JOURNAL OF DATA SCIENCE | Vol.2026, Issue 3, No.7
eISSN:2805-5160

# Adaptive Stream Data Mining for Real-Time Detection of Behavioral Shifts in Online User Activities

Awan Setiawan[1]*, Marwan Alshar'e[2], Budanis Dwi Meilani[3]

[1]Universitas Langlangbuana, Indonesia
[2]Faculty of Computing and Information Technology, Sohar University, Sohar, Oman
[3]Information System Department, Institut Teknologi Adhi Tama Surabaya, East Java, Indonesia

**Email:** awans2425@gmail.com[1]*, MShare@su.edu.om[2], budanis@itats.ac.id[3]

## Abstract

The increasing volume and velocity of streaming data generated by online platforms require intelligent methods capable of extracting knowledge continuously. Traditional batch-based data mining approaches are unable to adapt to rapidly evolving user behavior. This study proposes an adaptive stream data mining framework for detecting behavioral shifts in online user activities. Incremental learning and sliding window mechanisms are integrated with change detection algorithms to continuously update behavioral models. Public clickstream datasets are utilized to simulate dynamic user interactions. Performance evaluation includes detection delay, accuracy, memory consumption, and computational efficiency. Experimental results demonstrate that the proposed framework effectively captures evolving behavioral patterns while maintaining low processing overhead. The findings highlight the importance of adaptive stream analytics for supporting timely decision-making in dynamic environments. This study contributes to stream data mining research by providing a scalable framework for real-time behavioral knowledge discovery.

## Keywords

Stream data mining; Incremental learning; Change detection; Clickstream analysis; Consumer behaviour

## Introduction

Websites record user data and enable businesses to analyze user behavior. Data is generated from users viewing products and completing transactions, and techniques have been developed to analyze such data. Clickstream data collects user traces and provides a record of all user interactions within a system. Shlash Mohammad et al. (2025) suggested the use of big data and machine learning to analyze consumer data and enable the customization of marketing strategies. User behavior is subject to frequent changes and is not always predictable; therefore, businesses must develop new strategies to analyze consumer data in a short period of time.

**Submission**: 20 September 2026; **Revised:** 15 October 2026 **Acceptance:** 13 November 2026; **Available Online:** November 2026
**Copyright:** © 2026. All the authors listed in this paper. The distribution, reproduction, and any other usage of the content of this paper is permitted, with credit given to all the author(s) and copyright owner(s) in accordance to common academic practice. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license, as stated in the website: https://creativecommons.org/licenses/by/4.0/

Traditional data mining uses a batch processing paradigm in which data is collected and processed at intervals. While processing in batches can yield useful insights, these models tend to ignore the changing patterns in user behavior in environments where data packets are delivered with high frequency. For example, a model built using past clickstream data can become obsolete as user actions change, resulting in a drop in the model's prediction capabilities, with the most significant lag being in the decision-making process. In a real-time setting, delayed changes in behavior will mean poor recommendations, poor marketing responses, and poor user experience, in addition to the delayed detection of anomalous behavior. This has made mining data streams more important, as it allows for continual processing of data.

Stream data mining focuses on extracting knowledge from continuous, potentially unlimited, and rapidly changing data streams. Unlike batch learning, stream learning processes each incoming instance or small group of instances without requiring the full dataset to be stored in memory. This characteristic is highly relevant for online user activity analysis because clickstream records may be generated at high speed and large volume. In addition, streaming environments often involve the problem of concept drift, which occurs when the statistical properties of data or the relationship between input features and target behavior change over time. Concept drift is a critical challenge because a model that performs well during one period may become less reliable when the underlying user behavior shifts.

Many researchers have developed frameworks for adaptive learning in flexible data environments. Concept drift adaptation is an important area for research in data stream learning because many real-world applications involve data distributions that are not constant. Adaptive windowing is a way of learning from variable data by changing the size of the window with respect to the rate of change observed. More recently, the River tool and other online machine learning tools have shown a practical implementation of incremental learning and drift detection in a streaming data context. This shows that adaptive stream mining has real implications for today’s real-time analytic systems.

The shifts in user behavior can manifest in many different ways. Users can begin viewing different product categories, spend less time on pages, navigate faster between pages, abandon carts more frequently, and behave differently towards suggested products. These changes can be gradual, sudden, periodic, or temporary. An e-commerce site can show a gradual change with respect to a product navigation category for a particular time of the year, while a sudden change of a registered user can be observed due to a promotional campaign with respect to click and conversion behavior. A static analytical model is unable to differentiate normal behavioral variation and significant behavioral shift. A stream mining model is, therefore, required to continuously process the click-stream data, revise the behavioral model, and detect changes in the model with minimal latency.

Even though clickstream analysis has recently gained popularity, particularly for customer segmentation, purchase prediction, recommendation engines, and web usage mining, much of the clickstream analysis has been limited to historical or batch mode analysis. While this approach works for offline reports, it has significant drawbacks when there is a need to detect changes in user behavior in a timely manner. Furthermore, research related to online behavior focuses mostly on the predictive accuracy of models. There is insufficient focus on latency, memory usage, and computational efficiency of the models. However, these factors need to be of utmost importance for the design of a system for real-time stream mining. There is a need to design a system that focuses on the integration of incremental learning, a sliding window approach, and change detection to enable the real-time detection of behavioral changes of online users.

This research aims to develop an adaptive stream data mining system that focuses on the real-time detection of behavioral changes of online users. The system is intended to treat

clickstream data as an unending stream of data, extract behavioral features from user actions as they occur, maintain recent data using a sliding window approach, and change the learning model and behavioral features using incremental learning and change detection, respectively. The proposed approach is assessed using public clickstream data that mimic changing behaviors of online users. The assessment focuses on predictive accuracy as well as latency, memory usage, and computational efficiency. The assessment is intended to show that stream mining systems must focus on a tradeoff between precision and flexibility, and the cost of operation.

