Section V-B, and the experiments for real data are detailed in Section V-C. Corresponding statistical tests are detailed in Section V-D.

A. Experiment design

We validate the effectiveness of DAR in terms of three aspects: 1) DAR on 1-dimensional linear case. This corresponds to the theoretical conclusion in Section IV-C1. As the 1-dimensional case is available for graphic presentation, the error will be given for every tested instance to show that the error decreases faster on the reconstructed space and therefore improves the adaptation performance; 2) DAR on multi-dimensional non-linear cases. This corresponds to the theoretical conclusion in Section IV-C2. It also validates our assumption that DAR performs better if the data stream $DS_t^{(\boldsymbol{X},y)}$ (Definition 12) has larger s; 3) DAR is compared to other drift adaptation methods. DAR uses different parameters for synthetic data and real-world data, which will be specified in each subsection. The organization of the experiments is detailed in Table III. Two criteria are used for evaluation, mean absolute error (MAE) and mean absolute percentage error (MAPE).

B. Experiments on synthetic data

In this paper, we test our method on synthetic data containing sudden drift and incremental drift. There are two kinds of synthetic data: 1-dimensional linear data (contain sudden and incremental drift), and multi-dimensional non-linear data (only contains sudden drift).

1) Generation of the synthetic data:

- **Simple linear data (the feature variable is 1-dimensional)**

Three data were generated by several parameter-changing linear models, in a similar way to the generation procedure of synthetic data in [27], [40]: *Non-Drift* is a data stream with no drift; *Sudd-Drift* contains a real sudden drift; *Incr-Drift* means that real incremental drift occurs over a period. The generated data are presented in Figure 2. The data were generated as follows:

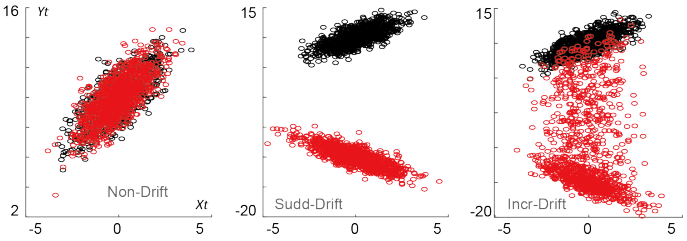


Fig. 2. Generated 1-dimensional linear data. The data with real sudden drift and real incremental drift are generated.

Non-Drift. Two random samples x_1 and x_2 were drawn from a normal distribution $\mathcal{N}(\mu,\sigma^2)$ where $\mu=10,\sigma^2=100$ as the first two values for the input. The rest of the input was generated by $x_t=\beta_1 x_{t-1}+\beta_2 x_{t-2}+\eta_t$ where $\{\eta_t\}$ is a random error series. We create 2002 values of x (including x_1 and x_2) with $\beta_1=0.5,\beta_2=-0.2$ and delete x_1 and x_2 from the data. These 2000 samples are denoted by $X_t=\{x_3,x_4,\ldots,x_{2002}\}$. The output series was generated by $Y_t=\theta_0+\theta_1 X_t+\epsilon_t$ where $\{\epsilon_t\}$ is a random error series, and $\theta_0=10,\theta_1=1$. The data is evidently time-dependent.

Sudd-Drift The first 998 data samples were generated in the same way as is in the Non-Drift samples. The subsequent 1002 samples were generated in the same way as the Non-Drift samples but with the parameters $\theta_0=-10,\theta_1=-1$. Real sudden drift occurred at the 999th sample.

Incr-Drift The first 998 data samples were generated in the same way as is in the Non-Drift samples. The input of the subsequent 1002 samples were the same as the Non-Drift samples. The output of the 999th to 1498th samples was generated by $\{\theta_0+f_0(t)\}+\{\theta_1+f_1(t)\}X_t+\epsilon_t$ where $f_i(t)=(\theta_i'-\theta_i)\times\frac{t-1000}{500}$. The output of the last 500 samples were generated by $Y_t=\theta_0'+\theta_1' X_t+\epsilon_t$. The parameters were $\theta_0=10,\theta_0'=-10,\theta_1=1$ and $\theta_1'=-1$. An incremental drift occurred from the 999th to 1498th samples, and the new pattern proceeded after the 1499th sample.

It should be noted that, although Non-Drift, Sudd-Drift, and Incr-Drift have a similar generating process, they have different samples. This is because x_1 and x_2 are randomly drawn for each data, and η_t and ϵ_t have different values for each data.

- **Multi-dimensional non-linear data (only considering sudden drift)**

TABLE III
EXPERIMENT DESIGN

Section	Data	Experimental aim	Main Results
Section V-B3	d1	Validate that the error decreases faster on a restructured space on simple linear cases (corresponding to Section IV-C1)	Table V, Figure 3
Section V-B3	d2	Validate DAR on multi-dimensional non-linear cases (corresponding to Section IV-C2)	Table VI,Table VII, Figure 4
Section V-C	d3	Compare DAR with the-state-of-the-art drift adaptation methods	Table VIII
Section V-D	d3	Statistical test for comparison between methods	Table IX

d1 contains three data generated in Section V-B1: 1-dimensional linear data
d2 contains six data generated in Section V-B1: multi-dimensional non-linear data
d3 real-world data

We generate the multi-dimensional non-linear data by referring to the Python package [41] and paper [42]. The original data is not used to validate the concept drift problem. In this paper, we use similar mapping functions to the mapping functions used in [41]. Drift is added by using a negative label variable after the drift point, and the time dependency is added by using autoregressive feature variables instead of the temporally independent feature variables in the original version. Six data streams are generated, and the details of the generation process are explained as follows:

Step 1: Generate five feature variables $X_{1,t}, X_{2,t}, \ldots, X_{5,t}$, and each of the feature variables is generated in the same way as X_t in *Non-Drift*. Namely, we have five AR(2) time series.

Step 2: Normalize each feature variable generated in Step 1. This is because the mapping function in [41] requires the feature variables on the interval $[0, 1]$.

Step 3: Generate the label variable using the mapping function in (22). Six groups of parameters are used to generate six data. They are listed in Table IV.

$$y_t = \theta_1 \sin(\pi X_{1,t} X_{2,t}) + \theta_2 (X_{3,t} - 0.5)^2 + \theta_3 X_{4,t} + \theta_4 X_{5,t} + \theta_5 \epsilon_t \quad (22)$$

Step 4: Drift is added by letting $y_{t>1000} = -y_{t>1000}$. Clearly, a sudden drift occurs at $t = 1001$ for all these six data.