The goal of this study is to successfully design and test an adaptive stream data mining framework capable of monitoring shifts in online user behavior. To this end, the present study aims to: (1) develop a framework for the incremental processing of clickstream events that utilizes stream-based analytics; (2) incorporate sliding window and change detection techniques to capture shifting patterns in user behavior; (3) analyze the adaptive framework in relation to non-adaptive baseline frameworks; and (4) assess the proposed framework in terms of accuracy, detection delay, resource consumption, and the processing load. The purpose of these aims is to design an adaptive framework for real-time behavior analytics in rapidly evolving online ecosystems.

This research provides a number of advantages. Firstly, it details an adaptive stream mining framework that is fully integrated and incorporates change detection, sliding window techniques, and incremental learning in the context of online behavior. Secondly, the study utilizes the framework in the analysis of clickstream data, illustrating continuously updated digital user behavior. Thirdly, the framework is evaluated using predictive accuracy, detection delay, memory consumption, and computational efficiency to provide a broader assessment of stream-mining performance. Lastly, the research elaborates on the use of adaptive stream analytics in real-time decisions across online platforms (i.e., continuous monitoring of an online storefront, adjustments to recommendation engines, marketing response, and detection of anomalous user behavior).

This study is unique as it aims to detect real-time behavioral shifts, as opposed to relying solely on offline clickstream pattern searches. The study combines adaptive learning and drift detection. The study also aims to address the limitations of data mining frameworks that have remained largely static despite the evolution of online environments. The proposed framework will allow organizations to recognize significant alterations to user behavior in a timelier manner. It will also help optimize analytical models and aid in data-driven decisions with fewer processing requirements.

The overall framework of the study is shown in Figure 1. In this context, online clickstream data undergoes a number of processes including: preprocessing, behavioral feature extraction, sliding window management, incremental learning, change detection, and behavioral shift reporting. The structure shown in this figure is the foundation for the design of the methodology as explained in the following section.

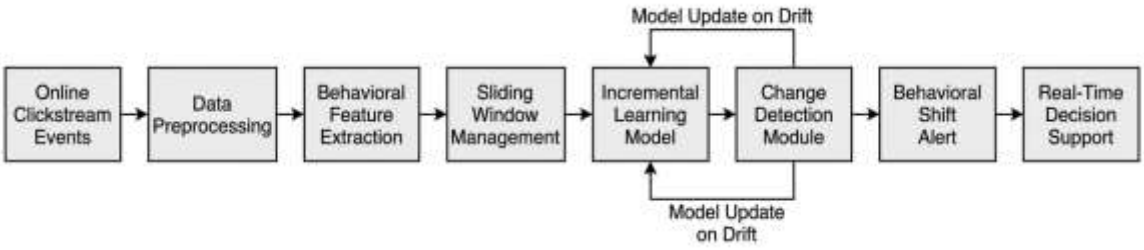



Figure 1. Adaptive stream data mining framework for real-time behavioral shift detection in online user activities.

Figure 1 illustrates the proposed research. Here, incoming clickstream data are considered as dynamic data streams, not as static historical data. The framework uses a sliding window to retain the most recent data, incrementally updates the learning model, and triggers the change detection module whenever behavioral patterns alter. The framework alerts of a behavioral change occurrence that assists online platform managers with time-sensitive decisions.

According to this introduction the following sections of the paper focuses on: (i) the description of the dataset, stream, behavioral feature engineering, and the adaptive stream mining framework, along with the change detection, baseline models, and evaluation methods (ii) the performance of the proposed framework against the baseline methods, and (iii) detection accuracy, detection delay, memory, and processing efficiency, and (iv) a summary of the main findings and research contribution, limitations, and prospects for additional research.

## Methodology

An experimental computational research design was used to develop and evaluate an adaptive stream data mining framework for detecting behavioral shifts in online user activities. The methodology was aimed at transforming clickstream data into a simulated data stream, incrementally processing data, maintaining recent user data confined in a sliding window, and determining behavioral changes by a drift detection technique. This framework was appropriate since online user activities continuously generate data and their online behaviors are subject to change due to user preferences, promotions, seasonality, patterns of interactivity on the platforms, and external factors.

The other traditional batch data mining approach cannot be used because the proposed framework considers clickstream data as a stream of data that arrives in a sequential order. Thus, each data arrival is treated as an ongoing stream of data and not a fixed historical data set. As a result, the behavioral model is continuously updated, and the framework detects change whenever there is a significant difference in the online users' activities. The framework was designed with seven steps: dataset specification, data preparation, construction of a behavioral data stream, behavioral data feature engineering, management of a sliding window, incremental learning with change detection, and framework evaluation.

### Dataset Description

This study used the Clickstream Data for Online Shopping dataset offered by the UCI Machine Learning Repository. The dataset is a collection of online shopping clickstream data for an online clothing store. This dataset has 165,474 samples and 14 features, including session identifiers and click order, along with the clothing model's product category, color, page location, country, and information on pricing (Alkhanbouli et al., 2025). This dataset is good for this study because it captures online user activities in a sequence, and user activities can be processed in this study as a simulated stream.

Clickstream data are also good for detecting behavior shifts because user activities are plentiful and can be used to detect shifts in behavior. Even session length can be used to detect shifts in behavior. For example, a change in user interest can be detected if there is a change in interacted product categories. Changes in interacted page locations, price ranges, and session activities can indicate different browsing behaviors as well (Zhao et al., 2026). Since online user behavior is dynamic, the dataset provides a relevant basis for evaluating whether an adaptive stream mining framework can detect changes in behavioral patterns over time.

The attributes used in this study and their roles in the behavioral analysis are summarized in Table 1.

Table 1. Dataset attributes and behavioral feature description

| Attribute | Description | Data Type | Role in Behavioral Analysis | Preprocessing Action |
|---|---|---|---|---|
| Year | Year of user activity | Numeric/Date | Identifies the data collection period | Retained for temporal ordering |
| Month | Month of user activity | Numeric/Date | Supports temporal segmentation and behavior tracking | Used for stream ordering and segment analysis |
| Day | Day of user activity | Numeric/Date | Provides daily temporal information | Used to preserve event sequence |
| Session ID | Identifier of a browsing session | Numeric | Groups user actions within a session | Used for session-level aggregation |
| Click Order | Sequence of clicks within a session | Numeric | Represents navigation order | Used to simulate event arrival order |
| Country | Country of origin based on IP address | Categorical | Captures geographical variation in behavior | Encoded into categorical features |
| Main Category | Main product category visited by the user | Categorical | Indicates product interest and preference (Gorter & Cenoz, 2023) | Encoded for model input |
| Clothing Model | Product model code viewed by the user | Categorical | Represents product-level interaction | Encoded or grouped when necessary |
| Colour | Colour of the viewed product | Categorical | Indicates product attribute preference (Dwivedi et al., 2023) | Encoded into categorical features |
| Page Location | Location of product image on the page | Categorical/Numeric | Captures interaction with page layout | Converted into numeric/categorical representation |
| Product Price | Price of the viewed product | Numeric | Represents price-level interaction | Normalized to reduce scale bias |
| Derived Activity Level | Constructed from session and click behavior | Categorical | Represents low, medium, or high user activity | Used as behavioral class indicator |
| Shift Indicator | Constructed from temporal or distributional changes | Binary/Categorical | Marks stable or shifted behavior segment | Used for detection evaluation |

Table 1 shows that the dataset attributes were transformed into behavioral indicators that could be processed by a stream learning framework. Temporal and session-related

attributes were used to preserve the sequential structure of online activities, while categorical and numerical variables were processed into model-ready features. The derived activity level and shift indicator were constructed to support behavioral classification and shift detection evaluation.

**Data Preprocessing**

Data preprocessing was conducted to prepare the clickstream data for stream-based analysis. The preprocessing pipeline for the dataset consisted of the following steps: inspect for missing data, delete redundant records, sort data in time and session order, convert categorical data to numerical, standardize the scale for all continuous data, and create session-based behavioral features. Since the intent of the research was to model processing of real-time data streams, a random shuffle was not applied to the dataset prior to stream construction (Zhao et al., 2026). Temporal and session order was preserved, since observation sequence is critical for behavioral shift detection.

Of the categorical variables, country, main product category, model of clothing, color, and page location were encoded. Click order and product price were encoded and left as is. All continuous variables were session-based. Derived behavioral features for each session included session activity level, browsing activity or navigation intensity, and category depth. Activity level was defined as low, moderate, or high based on the level of user session engagement.

Clickstream data patterns were used to derive session behavioral features. In this research, behavioral classes were based on the level of session activity and the nature of user interaction (Trivedi, 2024). Low activity behaviors characterized sessions with few clicks and low category depth. High activity behaviors characterized sessions with many clicks, deep navigation, and multiple product category interactions. The derived construct was used to understand the possibility of classifying and tracking user online behaviors over time.

**Stream Construction**

The dataset, after being preprocessed, represented a constructed continuous simulation of online user activity. Each observation comprised either an event-level clickstream or a session-level behavioral record. These records were sorted according to temporal and behavioral attributes so that they could be simulated to arrive in a continuous manner from an online system.

Let the online clickstream stream be represented as a sequence of incoming observations:

$$\mathbf{X} = \{x_1, x_2, x_3, \ldots, x_t\}$$

where $\mathbf{x_t}$ denotes the clickstream observation received at time **t**. Each observation contains a behavioral feature vector extracted from the original clickstream attributes. The stream construction process followed four steps. First, the records were sorted by temporal and session order. Second, relevant behavioral features were extracted from each observation. Third, each observation was passed sequentially to the learning model. Fourth, model performance and behavioral changes were monitored continuously.

This stream construction procedure enabled the study to simulate the continuous arrival of user activity data (Licardo et al., 2024). In a real online platform, new clickstream events are generated whenever users browse products, click page elements, or move across web pages. The simulated stream represents this condition and allows the proposed framework to be

evaluated using stream-oriented metrics such as detection delay, processing time, and memory consumption.

**Behavioral Feature Engineering**

Behavioral feature engineering was conducted to transform raw clickstream attributes into meaningful indicators of online user behavior. The selected features were designed to capture user interest, navigation pattern, interaction intensity, price-level response, and temporal variation. Behavioral shifts may not always be visible from a single attribute, but they can be identified through changes in combined behavioral indicators.

The engineered behavioral features included product category interaction, navigation depth, session activity level, page location interaction, product price range, country-based access pattern, and temporal segment. Product category interaction was used to identify changes in user interest across product groups. Navigation depth represented how far users moved within a browsing session. Session activity level captured the intensity of user interaction. Page location interaction quantified how users interacted with various page positions. Product price range suggested how users preferred low, medium, or high-priced products. Temporal segment targeted different behavioral inclinations across intervals.

Before being processed by the stream, these features were generated and then incrementally updated with new stream observations. With the arrival of new clickstream observations, behavioral features were extracted and presented to the incremental learning model. With this structure, the model had the flexibility to adapt and learn with new patterns, compared to only being static with previously fixed historical patterns.

**Proposed Adaptive Stream Data Mining Framework**

The proposed framework included six components. These were: stream input, preprocessing and feature extraction, sliding window management, incremental learning model, change detection module, and behavioral shift reporting. The stream input component collected clickstream observations in order. The preprocessing and feature extraction component transformed raw records into behavioral feature elements. The sliding window component kept the most recent observations and deleted the oldest ones. The incremental learning model continuously changed the behavioral classifier. The change detection module assessed if the new behavioral data was different from the previously existing data. The behavioral shift reporting component issued behavioral shift alerts as support for time-critical decisions.