TABLE IV
PARAMETERS FOR GENERATING DATA IN (22)

	θ_1	θ_2	θ_3	θ_4	θ_5
Para0	10	20	10	5	1
Para1	10	10	15	10	40/35
Para2	5	10	15	15	40/35
Para3	1	2	15	15	32/35
Para-1	10	20	2	1	23/35
Para-2	10	20	0	0	20/35

Remark. This mapping function presents a non-linear relationship of polynomial and sine transforms where θ_1 and θ_2 control their weights separately. In addition, it contains linear relationships where θ_3 and θ_4 control their weights separately. Para0 is the parameters used in [41]. Compared to Para0, Para1, Para2 and Para3 have more weights on the linear relationship while Para1 and Para2 have less weights on the linear relationship. Para2 does not have the term of a regular temporally dependency as θ_3 and θ_4 are 0. We design these parameters to validate the claim that our method will perform better with a larger s in $DS_t^{(\boldsymbol{X},y)}$ in Definition 12. The value of θ_5 is not 1 except for Para0. This is because the max absolute values of y in other data are not the same as those in Para0, which is 35. Therefore, θ_5 changes to alleviate the difference caused by the disturbance when the estimation results on these six data are compared.

2) Preassigned parameters: In the experiments of synthetic data, the first 500 instances were set as the historical data for the first training, namely $N = 500$. The max lag order P is 3, the length of the windows (w) is 100, and the number of neighbors (K) is 50. The parameters of SVR and tree models are the default in MATLAB (listed in Appendix G).

3) Results of the synthetic data:

- **Simple linear cases (X_t is 1-dimensional)**

We conduct the ablation study to validate the effectiveness of each module in DAR. The results are shown in Table V. The models tested are

- DAR w/o LDD$^+$&$\mathcal{L}$ is DAR without $\mathcal{L}$ and LDD$^+$ modules i.e., a plain locally weighted regression model;
- DAR w/o LDD$^+$ is DAR without LDD$^+$ module. Namely, DAR w/o LDD$^+$ can only handle temporal dependency;
- DAR w/o $\mathcal{L}$ is DAR without $\mathcal{L}$ module. DAR w/o $\mathcal{L}$ only handle concept drift;
- DAR-linear is our framework with a linear base learner.
- DAR-SVR is with a SVR base learner;
- DAR-tree is with a tree base learner.

The accuracy results in Table V indicate that: 1) LDD$^+$-based adaptation can solve the drift problem in data streams; 2) Reconstructed space helps to improve the drift adaptation process. The accuracy of DAR w/o LDD$^+$&$\mathcal{L}$ is as the same bad as DAR w/o LDD$^+$ but the MAE of DAR-linear is 1.7392 for SuddDrift and 1.4387 for IncrDrift which is much less than the MAE of DAR w/o $\mathcal{L}$. This verifies the discussion in Section IV-C that reconstructed space improves drift adaptation for time-dependent data when the training set mixes samples from different patterns. The final results are not much different from the linear case, indicating DAR's robustness to base learners.

Next, we verify that reconstructed space improves drift adaptation because the error decreases faster. Figure 3 shows the complete estimation process of DAR w/o $\mathcal{L}$ and DAR-linear. In general, DAR-linear chases the various trends faster than DAR w/o $\mathcal{L}$. Subplot A is drawn before drift occurs, where the estimation of both methods is accurate. In contrast, when drift starts to appear in subplot B, DAR-linear gradually chases the new trend while DAR w/o $\mathcal{L}$ remains in the old

TABLE V
MAE RESULTS OF ABLATION STUDY ON DIFFERENT MODULES OF DAR (SIMPLE LINEAR CASES).

Data Streams	DAR w/o LDD$^+$&$\mathcal{L}$	DAR w/o LDD$^+$	DAR w/o $\mathcal{L}$	DAR-linear	DAR-SVR	DAR-tree
NonDrift	0.8017	0.8029	0.8055	0.8079	0.8231	1.0638
SuddDrift	13.0991	13.2057	4.7669	1.7392	1.9061	1.8015
IncrDrift	9.542	9.315	4.5388	1.4387	1.2346	1.8967

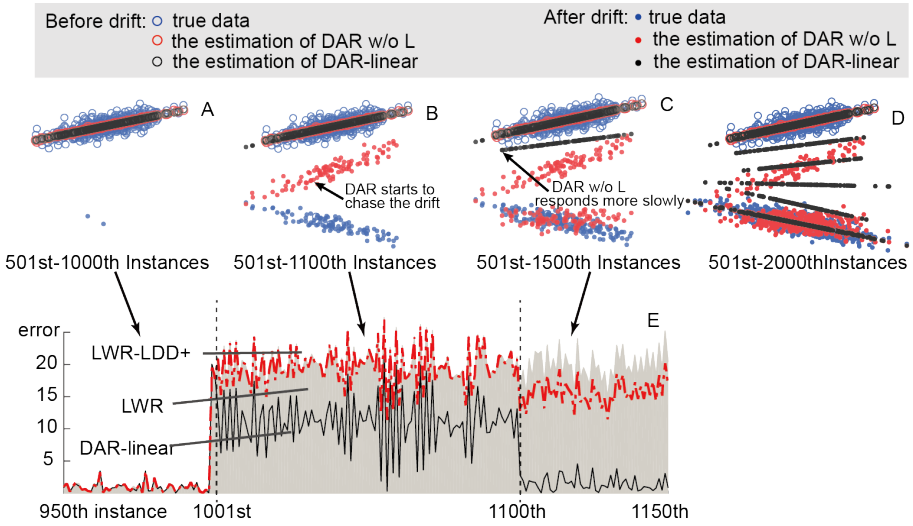


Fig. 3. Error decreasing process. The circles represent the scatter plots of inputs and outputs before drift occurs, while the dots show the results after drift has occurred. Colors denote the different estimation results: blue is for real values, red is for DAR w/o $\mathcal{L}$, and black is for DAR-linear. The testing errors of 950th-1150th instances are given in subplot E, where the gray shadow represents DAR w/o LDD$^+$&$\mathcal{L}$, the red dotted line represents DAR w/o $\mathcal{L}$ and the black line represents DAR-linear.

pattern, as all the black dots are still near to the circles denoting the old pattern. In subplot C, DAR-linear has already adapted to the new concept after the 1101st point, but DAR w/o $\mathcal{L}$ has only just started to adapt to the new concept. Subplot D shows that DAR w/o $\mathcal{L}$ finally chases the new concept after the 1501st instance which has 400 points delay than DAR-linear.

The testing error of each instance from 950th to 1150th is given in subplot E. As the 950th-1000th instances are estimated by the predictor trained by instances before the 950th instance, there is no instance from the new pattern in the training set, and the testing error for the 999th and 1000th instance is very high. After the arrival of the 1000th instance, the training set is updated and contains two instances from the new pattern, namely the 999th and 1000th instance. The training set is now a mixture of instances from the old and new patterns. If DAR w/o $\mathcal{L}$ is applied, the effectiveness of the adaptation is subtle and the testing error of the 1001st-1100th instances is still as high as in the non-adaptation predictor. However, when DAR-linear is applied, the testing error clearly decreases. After the 1100th instance has been obtained, the training set is updated again. The testing error of DAR-linear is now as low as it was before the drift occurred, which means that DAR-linear has already adapted to the new pattern. In contrast,DAR w/o $\mathcal{L}$ has only just started to adapt.