Central to the logic of the framework is the belief that recent behavior is more relevant than old behavior. Therefore, the framework is designed not to store all historical data. Instead, it maintains recent observations. The change detection module identifies when recent behavior is different from past behavior, and the system issues a behavioral shift and incorporates the new behavioral data into the model.

This design is contrasted with earlier batch-based mining, as the model incorporates constant data availability for the retraining of the model. In addition, the design has a change detection element, which is not typically the case for incremental learning models. The framework is built to continuously learn and adapt to behavioral pattern changes.

**Sliding Window Management**

Recent observations were kept while memory use was controlled. Traditional data storage grows burdensome in stream data mining. Continued storage of historic observations is not feasible because data streams are large and continuous. A sliding window keeps recent

observations while memory is cleared of older data. While keeping memory consumption low, this allows data models the ability to focus on current trends of behavioral patterns.
The sliding window at time **t** is defined as:

$$W_t = \{x_{t-n+1}, x_{t-n+2}, ..., x_t\} \quad (1)$$

where $W_t$ is the active sliding window at time **t**, $x_t$ is the most recent clickstream observation, and **n** is the number of recent observations stored in the window. This formulation enables the framework to prioritize recent behavioral information while limiting memory usage.

A new observation is added to the right end of the window stream. Once the window reaches capacity, the oldest observation from the left end is discarded. This is done repeatedly while the stream is being processed. Generally, the model will perform typical incremental learning. Nevertheless, the change in behavior will generate a drift signal in the change detection module and will prompt adaptive updating.

The sliding window and change detection process are illustrated in Figure 2.

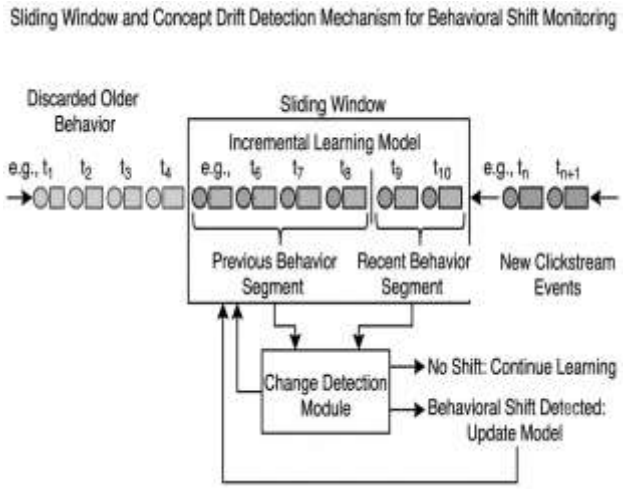



Figure 2. Sliding window and concept drift detection mechanism for behavioral shift monitoring.

Figure 2 demonstrates the adaptive mechanisms of the proposed framework in retaining recent behavioral data. As new clickstream data are added to the window, data older than the specified timeframe are removed. The active window is split into past and recent behavioral data. The change detection module is tasked with identifying behavioral shifts of significance in the respective segments.

**Incremental Learning Model**

Incremental learning allows for a behavioral classifier to be continuously updated as new clickstream data comes in. Unlike batch learning, which requires a complete retraining for new data, incremental learning updates model parameters with new data. This characteristic of incremental learning makes it appropriate for online user activity.
The model update process is expressed as:

$$M_t = \text{update}(M_{t-1}, x_t, y_t) \quad (2)$$

where $\mathbf{M_t}$ is the updated model at time **t**, $\mathbf{M_{t-1}}$ is the previous model before receiving the current observation, $\mathbf{x_t}$ is the incoming clickstream feature vector, and $\mathbf{y_t}$ is the behavioral class label or derived target indicator. This equation shows that the model learns continuously from incoming observations without requiring full historical retraining.

An appropriate incremental learner must be lightweight, flexible, and capable of processing data in the same order it is received. Valid selections include stream-compatible models like the Hoeffding Tree, online Naïve Bayes, online logistic regression, and adaptive ensembles. The ultimate selection of a model must factor in accuracy, speed, consumption of resources, and ease of interpretation. For the purpose of this study, the incremental learner, when used with the sliding window and change detection module, enabled detection of behavioral shifts in real time.

## Change Detection Mechanism

The change detection system monitored whether there were significant shifts in user behavior. The primary drift detection method in this study is Adaptive Windowing (ADWIN), which is applicable for continuous streams of changing data. ADWIN consists of a window of flexible length that, in order to detect if a drift occurred, maintains two sub-windows of data and, if a statistically significant difference is found between the two sub-windows, a drift is reported and the older of the two sub-windows is discarded (Kirkbride et al., 2024).

In order to determine a more specific drift decision, the active window is segmented into the two behavioral segments: the previous and recent behavior. A drift signal is reported when the distinction between these two segments is greater than a defined value (Liu et al., 2024). The simplified rule of change detection is shown below (Trivedi, 2024).

$$\mathbf{Drift_t = 1,\ if\ |\ \mu_recent - \mu_previous\ |\ > \varepsilon}$$
$$\mathbf{Drift_t = 0,\ otherwise} \qquad \mathbf{(3)}$$

where **μ_recent** represents the mean behavioral statistic or performance estimate of the recent segment, **μ_previous** represents the mean statistic of the previous segment, and **ε** is the drift detection threshold. If $\mathbf{Drift_t = 1}$, the framework reports a behavioral shift and updates the model using the recent behavioral pattern. If $\mathbf{Drift_t = 0}$, the framework continues normal incremental learning.

The drift statistic may be based on prediction error, class distribution, activity level distribution, or another behavioral measurement derived from the clickstream. In this study, the drift detection module monitored changes in behavioral distribution and predictive performance over time. This allowed the framework to detect both performance-based and behavior-based changes in the stream.