So far, we have validated DAR on a simple linear regression for data stream with drift and temporal dependence. We have shown the estimated value of the label variable at each time point before and after drift occurs because the simple linear regression is suitable for the graphic presentation. In the next section, a general case of multi-dimensional non-linear data is presented, where the average accuracy is used for validation.

- **Multi-dimensional non-linear cases**

In this section, DAR is validated in six multi-dimensional non-linear cases. The results of the ablation study are listed in Table VI (MAE) and Table VII (MAPE). According to Table VI and Table VII, the performance of each module in DAR shows consistent effectiveness to that in a simple linear case. These results strengthen the conclusion in a simple linear case that DAR can handle data streams with drift and temporal dependency. In addition, we found that replacing the linear predictor with other non-linear predictors such as SVR and tree in DAR may have better prediction results for the data with large weights on the non-linear terms. For example, in Table VI, the MAE of DAR-tree is much smaller than the MAE of DAR-linear on data Para2 which only contains non-linear terms.

We also found that compared to adaptation on the original feature space, the improvement of DAR increases if the data is generated with larger weights of the linear term. This validates our claim that *'for an arbitrary data stream* $DS_t^{(\boldsymbol{X},y)}(s,\cdot)$ *(Definition 12), a bigger* s *means a better estimation by DAR'*. To clearly present this, we compute the improvement percentage of DAR-linear compared to DAR w/o $\mathcal{L}$, and present the results in Figure 4. The x-axis starts from Para2 and ends with Para3 which is arranged in an increased order

TABLE VI
MAE RESULTS OF ABLATION STUDY ON DIFFERENT MODULES OF DAR (MULTI-DIMENSIONAL NON-LINEAR CASES).

Data Streams	DAR w/o LDD$^+$&$\mathcal{L}$	DAR w/o LDD$^+$	DAR w/o $\mathcal{L}$	DAR-linear	DAR-SVR	DAR-tree
Para0	20.1489	18.4213	8.2653	5.3583	4.7753	3.5539
Para1	24.2418	24.606	9.0504	5.66	5.2406	4.7441
Para2	25.1195	24.6822	8.7878	4.3891	4.6255	4.3482
Para3	20.8337	19.5359	6.8099	2.8586	3.9276	2.8313
Para-1	11.9175	11.8405	5.5357	4.4129	3.9263	2.5419
Para-2	10.2588	10.5276	4.8849	4.1026	3.6865	2.2622

TABLE VII
MAPE RESULTS OF ABLATION STUDY ON DIFFERENT MODULES OF DAR (MULTI-DIMENSIONAL NON-LINEAR CASES).

MAPE	DAR w/o LDD$^+$&$\mathcal{L}$	DAR w/o LDD$^+$	DAR w/o $\mathcal{L}$	DAR-linear	DAR-SVR	DAR-tree
Para0	142.53%	129.45%	61.41%	42.68%	36.85%	25.98%
Para1	138.59%	140.90%	52.96%	35.55%	32.16%	29.03%
Para2	133.36%	130.66%	48.71%	25.41%	26.13%	25.30%
Para3	130.28%	119.73%	43.77%	24.73%	28.08%	19.01%
Para-1	175.37%	173.89%	88.64%	77.18%	62.29%	34.93%
Para-2	312.37%	323.73%	205.97%	215.42%	161.17%	42.14%

of s. Each dot in Figure 4 is computed by subtracting the accuracy of DAR-linear from that of DAR w/o $\mathcal{L}$ and then divided by the accuracy of DAR w/o $\mathcal{L}$. For example, the value of the first blue dot is computed as $(4.8849-4.1026)/4.8849$, which is 16.01%.

It should be *noted* that both DAR w/o $\mathcal{L}$ and DAR-linear use linear predictors. Therefore, the difference between these two methods is not caused by the increased linear relationship between $\boldsymbol{X}$ and y. The improvement clearly exists because there is a larger proportion of regular temporal dependency in the data stream. During the process of generating these six data, the temporal dependency on $\boldsymbol{X}$ disappears when this term involves non-linear transforms. Therefore, using larger weights on the linear terms in (22) corresponds to the case of bigger s in Definition 12. These experimental results explain why we define the mixed time series for data streams.

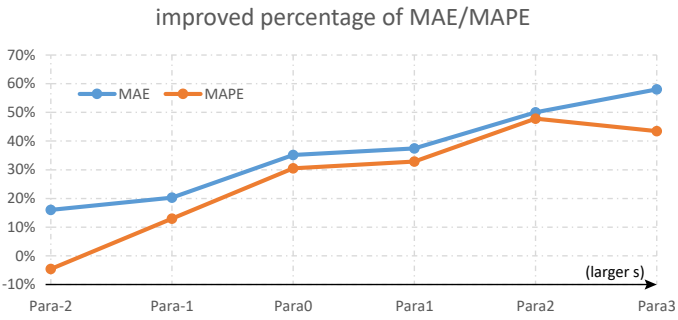


Fig. 4. The improved percentage of MAE/MAPE of DAR-linear from DAR w/o $\mathcal{L}$. Both methods have the drift adaptation process. The difference is that DAR-linear is implemented on the reconstructed space.

Discussion: According to the synthetic data experiments, we conclude that: 1) feature space reconstruction will not obtain a worse estimation when there is no drift in the data; 2) the predictor trained on reconstructed feature space ties with the predictor on the original space if the training set only contains a single pattern; 3) when the data has drift and temporal dependency, and the training set mixes data examples from different patterns, space reconstruction enables faster adaptation to the new pattern; 4) adaptation by LDD$^+$ is effective for data streams with concept drift; 5) the effectiveness of reconstruction and adaptation by LDD$^+$ is not affected by the type of predictors; 6) DAR performs better for the data streams with strong temporal dependency.

C. Experiments on real-world data streams

In this subsection, we test the DAR framework on seven real-world data streams. Seven techniques to tackle regression drift are introduced as baselines. One baseline is DAR w/o LDD$^+$&$\mathcal{L}$ which is the non adaptation version of DAR-linear, and the other six are: FIMT-DD [43], ORTO [44], AMR and its ensemble version, metaAMR [15], Perceptron [45] and FUZZ-CARE [27]. The previous five baselines are implemented by MOA [46](https://moa.cms.waikato.ac.nz/) and FUZZ-CARE is implemented by the code in (https://github.com/songyiliao/FUZZ-CARE). The following three subsections present the data description, preassigned parameters, and experiment results of the data streams.

1) Data description: There are seven data streams, taken from five data sources, listed below.

CCPP: The data contains 47, 840 data points collected from a Combined Cycle Power Plant over six years (2006-2011), when the power plant was set to work at full load. Features consist of the hourly average ambient variables temperature (T), ambient pressure (AP), relative humidity (RH) and exhaust vacuum (V) to predict the net hourly electrical energy output (EP) of the plant. It is available from the UCI machine learning repository (http://archive.ics.uci.edu/ml).

Sensor: The data contains 2,219,803 consecutive records of temperature, humidity, light and sensor voltage collected from 54 sensors deployed in the Intel Berkeley Research Lab over a two-month period. To make it a regression task, we selected the records of four sensors—sensors 3, 8, 20, 46–located in different parts of the Lab, and used temperature, humidity, and light as features, and sensor voltage as the dependent variable. The light during working hours is generally stronger

than during the night, and the temperature of specific sensors may rise regularly during meetings. This is available in [47].

SMEAR: This is a 30-minute interval environment observation dataset collected from the SMEAR II station which contains 140,576 instances of 43 variables from 0:15 on 15th April 2005 to 23:45 on 14th April 2013. The regression task based on this data is to predict solar radiation using 37 variables, as six variables are time labels which will not be considered as prediction features in the model. The remaining environmental features have been introduced in [48].There are many missing values in this dataset. In this paper, we eliminate the missing values in the same way as [48].

Solar is provided by NASA and contains 32,686 records of meteorological data from the HI-SEAS weather station from 23:55:26 29 September 2016 to 00:00:02 1 December 2016 (Hawaii time) in the period between Mission IV and Mission V. The data interval is roughly 5 minutes. The input features are temperature (unit: degrees Fahrenheit), humidity (unit: percent), barometric pressure (unit: Hg), wind direction (unit: degree), wind speed (unit: miles per hour) and the label variable is solar radiation (unit: watts per meter^2). The dataset is available at https://www.kaggle.com/dronio/SolarEnergy/data.

2) Preassigned parameters: There are four preassigned parameters: length of training set, max lag order, length of windows, and the number of neighbors. All the experiments use a parameter combination as follows: the length of the training set (N) is 2000, the max lag order (P) is 3, the length of the windows (w) is 200, and the number of neighbors (K) is 50. The parameters of SVR and tree models are the default in MATLAB (listed in Appendix G).

3) Results of real data streams: The MAE and corresponding rank of the DAR framework and other methods are listed in Table VIII, where a lower rank means better performance. The accuracy rank of each method shows that DAR-based methods perform better than the other methods. The variables are very likely to exhibit the temporal dependency problem, and the overwhelming good performance of DAR-based methods demonstrates that our proposed DAR framework is an effective adaptation framework for data streams with drift and temporal dependency.

D. Friedman Test and Post-hoc Test after Conover on Data Streams

In this section, we use the Friedman test and its Post-hoc test after Conover [49] to validate whether our proposed method is significantly better than the baselines. These two statistical tests are conducted on the average MAE of different methods among all the data streams. We conduct Friedman test and the post-hoc test among methods based on their performance listed in Table VIII. The test results are shown in Table IX(only the results of the DAR-linear in the post-hoc test part). $R_i - R_{DAR}$ is the difference between the rank sums of other baselines and DAR-linear, and the p-value tests the significance of this difference.

The test results show that the prediction accuracy of the various methods is different and DAR-linear is significantly better than the other baselines.

VI. Conclusion and Future Studies

Learning in a non-stationary environment is a big challenge for current machine learning methods. Existing related studies mainly focus on data streams with concept drift problems, breaking the assumption of identical distribution. However, few studies have discussed the problem of temporal dependency, which makes the independence assumption invalid.

To fill this gap, this paper conducted a theoretical study for the regression of data streams with concept drift and temporal dependency, and based on this study proposed a drift-adapted regression (DAR) framework, to predict non-stationary data streams. The DAR framework is able to deal with drift and the temporal dependency problem simultaneously. The experiment evaluation on synthetic data verifies the theoretical conclusion, and the experiments on real data streams illustrate several advantages of the proposed DAR framework.

Based on the findings in this paper, we suggest that future studies in this field should include 1) multi-output regression of data streams with concept drift and temporal dependency; 2) consideration of a more complex time series than an autoregressive process.

Acknowledgments

The work presented in this paper was supported by the Australian Research Council (ARC) under Discovery Grant DP190101733 and FL190100149.

References

[1] A. Haque, L. Khan, M. Baron, B. Thuraisingham, and C. Aggarwal, "Efficient handling of concept drift and concept evolution over stream data," in *IEEE 32nd International Conference on Data Engineering*. Helsinki, Finland, May. 16-20: IEEE, 2016, pp. 481–492.

[2] Z. Yang, S. Al-Dahidi, P. Baraldi, E. Zio, and L. Montelatici, "A novel concept drift detection method for incremental learning in nonstationary environments," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 1, pp. 309–320, 2020.

[3] J. Lu, A. Liu, Y. Song, and G. Zhang, "Data-driven decision support under concept drift in streamed big data," *Complex & Intelligent Systems*, vol. 6, no. 1, pp. 157–163, 2020.

[4] K. Malialis, C. G. Panayiotou, and M. M. Polycarpou, "Online learning with adaptive rebalancing in nonstationary environments," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2020.

[5] J. Gama, R. Sebastião, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Machine learning*, vol. 90, no. 3, pp. 317–346, 2013.

[6] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys*, vol. 46, no. 4, p. 44, 2014.

[7] H. M. Gomes, J. P. Barddal, F. Enembreck, and A. Bifet, "A survey on ensemble learning for data stream classification," *ACM Computing Surveys*, vol. 50, no. 2, p. 23, 2017.

[8] I. Žliobaitė, A. Bifet, J. Read, B. Pfahringer, and G. Holmes, "Evaluation methods and decision theory for classification of streaming data with temporal dependence," *Machine Learning*, vol. 98, no. 3, pp. 455–482, 2015.

[9] Y. Song, J. Lu, A. Liu, H. Lu, and G. Zhang, "A segment-based drift adaptation method for data streams," *IEEE transactions on neural networks and learning systems*, 2021.

[10] Y. Sun, K. Tang, Z. Zhu, and X. Yao, "Concept drift adaptation by exploiting historical knowledge," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, pp. 4822–4832, 2018.

[11] M. Pratama, J. Lu, E. Lughofer, G. Zhang, and M. J. Er, "An incremental learning of concept drifts using evolving type-2 recurrent fuzzy neural networks," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 5, pp. 1175–1192, 2017.

TABLE VIII
MAE COMPARISONS BETWEEN DIFFERENT METHODS ON REAL-WORLD DATA STREAMS. THE DAR METHODS OUTPERFORM THE COMPARED METHODS ON DATA STREAMS WHERE CONCEPT DRIFT OCCURS.