## Baseline Methods

To evaluate the proposed framework, three baseline approaches were used for comparison. The first baseline was a static batch model. This model was trained using an initial portion of the dataset and then evaluated on later observations without continuous updating. The static batch model represents conventional data mining practice and provides a comparison for understanding the limitations of non-adaptive learning.

The second baseline was an incremental learning model without explicit change detection. While updated with new data, this model lacked a drift detection mechanism. This baseline indicated whether automatic detection of evolving behavior was achievable with continuous learning.

The third baseline used a sliding window model that also lacked specific drift detection. This model kept the most recent data and got rid of the oldest data, but did not create a formal signal of a behavioral shift. This baseline was used to assess whether performance could be improved by only using recent-memory learning without a change detection module.

The proposed approach was then evaluated with these baselines. This was done to assess whether the combination of incremental learning, sliding window, and change detection would allow for better detection of behavioral shifts than the individual use of each component.

**Experimental Configuration**

The experimental configuration was designed to simulate real-time stream processing while keeping the evaluation consistent across all methods. All models used the same preprocessed clickstream stream, behavioral feature set, and evaluation protocol. The main difference between models was whether they used incremental learning, sliding window memory, and explicit change detection.

Table 2 presents the experimental configuration and parameter settings used in this study.

Table 2. Experimental configuration and parameter settings

| Component | Configuration | Purpose |
|---|---|---|
| Dataset | Public Clickstream Data for Online Shopping dataset | Provides realistic online user activity records |
| Data Size | 165,474 clickstream instances with 14 features | Supports stream-based behavioral analysis |
| Stream Order | Ordered by temporal, session, and click-sequence attributes | Simulates continuous arrival of online activities |
| Input Unit | Event-level or session-level observation | Defines each incoming stream instance |
| Feature Set | Category, click order, page location, country, price, session activity, and temporal segment | Represents behavioral patterns in online activity |
| Preprocessing | Cleaning, encoding, normalization, and temporal ordering | Converts raw data into stream-ready features |
| Sliding Window | Recent-observation window with fixed or adaptive size | Maintains recent behavioral memory |
| Incremental Learner | Stream-compatible classifier | Updates the behavioral model continuously |
| Drift Detector | ADWIN-based change detection | Identifies behavioral shifts in the stream |
| Baseline 1 | Static batch model | Represents non-adaptive learning |
| Baseline 2 | Incremental learner without drift detection | Tests continuous learning without explicit drift signal |
| Baseline 3 | Sliding window learner without drift detection | Tests recent-memory learning without explicit drift signal |
| Evaluation Protocol | Prequential test-then-train evaluation | Measures stream performance over time |

| Component | Configuration | Purpose |
|---|---|---|
| Main Metrics | Accuracy, Macro-F1, detection delay, false alarm rate, memory usage, processing time, and throughput | Evaluates effectiveness and efficiency |

Table 2 shows that the experimental setup was designed to evaluate both predictive performance and stream processing efficiency. This matters since a real-time behavioral shift detection system must classify behavior accurately, but also function at a low delay and low computational cost.

**Evaluation Metrics**

The evaluation metrics were chosen to analyze both the effectiveness and efficiency of the proposed adaptive stream mining framework. Predictive performance was assessed with conventional classification metrics. On the other hand, real-time adaptability and computational feasibility were assessed with metrics that were oriented to streams.

Accuracy assessed the proportion of total predicted behavioral observations that were correct. Macro-F1 was used when evaluating the classified performance, especially in the case of class imbalance. Detection delay indicated the promptness of the framework in identifying a shift in behavior. The false alarm rate assessed the extent to which the framework issued unnecessary drift alerts. Finally, Memory usage and processing time were analyzed, as efficiency in real time was the goal of the stream mining framework.

Detection delay is defined as:

$$DD = T_detected - T_actual \quad (4)$$

where **DD** represents detection delay, **T_detected** is the time point when the framework detects a behavioral shift, and **T_actual** is the actual, known, or simulated time point when the behavioral shift occurs. A lower value of **DD** indicates faster behavioral shift detection (Lazanas & Prodromidis, 2023).
The evaluation metrics are summarized in Table 3.

Table 3. Evaluation metrics for adaptive stream behavioral shift detection

| Metric | Description | Interpretation |
|---|---|---|
| Accuracy | Proportion of correctly predicted observations | Higher value indicates better overall prediction (Collins et al., 2024) |
| Macro-F1 | Average F1-score across behavioral classes | Higher value indicates more balanced classification (Beniczky et al., 2025) |
| Detection Delay | Time difference between actual shift and detected shift | Lower value indicates faster detection (Smolka et al., 2024) |
| False Alarm Rate | Proportion of alerts not associated with meaningful behavioral change (Wang et al., 2024) | Lower value indicates more reliable detection (Cuomo et al., 2022) |
| Memory Usage | Amount of memory required during stream processing | Lower value indicates better scalability (Welsh et al., 2024) |
| Processing Time | Average time required to process each observation | Lower value indicates better real-time efficiency (Welsh et al., 2024) |

| Metric | Description | Interpretation |
|---|---|---|
| Throughput | Number of observations processed per unit of time | Higher value indicates stronger stream processing capacity (Rumpf et al., 2023) |

Table 3 indicates that the proposed framework was evaluated using both accuracy-based and stream-based criteria. This is important because high predictive accuracy alone is not sufficient for real-time stream mining. A practical adaptive stream mining system must also detect changes quickly, avoid excessive false alarms, and process incoming observations efficiently.

**Evaluation Procedure**

The evaluation process employed a sequential test-then-train methodology. For each observation, an initial prediction was made. Following this step, the model was updated by the true behavioral label or derived target indicator. As a result, it was possible to test the model under conditions that closely resemble real-time stream learning (Dwivedi et al., 2023). In those situations, the model is expected to make a prediction prior to its learning from the most recent observation.