MAE/Rank	CCPP	Sensor3	Sensor8	Sensor20	Sensor46	SMEAR	Solar	Avg. Rank
DAR w/o LDD$^+$&$\mathcal{L}$	3.6E+00	2.9E-01	6.5E-02	1.9E-01	2.2E-01	1.8E+01	1.9E+02	
	5	10	9	7	9	6	9	7.86
FIMT-DD	3.6E+00	7.1E-03	9.7E-03	8.0E-01	1.6E-01	2.3E+01	1.1E+02	
	4	4	7	9	6	8	7	6.43
ORTO	4.5E+02	6.6E-02	1.7E-01	9.6E-01	4.0E-01	3.4E+01	2.3E+02	
	10	9	10	10	10	9	10	9.71
AMR	3.4E+00	7.7E-03	6.6E-03	8.2E-03	2.0E-01	1.4E+01	9.5E+01	
	3	5	3	5	8	5	6	5.00
metaAMR	3.3E+00	1.6E-02	7.3E-03	1.1E-02	1.7E-01	2.0E+01	9.4E+01	
	1	8	4	6	7	7	5	5.43
Perceptron	3.7E+00	6.9E-03	6.0E-03	7.9E-01	1.6E-01	3.8E+01	1.3E+02	
	7	3	1	8	5	10	8	6.00
FUZZ-CARE	5.6E+00	1.6E-02	1.7E-02	7.9E-03	5.3E-02	1.0E+01	8.7E+01	
	9	7	8	4	4	3	4	5.57
DAR-linear	3.6E+00	5.1E-03	6.4E-03	3.9E-03	5.0E-03	9.9E+00	4.0E+01	
	6	1	2	1	1	2	2	2.14
DAR-SVR	3.3E+00	5.9E-03	7.8E-03	4.6E-03	5.8E-03	8.9E+00	3.1E+01	
	2	2	5	2	2	1	1	2.14
DAR-tree	4.7E+00	7.7E-03	9.4E-03	6.9E-03	7.7E-03	1.1E+01	4.6E+01	
	8	6	6	3	3	4	3	4.71

TABLE IX
FRIEDMAN TEST AND ITS POST-HOC TEST OF ALL THE METHODS OVER ALL SEVEN DATA STREAMS, WHERE "FRIEDMAN TEST" IS THE RESULT FOR FRIEDMAN TEST AND "FRIEDMAN - POST-HOC TEST AFTER CONOVER" IS FOR THE PAIRWISE COMPARISON. "*", "**", AND "***" MEANS THIS VALUE IS SIGNIFICANT AT THE LEVEL OF 0.05, 0.01 AND 0.001 RESPECTIVELY. "DF" DENOTES THE FREEDOM DEGREE.

Friedman Test	χ^2_R	χ^2_{pvalue}	df
	36.5376	3.18E-05***	7

Friedman - post-hoc test after Conover							
DAR-linear	DAR w/o LDD$^+$&$\mathcal{L}$	FIMTDD	ORTO	AMR	metaAMR	Perceptron	FUZZ-CARE
$R_i - R_{DAR}$	40	30	53	20	23	27	24
$\text{diff}_{\text{pvalue}}$	0.000***	0.0013**	0.000***	0.020*	0.009**	0.003**	0.007**

[12] L.-Y. Wang, C. Park, K. Yeon, and H. Choi, "Tracking concept drift using a constrained penalized regression combiner," *Computational Statistics & Data Analysis*, vol. 108, pp. 52–69, 2017.

[13] R. Bakirov, B. Gabrys, and D. Fay, "On sequences of different adaptive mechanisms in non-stationary regression problems," in *28th International Joint Conference on Neural Networks*. Killarney, Ireland, Jul.12-17: IEEE, 2015, pp. 1–8.

[14] F. Dong, J. Lu, Y. Song, F. Liu, and G. Zhang, "A drift region-based data sample filtering method," *IEEE Transactions on Cybernetics*, pp. 1–14, 2021.

[15] J. Duarte, J. Gama, and A. Bifet, "Adaptive model rules from high-speed data streams," *ACM Transactions on Knowledge Discovery from Data*, vol. 10, no. 3, p. 30, 2016.

[16] B. Krawczyk and A. Cano, "Adaptive ensemble active learning for drifting data stream mining," in *International Joint Conference on Artificial Intelligence*, Macao, China, Aug. 10-16, 2019, pp. 2763–2771.

[17] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, 2018, doi:10.1109/TKDE.2018.2876857.

[18] L. Bu, C. Alippi, and D. Zhao, "A pdf-free change detection test based on density difference estimation," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 2, pp. 324–334, 2018.

[19] A. Haque, L. Khan, and M. Baron, "SAND: Semi-supervised adaptive novel class detection and classification over data stream," in *the 30th AAAI Conference on Artificial Intelligence*, Phoenix Arizona, USA, Feb. 12-17, 2016, pp. 1652–1658.

[20] L. Rutkowski, M. Jaworski, L. Pietruczuk, and P. Duda, "A new method for data stream mining based on the misclassification error," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 5, pp. 1048–1059, 2015.

[21] J. Carmona and R. Gavaldà, "Online techniques for dealing with concept drift in process mining," in *the 11th International Symposium on Intelligent Data Analysis*, vol. 7619 LNCS, Helsinki, Finland, Oct. 25-27, 2012, pp. 90–102.

[22] A. Liu, J. Lu, and G. Zhang, "Concept drift detection via equal intensity k-means space partitioning," *IEEE Transactions on Cybernetics*, 2020.

[23] D. M. dos Reis, P. Flach, S. Matwin, and G. Batista, "Fast unsupervised online drift detection using incremental kolmogorov-smirnov test," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco California, USA, Aug. 13-17: ACM, 2016, pp. 1545–1554.

[24] H. Yu, J. Lu, and G. Zhang, "Topology learning-based fuzzy random neural network for streaming data regression," *IEEE Transactions on Fuzzy Systems*, 2020.

[25] ——, "An online robust support vector regression for data streams," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020.

[26] ——, "Continuous support vector regression for nonstationary streaming data," *IEEE Transactions on Cybernetics*, pp. 1–14, 2020.

[27] Y. Song, J. Lu, H. Lu, and G. Zhang, "Fuzzy clustering-based adaptive regression for drifting data streams," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 3, pp. 544–557, 2019.

[28] G. Ditzler and R. Polikar, "Incremental learning of concept drift from streaming imbalanced data," *IEEE transactions on knowledge and data engineering*, vol. 25, no. 10, pp. 2283–2301, 2013.

[29] S. Wang, L. L. Minku, and X. Yao, "A systematic study of online class imbalance learning with concept drift," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, pp. 4802–4821, 2018.

[30] Y. Lu, Y.-m. Cheung, and Y. Y. Tang, "Dynamic weighted majority for incremental learning of imbalanced data streams with concept drift." in *the 26th International Joint Conference on Artificial Intelligence*, Melbourne, Australia, Aug.19-25, 2017, pp. 2393–2399.

[31] N. Lu, J. Lu, G. Zhang, and R. L. De Mantaras, "A concept drift-tolerant case-base editing technique," *Artificial Intelligence*, vol. 230, pp. 108–133, 2016.

[32] Y. Song, G. Zhang, H. Lu, and J. Lu, "A noise-tolerant fuzzy c-means based drift adaptation method for data stream regression," in *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. New Orleans, Louisiana, USA, Jun. 23-26: IEEE, 2019, pp. 1–6.

[33] B. Dong, Y. Li, Y. Gao, A. Haque, L. Khan, and M. M. Masud, "Multistream regression with asynchronous concept drift detection," in *2017 IEEE International Conference on Big Data (Big Data)*. Boston, MA, USA, Dec. 11-14: IEEE, 2017, pp. 596–605.

[34] Y.-F. Li, Y. Gao, G. Ayoade, H. Tao, L. Khan, and B. Thuraisingham, "Multistream classification for cyber threat data with heterogeneous feature space," in *The World Wide Web Conference*, San Francisco, CA, USA, May. 13-17, 2019, pp. 2992–2998.

[35] Y. Song, G. Zhang, H. Lu, and J. Lu, "A fuzzy drift correlation matrix for multiple data stream regression," in *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. Glasgow, UK, Jul. 19-24: IEEE, 2020, pp. 1–6.

[36] J. D. Hamilton, *Time series analysis*. Princeton university press Princeton, NJ, 1994, vol. 2.

[37] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine learning*, vol. 23, no. 1, pp. 69–101, 1996.

[38] A. Liu, Y. Song, G. Zhang, and J. Lu, "Regional concept drift detection and density synchronized drift adaptation," in *the 26th International Joint Conference on Artificial Intelligence*, Melbourne, Australia, Aug. 19-25, 2017, pp. 2280–2286.

[39] Y. Sakamoto, M. Ishiguro, and G. Kitagawa, *Akaike information criterion statistics*. KTK Scientific, Tokyo, with D. Reidel, Dordrecht, Holland, 1986.

[40] C. Li, F. Wei, W. Dong, X. Wang, Q. Liu, and X. Zhang, "Dynamic structure embedded online multiple-output regression for streaming data," *IEEE Transactions on Pattern Analysis and Machine Intelligence.* , 2018.

[41] P. scikit-learn package, "sklearn.datasets.make_friedman1," 2007. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_friedman1.html#sklearn.datasets.make_friedman1

[42] J. H. Friedman, "Multivariate adaptive regression splines," *The annals of statistics*, pp. 1–67, 1991.

[43] E. Ikonomovska, J. Gama, and S. Džeroski, "Learning model trees from evolving data streams," *Data mining and knowledge discovery*, vol. 23, no. 1, pp. 128–168, 2011.

[44] E. Ikonomovska, J. Gama, B. Ženko, and S. Džeroski, "Speeding-up hoeffding-based regression trees with options," in *the 28th International Conference on Machine Learning*. Bellevue Washington, USA, Jun. 28- Jul. 2: Citeseer, 2011, pp. 537–544.

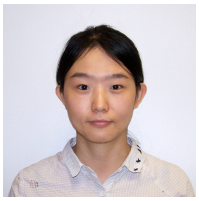
[45] A. Bifet, G. Holmes, B. Pfahringer, and E. Frank, "Fast perceptron decision tree learning from evolving data streams," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Hyderabad, India, Jun. 21-24, 2010, pp. 299–310.

[46] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive online analysis," *Journal of Machine Learning Research*, vol. 11, no. May, pp. 1601–1604, 2010.

[47] X. Zhu, "Stream data mining repository, http://www.cse.fau.edu/ xqzhu/stream.html," 2010.

[48] I. Žliobaitė, J. Hollmén, and H. Junninen, "Regression models tolerant to massively missing data: a case study in solar-radiation nowcasting," *Atmospheric Measurement Techniques*, vol. 7, no. 12, pp. 4387–4399, 2014.

[49] T. Pohlert, "The pairwise multiple comparison of mean ranks package (PMCMR)," *R package*, pp. 2004–2006, 2014.



Yiliao Song (S'17) received a M.S. in probability and statistics in mathematics from the School of Mathematics and Statistics, Lanzhou University, China, in 2015. She is working toward a Ph.D. with the Faculty of Engineering and Information Technology, University of Technology Sydney, Australia. Her research interests include regression, prediction, concept drift and data stream mining. She has published 17 papers related to concept drift, and data stream prediction.

Jie Lu (F'18) is a Distinguished Professor and the Director of Australian Artificial Intelligence Institute (AAII) at the University of Technology Sydney, Australia. She is also an IFSA Fellow and Australian Laureate Fellow. She received a PhD degree from Curtin University, Australia, in 2000. Her main research expertise is in transfer learning, concept drift, decision support systems and recommender systems. She has been awarded 10 Australian Research Council (ARC) discovery grants and led 15 industry projects. She has published over 450 papers in IEEE transactions and other journals and conferences, supervised 40 PhD students to completion. She serves as Editor-In-Chief for Knowledge-Based Systems (Elsevier) and Editor-In-Chief for International Journal on Computational Intelligence Systems (Atlantis). She has delivered 27 keynote speeches at IEEE and other international conferences and chaired 15 international conferences. She has received the UTS Medal for Research and Teaching Integration (2010), the UTS Medal for research excellence (2019), the IEEE Transactions on Fuzzy Systems Outstanding Paper Award (2019), the Computer Journal Wilkes Award (2018), and the Australian Most Innovative Engineer Award (2019).

Haiyan (Helen) Lu (SM'15) is an Associate Professor and a core member of the Decision Systems and e-Service Intelligent (DeSI) Research Laboratory with the Australian Artificial Intelligence Institute, University of Technology Sydney, Australia. She received a Ph.D. degree in Electrical Engineering from University of Technology Sydney, Australia, in 2002. Her main research interests are heuristic optimization techniques, forecasting and prediction of time series, ontology-based knowledge representation, recommendation systems and causal relationship, inference and reasoning in data streams. She has authored/co-authored three book chapters, 77 refereed journal papers and 86 refereed international conference papers.

Guangquan Zhang is an Australian Research Council (ARC) QEII Fellow, Associate Professor and the Director of the Decision Systems and e-Service Intelligent (DeSI) Research Laboratory at the Australian Artificial Intelligence Institute, University of Technology Sydney, Australia. He received his Ph.D in applied mathematics from Curtin University, Australia, in 2001. From 1993 to 1997, he was a full Professor in the Department of Mathematics, Hebei University, China. His main research interests lie in the area of fuzzy multi-objective, bilevel, and group decision making, fuzzy measure, and machine learning. He has published six authored monographs, five edited research books, and over 450 papers including 240 refereed journal articles. Dr. Zhang has won nine ARC Discovery Project grants and many other research grants, supervised 30 PhD students to completion. He has served as a Guest Editor for five special issues of IEEE Transactions and other international journals.