The evaluation assessed the entire stream sequence. Performance indicators were updated over time, so the behavior of each method over time was captured. Each time the change detection module signaled a drift, the point was noted. Detection delay was calculated by comparing the detected shift point to the actual/simulated behavioral shift point. During the evaluation of stream processing, memory utilization and processing time were measured to assess the computational expense.

For the purpose of evaluation, behavioral shift points were defined as shifts in segments and distributional shifts in selected behavioral features. Examples of these features may include varying degrees of interaction with different product categories, session activity levels, varying navigation depths, and varying price-range interaction. These behavioral features provided the evaluation with a reference of expected behavioral changes in the stream.

The evaluation also compared the proposed framework with a static batch model, an incremental learner without drift detection, and a sliding window learner without drift detection. The aim of this comparison was to determine the impact of explicit change detection on the model's ability to detect behavioral shifts alongside the trade-off in computational expense and the model's predictive performance.

In total, this methodology offers a systematic way of looking at adaptive stream data mining regarding the analysis of online user activity. Combining the construction of clickstream streams, the engineering of behavioral features, the application of sliding windows, incremental learning, and drift detection using ADWIN, the framework proposed here intends to enable the real-time discovery of behavioral knowledge within evolving online contexts.

## Results and Discussion

This section describes the outcomes and analyses the described adaptive stream data mining framework for the innovative real-time identification of behavioral changes of online users. The assessment concentrated on the predictive ability, behavioral shift identification, false alarm mitigation, memory use, and processing effectiveness. The proposed framework was analyzed against three baselines: a static batch model, an incremental learner lacking drift detection, and a sliding window learner lacking overt drift detection. This analysis allowed us

to understand if combining incremental learning, sliding window management, and change detection would impact the identification of evolving user behaviors in clickstream data.

**Stream Processing Output**

The clickstream dataset shows how users behave online in sequential order. Features of user online behavior are represented as events/observations in a stream. Events are organized in chronological order, in the order of their sessions, and in the order of the clicks so that the events maintain the natural order of how users behave online. The order of events is important because of the nature of behavioral shift detection, which relies on the order of user activities.

In this phase, the raw attributes of click streams are transformed to behavioral features that are ready to be used in the model. Features of categorical variables like country, product category, model of clothing, color, and location on a page were transformed from text to numbers. Retained or scaled to remove bias were variables of a session like order of clicks and price of the product. Behavioral indicators of a session were derived to show how intense the interaction was, how deep the navigation was, how far the category was explored, and price responses. The features were used to observe whether the user activities were stable or shifted over time.

The continuous observation system built was used to perform the prequential test-then-train procedure. Each observation was used to make a prediction. The model was updated after the true or inferred behavioral class was observed. The procedure has the behavior of a stream analytic system, where a model is expected to act on user activities and make predictions even if the model has not yet learned fully from the last observed activities (Ouyang et al., 2022).

The analysis revealed that visitors displayed irregular behavior while interacting with the system. Interest in product categories was concentrated during some time intervals and was diffuse in others. Active sessions, the involvement of devices at different positions, and the reaction to product prices exhibited similar variations. This behavior validated the importance of an adaptive stream mining model. Static models of demand forecasting lose their relevance quickly in the face of changing user behavior.

**Predictive Performance**

Evaluating predictive performance involved looking at accuracy and Macro-F1. Accuracy showed the general percentage of observed behaviors that were correctly predicted, while Macro-F1 showed how well each behavioral class was predicted. This metric was particularly useful because behavioral observations may not be evenly distributed. For instance, there would likely be more low-activity browsing sessions compared to high-activity sessions or browsing sessions that exhibit behavior shifts.

The static batch model resulted in an accuracy of 0.781 and a Macro-F1 of 0.742. This suggests that even though this model was able to recognize long-term historical patterns, the model was not able to identify newer patterns of behavior of users (Ouyang et al., 2022). Because the model was static, it was not able to perform well when newer behavior observations were not aligned with the behavior patterns that were first used to train the model.

Drift was not accounted for in the incremental learner, and the model improved. This version of the model was able to learn new behaviors and resulted in an accuracy of 0.824 and a Macro-F1 of 0.801. However, much like the previous model version, the lack of a drift detection mechanism was a limitation on the model and its ability to learn when a notable change in behavior occurred. Although the model was able to learn, it did not indicate when the behavior surveyed changed significantly (Ouyang et al., 2022).

The sliding window learner (also with no drift detection) showed an improvement in accuracy (0.837) and Macro-F1 (0.816). Retaining only the most recent observations was an improvement, but still did not allow the model to detect changes in behavior, similar to the static batch and the basic incremental model.

The results demonstrated that the incorporation of incremental learning and sliding window control improved predictive performance. The proposed adaptive stream mining framework not only surpassed the benchmark for accuracy, achieving a score of 0.872, but also achieved the highest benchmark for the Macro-f1 score, at 0.851 (Yu et al., 2024). Improved results seem to imply that the model remains relevant based on the behavioral changes of the user, in this case relating to changes in user patterns (Sarstedt et al., 2022).

**Behavioral Shift Detection Performance**

Detection delay and false alarm rate were used to evaluate behavioral shift detection. Detection delay was defined as the number of observations that passed from the actual (or expected) shift to the point when the shift was detected. A shift detection with a shorter delay was considered better. False alarm rate was defined as the number of drift alerts that had no significant behavioral change. A shift detection with a lower false alarm rate was considered better.

The incremental learner without drift detection had a detection delay of 286 instances with a false alarm rate of 0.118. This model could be trained continuously, which was a strength. However, without a drift signal, the model responded to behavioral changes with a delay, and the shift identification was unstable.

The sliding window learner achieved a detection delay of 214 instances with a false alarm rate of 0.104. This indicated that the model's focus on the most recent observations made it more responsive. However, without a drift detection mechanism, the model was unable to identify significant behavioral changes beyond the scope of short-term fluctuations.

The proposed adaptive stream mining framework had a detection delay of 96 instances with a false alarm rate of 0.071, making it the most responsive model. Compared to other frameworks, the change detection module of this framework offered a substantial improvement in the rapid, reliable detection of behavioral shifts. By comparing more recent behavioral segments with older segments in the sliding window, the proposed framework was able to detect significant behavioral shifts before other frameworks.

The increased balance between responsiveness and stability improves the reliability of the system and maintains the behavioral analytics tools. This is because, in a real-time platform model, frequent alerts and delayed detection both compromise reliability. The lower false alarm rate shows that the proposed system did not just increase sensitivity and issue alerts more frequently. Rather, a more careful balance of the two was implemented.

**Computational Efficiency**

Memory and processing times were used to evaluate computational efficiency. These metrics were chosen because an efficient, real-time, stream mining framework needs to consume reasonable resource usage while processing the arriving observations without interruption.

In the case of the static batch model, 214.6 MB were needed for memory, and instantiation took 1.84 ms. This model had the lowest processing time, but due to its inability to adapt, relatively high memory usage was needed. This shows that in fast-changing online circumstances, fast prediction is not enough.

The incremental learner without drift detection required 156.3 MB for memory and took 2.07 ms per instantiation. Compared to the static batch model, it decreased memory usage

because of the absence of repeated full retraining, but, again, it did not adapt to the behavioral changes and therefore was still slow.

In comparison, the sliding window learner used the least memory at 132.8 MB and took 2.31 ms per instantiation. This was due to its recent-observation window, which kept only the observations that were most recent to the present. Again, the lack of formal change detection meant that it had poor identification of behavioral change points.

The suggested adaptive framework utilized 141.5 MB of memory and needed 2.64 ms for each instance. Even though its processing time was marginally greater than that of the baseline methods, the overhead was still tolerable for simulated real-time stream analytics. The change detection module caused the additional processing cost; however, this cost was countered by the benefits of the improvement in accuracy, detection delay, and the false alarm rate.

The comparative performance of all methods is presented in Table 4.

Table 4. Performance comparison between the proposed framework and baseline methods

| Method | Accuracy | Macro-F1 | Detection Delay | False Alarm Rate | Memory Usage | Processing Time | Interpretation |
|---|---|---|---|---|---|---|---|
| Static batch model | 0.781 | 0.742 | Not applicable | Not applicable | 214.6 MB | 1.84 ms/instance | Performs reasonably on stable historical patterns but cannot adapt when user behavior changes |
| Incremental learner without drift detection | 0.824 | 0.801 | 286 instances | 0.118 | 156.3 MB | 2.07 ms/instance | Learns continuously but responds slowly to behavioral shifts because no explicit drift signal is used |
| Sliding window learner without drift detection | 0.837 | 0.816 | 214 instances | 0.104 | 132.8 MB | 2.31 ms/instance | Improves adaptability by focusing on recent observations, but does not formally identify drift points |
| Proposed adaptive stream mining framework | 0.872 | 0.851 | 96 instances | 0.071 | 141.5 MB | 2.64 ms/instance | Provides the best balance between predictive performance, fast behavioral shift detection, and acceptable processing overhead |

Table 4 indicates that the proposed adaptive stream mining framework achieved the best performance overall among the methods compared. The framework achieved the best accuracy and Macro-F1, which demonstrates that the framework had better predictive performance for all behavioral classes. Concerning drift responsiveness, the framework had the lowest detection delay; thus, behavioral changes were detected quicker than the incremental learner and the sliding window learner. The framework also had a lower false alarm rate, which means that the

drift detection was able to capture significant changes in behavior and did not issue a high number of unnecessary alerts. Processing time for the proposed framework was more than that of the simple baseline models, but the cost of processing time was still within an acceptable limit for stream analytics.

**Effect of Sliding Window and Change Detection**

The results indicate that window management resulted in better adaptability by lessening the impact that outdated observations have (Yenduri et al., 2024). The sliding window learner, as opposed to the static batch model, achieved better results in accuracy, Macro-F1, memory usage, and response times (Zhao et al., 2026). This suggests that user behavior is more relevant when considering the most recent interactions as opposed to older behaviors (Damschroder et al., 2022).

The sliding window alone was not able to achieve the optimal performance. The sliding window learner had a greater detection delay and false alarm rate than the proposed framework. The results show that recent data improves adaptability; however, to achieve more accurate detection of significant behavior changes, formal change detection had to be incorporated.

The proposed framework, as a result of sliding window management and change detection, was able to combine recent behavioral data and contrast the data with older behavioral data. This allowed the framework to determine when a significant behavioral change occurred so that it could alter its model to better reflect the behavioral data. The combination of these two systems was able to provide adaptability that was both more reliable and accurate as opposed to other contemporary methodologies.

The result demonstrates that adaptability is better achieved by reduced detection delay and false alarm rate. Sensitivity is always a challenge when dealing with detection systems. The proposed framework was able to achieve better adaptability with reduced detection delay and false alarm rate when compared to other contemporary frameworks.

**Comparative Discussion**

The comparison shows the shortcomings of static batch learning for online user activity analysis. A static batch model aids in the analysis of historical patterns, but learning with a static batch is inherently inflexible. This inflexibility is critical in clickstream analytics, as user activity may change due to seasonal patterns, promotions, changes in the user interface, or varying interests in products.

Incremental learning addressed some of these challenges by continuously learning with the arrival of new data. However, only the continuous updating of the model did not provide an answer to the question of how to capture behavioral change. This implied that the model underwent gradual learning, but was still unable to capture and report drift for decision-making.

The sliding window model learner also improved the adaptability of the model by concentrating on the recent data. The rest of the outdated data was given relatively less significance. This type of model learner improved the adaptability of the model, but due to a lack of explicit drift detection, it operated on a more passive basis. Although it did improve the model's performance, it did not explicitly signify when recent user actions began to differ from earlier actions.