APPENDIX

A. Proof for Theorem 1

Proof. The data stream $DS_t^{(\boldsymbol{X},y)}$ consists of the attribute variable $\boldsymbol{X}$ and the label variable y. Consider basing the estimated y (denoted by $\hat{y}$) on any predictor in the hypothesis set is $h(\boldsymbol{X})$ other than the conditional expectation, the loss would be

$$\begin{aligned} E\left[y-h(\boldsymbol{X})\right]^2 &= E\left[y-E(y|\boldsymbol{X})+E(y|\boldsymbol{X})-h\left(\boldsymbol{X}\right)\right]^2 \\ &= E\left[y-E(y|\boldsymbol{X})\right]^2 + E\left[E(y|\boldsymbol{x})-h(\boldsymbol{X})\right]^2 \\ &\quad + 2E\left\{[y-E(y|\boldsymbol{X})][E(y|\boldsymbol{X})-h(\boldsymbol{X})]\right\}. \end{aligned}$$

Let $\eta \equiv [y-E(y|\boldsymbol{X})][E(y|\boldsymbol{X})-h(\boldsymbol{X})]$. As the terms $E(y|\boldsymbol{X})$ and $h(\boldsymbol{X})$ are known constants under the condition of $\boldsymbol{X}$, $E([E(y|\boldsymbol{X})-h(\boldsymbol{X})]|\boldsymbol{X}) = E(y|\boldsymbol{X})-h(\boldsymbol{X})$. The expectation of η conditional on $\boldsymbol{X}$ can be written into:

$$\begin{aligned} E(\eta|\boldsymbol{X}) &= \left[E(y|\boldsymbol{X})-h(\boldsymbol{X})\right] E([y-E(y|\boldsymbol{X}))]\,|\boldsymbol{X}) \\ &= [E(y|\boldsymbol{X})-h(\boldsymbol{X})]\times 0 = 0. \end{aligned}$$

According to the law of iterated expectations, $E[\eta] = E(E[\eta|\boldsymbol{X}]) = 0$. Substituting $E[\eta]$ back into $E\left[y-h(\boldsymbol{X})\right]^2$ gives

$$E\left[y-h(\boldsymbol{X})\right]^2 = E\left[y-E(y|\boldsymbol{X})\right]^2 + E([E(y|\boldsymbol{x})-h(\boldsymbol{X})]^2).$$

On the right side of the above equation, the first term does not depend on $h(\boldsymbol{X})$, and the second term cannot be made smaller than 0. Therefore the predictor that makes $E\left[y-h(\boldsymbol{X})\right]^2$ as small as possible satisfies $E([E(y|\boldsymbol{x})-h(\boldsymbol{X})]^2) = 0$, namely $h(\boldsymbol{X}) = E(y|\boldsymbol{X})$. □

B. Proof for Theorem 2

Proof. According to OLS, the estimation of $\boldsymbol{\theta'}$ by the n-size training set is as follows:

$$\begin{cases} \hat{\theta}_1' = \dfrac{(1-r)\overline{Xy_1}+r\overline{Xy_2}}{\overline{X^2}} \\ \hat{\theta}_0' = (1-r)\overline{y_1}+r\overline{y_2}-\hat{\theta}_1'\overline{X} \end{cases} \tag{23}$$

where $\overline{Xy_1} = \frac{1}{n_1}\sum_{t<t_d} Xy$, $\overline{Xy_2} = \frac{1}{n_2}\sum_{t\geq t_d} Xy$ and $r = n_2/n$ (the inference of estimating $\boldsymbol{\theta'}$ is given in Appendix C). Based on $\hat{\boldsymbol{\theta'}}$, the estimation of an unknown y_u is:

$$\hat{y_u} = \hat{\theta}_0' + \hat{\theta}_1' X_u \tag{24}$$

An unbiased estimation of $\boldsymbol{\theta}$ is obtained by n_2 observations from the new pattern, that is:

$$\widetilde{\theta}_1' = \frac{\overline{Xy_2}}{\overline{X^2}}, \widetilde{\theta}_0' = \overline{y_2} - \widetilde{\theta}_1\overline{X} \tag{25}$$

Based on $\widetilde{\boldsymbol{\theta}}$, the unbiased estimation of an unknown y_u is:

$$\widetilde{y_u} = \widetilde{\theta}_0' + \widetilde{\theta}_1' X_u, \text{ and } E(\widetilde{y_u} - y_u) = 0 \tag{26}$$

Therefore, the testing error of $\hat{y_u}$ is:

$$\begin{aligned} e_u &= \hat{y_u} - \widetilde{y_u} \\ &= (1-r)(\overline{y_1}-\overline{y_2}) + \frac{(1-r)(\overline{Xy_1}-\overline{Xy_2})}{\overline{X^2}}(X_u-\overline{X}) \\ &= (1-r)\left[(\overline{y_1}-\overline{y_2}) + \frac{\overline{Xy_1}-\overline{Xy_2}}{\overline{X^2}}(X_u-\overline{X})\right] \\ &= (1-r)\left[(\widetilde{\theta_0}-\widetilde{\theta_0'}) + (\widetilde{\theta_1}-\widetilde{\theta_1'})X_u\right] \end{aligned} \tag{27}$$

$$E_{X,y}(e_u) = (1-r)E\left[(\widetilde{\theta_0}-\widetilde{\theta_0'}) + (\widetilde{\theta_1}-\widetilde{\theta_1'})X_u\right] \tag{28}$$

As the coefficient of $1-r$ is considered to be constant, $\lim_{r\to 1} E(e_u) = 0$, and $E(e_u)$ **linearly decreases** to 0 as more instances from the new pattern are included in the training set (r goes to 1). □

C. Estimation of model parameters for the mixed training set

The estimation of θ_1 by OLS is:

$$\begin{aligned} \hat{\theta}_1 &= \frac{\sum_{t=1}^n X_t y_t}{\sum_{t=1}^n X_t^2} = \frac{\sum_{t=1}^{n_1} X_t y_t + \sum_{t=1}^{n_2} X_t y_t}{\sum_{t=1}^n X_t^2} \\ &= \frac{n_1\overline{Xy_1}+n_2\overline{Xy_2}}{n\overline{X^2}} = \frac{(1-r)\overline{Xy_1}+r\overline{Xy_2}}{\overline{X^2}} \\ \hat{\theta}_0 &= \frac{\sum_{t=1}^n y_t}{n} - \hat{\theta}_1\overline{X} = \frac{\sum_{t=1}^{n_1} y_t + \sum_{t=n1+1}^{n} y_t}{n} - \hat{\theta}_1\overline{X} \\ &= \frac{n_1\overline{y_1}+n_2\overline{y_2}}{n} - \hat{\theta}_1\bar{X} = (1-r)\overline{y_1}+r\overline{y_2}-\hat{\theta}_1\overline{X} \end{aligned}$$

$\overline{X}$ needs not to split because the distribution of X is unchanged.

D. Proof for Theorem 3

Proof. The relationship between Y_{t-1} and Y_t given $t \in (t_d, n]$ can be rewritten by the following difference equation:

$$y_t = c + \phi y_{t-1} + \epsilon_t, \tag{29}$$

where ϵ_t is a white noise sequence (Definition 5). As y_t is covariance-stationary, $|\phi| < 1$, and the stable solution to (29) is:

$$y_t = c\frac{1-\phi^\infty}{1-\phi} + \sum_{\tau=0}^{t}\phi^{t-\tau}\epsilon_{t-\tau} + \phi^\infty y_{-\infty}. \tag{30}$$

Similarly, $y_{t-1} = c\frac{1-\phi^\infty}{1-\phi} + \sum_{\tau=0}^{t}\phi^{t-1-\tau}\epsilon_{t-1-\tau} + \phi^\infty y_{-\infty}$, and $y_t - y_{t-1} = \phi^t\epsilon_t$. Given $e_u = y_t - y_{t-1} = \phi^{nr}\epsilon_{nr}$, it **exponentially decreases** to 0 as r increases. □

E. Proof for Theorem 4

Proof. Define $\delta_1^{(i)}$ as (31), and define $\delta_2^{(i)}$, $\Delta_1^{(i)}$ and $\Delta_2^{(i)}$ in the same way.

$$\delta_1^{(i)} = \begin{cases} 1 & \text{ith point of } \delta_1 \text{ is in } \varpi \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

d_ϖ^+ can be written as $d_\varpi^+ = \sum_i \delta_1^{(i)} - \sum_j \delta_2^{(j)} = \overline{\delta_1} - \overline{\delta_2}$. In [38], it has been proved that $E(\overline{\delta_1}) = \overline{\Delta_1}$ and $E(\overline{\delta_2}) = \overline{\Delta_2}$. Therefore, when no drift occurs, $E(d_\varpi^+) = 0$. As data points

from δ_1 and points from δ_2 are independent, $var(d_{\varpi}^{+}) = var(\overline{\delta_1}) + var(\overline{\delta_2})$. To compute $var(\delta_1)$ and $var(\delta_2)$, we introduce a random variable I_i,

$$I_i = \begin{cases} 1 & \Delta_1^{(i)} \in \delta_1 \\ 0 & \text{otherwise} \end{cases}. \tag{32}$$

$var(\delta_1)$ can be rewritten as:

$$\begin{aligned} var(\delta_1) &= var\left(\frac{1}{n_{\delta_1}}\sum_i I_i \Delta_1^{(i)}\right) \\ &= \frac{1}{n_{\delta_1}^2}\left(\sum_i (\Delta_1^{(i)})^2 var(I_i) + 2\sum_{i\neq j}\Delta_1^{(i)}\Delta_1^{(j)} cov(I_i, I_j)\right) \end{aligned} \tag{33}$$

Considering select n units from N units, the probability that each unit will be selected in n draws is $C_(N-1)^(n-1)/C_N^n = n/N$ and the probability that two units will be selected in n draws is $n(n-1)/N(N-1)$. Under this condition, I_i satisfies the following equations:

$$\begin{cases} E(I_i) = \dfrac{n}{N} = f, \\ var(I_i) = \dfrac{n}{N}\dfrac{N-n}{N} = f(1-f), \\ cov(I_i, I_j) = E(I_i I_j) - E(I_i)E(I_j) = -\dfrac{f(1-f)}{N-1}. \end{cases} \tag{34}$$

Based on this, $var(\delta_1)$ is computed as:

$$\begin{aligned} var(\delta_1) &= \frac{f(1-f)}{n_{\delta_1}^2}\left[\sum_i (\Delta_1^{(i)})^2 - \frac{2\sum_{i\neq j}\Delta_1^{(i)}\Delta_1^{(j)}}{n_{\Delta_1}-1}\right] \\ &= \frac{f(1-f)}{n_{\delta_1}^2}\left[\sum_i (\Delta_1^{(i)})^2 + \frac{\sum_i (\Delta_1^{(i)})^2}{n_{\Delta_1}-1} - \frac{\left(\sum_i \Delta_1^{(i)}\right)^2}{n_{\Delta_1}-1}\right] \\ &= \frac{1-f}{n_{\delta_1} n_{\Delta_1}}\left(\frac{n_{\Delta_1}}{n_{\Delta_1}-1}\sum_i (\Delta_1^{(i)})^2 - \frac{\left(\sum_i \Delta_1^{(i)}\right)^2}{n_{\Delta_1}-1}\right) \\ &= \frac{1-f}{n_{\delta_1}(n_{\Delta_1}-1)}\sum_i (\Delta_1^{(i)} - \overline{\Delta_1})^2 \\ &= \frac{S_{\Delta_1}^2}{n_{\delta_1}}(1-f) = \frac{S_{\delta_1}^2}{n_{\delta_1}}. \end{aligned} \tag{35}$$

So far, the expectation and variance of d_{ϖ}^{+} has been obtained. As ϖ is an arbitrary subset in $\mathbb{R}^{m+1}$, each d_{ϖ}^{+} can be seen as an element from a simple random sample. Therefore, $var(d_{\varpi}^{+})$ obeys a normal distribution based on the central limit theorem. □

F. The locally weighted regression

Details of the locally weighted regression algorithm can be found in Algorithm 3

G. Parameters in SVR and tree models

Parameters in SVR model: svm_type : 4 (nu-SVR), kernel_type : radial basis function: exp(-gamma*$|u-v|^2$), degree in kernel function: 3, gamma in kernel function: 1/num_features, coef0 in kernel function: 0, parameter C of nu-SVR: 1, parameter nu of nu-SVR: 0.5, cache memory size in MB: 100, tolerance of termination criterion: 0.001, whether to use the shrinking heuristics: 1, whether to train a SVC or SVR model for probability estimates: 0.

Parameters in regression tree model: MinParentSize: 10, QuadraticErrorTolerance: 1e-6, Weights: ones(size(X,1),1), Holdout: 0, KFold: 10, MaxNumSplits: num_features, MinLeafSize: 1.

Algorithm 3: The locally weighted regression

Input : $learnset$: data used to train the model, it contains $learninput$ and $learnoutput$;
$\boldsymbol{X}_q$: the value of features for the query point;
k: the number of neighbors;
$model$: it can be linear, SVR or Tree model;
$\boldsymbol{\theta}$: the other parameters needed in $model$.

Output: $\hat{y}_q$: the estimation of the label variable.

$Distance = knn(\boldsymbol{X}_q, learnsetinput)$
$I = sort(Distance, `ascend')$
$index = I(1:k)$ % find the index of $\boldsymbol{X}_q$'s k nearest neighbors in $learnset$ $(\boldsymbol{X}, y) = learnset(index)$
$\hat{y}_q = model(\boldsymbol{X}, y, \boldsymbol{\theta})$
return $\hat{y}_q$