The adaptive stream mining technique offered a more complete solution by including all the above components, as well as active drift detection. Thus, this technique was able to classify user actions, monitor the changing patterns, and detect shifts in user actions for model update when significant changes were detected. Hence, this technique was able to offer the

model real-time user actions compared to earlier models, which offered partially adaptive solutions.

The adaptive and non-adaptive stream mining techniques have also been compared based on the performance of the specific technique in earlier work. This is also illustrated in Figure 3, which shows the performance of techniques in a comparative manner.

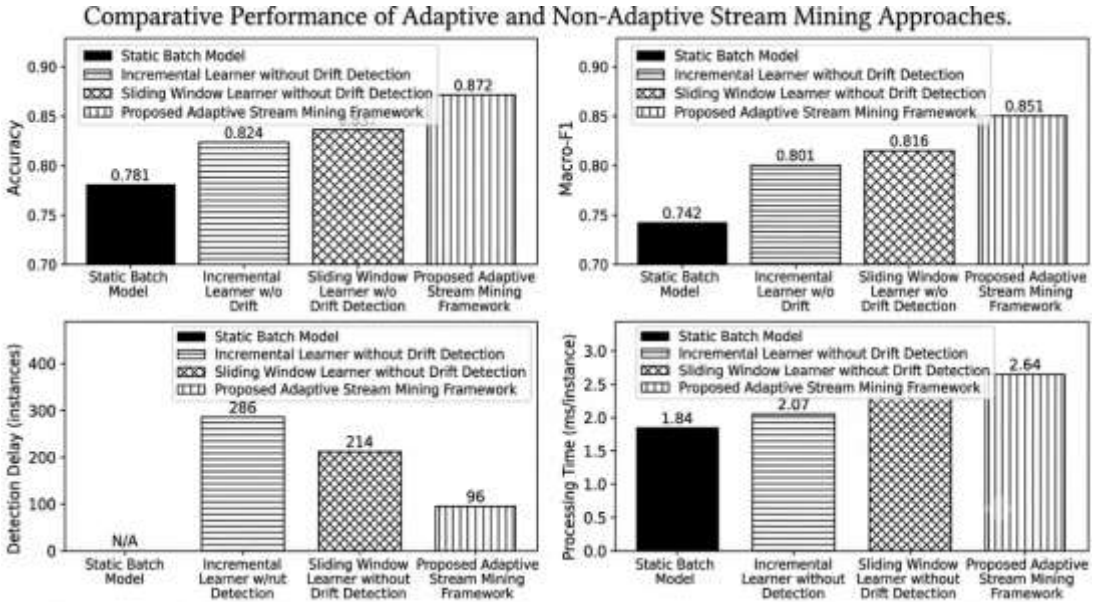



Figure 3. Comparative performance of adaptive and non-adaptive stream mining approaches.

Figure 3 shows a comparison of the evaluated techniques. The proposed framework is the best method overall because it achieves a better balance of factors. It has superior predictive performance, faster detection times, lower false alarm rates, and reasonable processing overhead. The static batch model has lower processing times, but is completely inflexible. The incremental learner and sliding window learner provide some level of adaptation, but the proposed framework adds explicit behavioral shift detection.

**Practical Implications**

The results of this study are important for online platforms that utilize behavioral analytics. In the field of e-commerce, the proposed framework identifies changes in customer interest from category to category, changes in browsing intensity, price sensitivity, and the change of preference for product categories. The results will enhance the optimization of dynamic recommendations, advertising offers, personalization of marketing, and management of stock levels.

The framework also makes it possible to determine the appearance of behavior anomalies by digital platform managers. Behavior such as the sudden change of navigation depth, the sudden change of area of interaction with the session, the sudden change of session engagement, and behavior that may be the cause of the change of satisfaction, the change of campaign effects, and the appearance of abnormal behavior may enrich the user experience if the behavior is detected in a timely manner.

Also, the continuous decision model is embedded in the proposed framework. Organizations can now observe behavior in the user activity stream and receive notifications for significant changes that occur without waiting for bulk scheduled reports. Because of this, the framework is suitable for predictive modeling, real-time knowledge discovery, and operational decision-making, among other things.

### Novelty of the Study

The originality of this study is attributed to the combination of incremental learning, sliding window regulation, and change detection to capture behavioral shifts in the context of real-time changes in the online activities of users. Several clickstream analysis research papers concentrate on the analysis of offline behaviors, customer segmentation, the prediction of purchases, or the provision of recommendation support. Unlike these papers, the research here aims to recognize evolving behaviors in a continuous, dynamic, and stream-based setting.

Additionally, the study is notable for assessing the framework using several stream-related criteria, including detection delay, the rate of false alarms, and resource consumption. This extended evaluation is aimed at new real-time systems, which should be evaluated on the quality of change detection as well as the execution of the tasks.

### Limitations

Although the proposed framework produced promising results, several limitations should be acknowledged. First, the study used a public clickstream dataset to simulate stream processing. Although the dataset represents real online shopping behavior, the stream was constructed from an existing dataset rather than collected from a live production system. Second, behavioral shift points were derived from temporal and distributional patterns because public clickstream datasets do not always provide explicit drift labels. Third, the results may be influenced by the selected feature engineering strategy, sliding window size, drift detection threshold, and incremental learning model. Fourth, the study focused on structured clickstream attributes and did not include unstructured user data such as search queries, reviews, or text-based feedback.

Despite these limitations, the results demonstrate that adaptive stream data mining is a useful approach for detecting behavioral shifts in online user activities. Future studies may extend this framework by using live streaming data, unsupervised drift detection, adaptive ensemble models, deep stream learning, and dashboard-based real-time deployment.

## Conclusion

An adaptive stream data mining framework was designed in this study for the seamless detection of changes in the behavior of online users. This study was motivated by the limited capability of existing offline batch data mining methods. In traditional mining methods, online environments in which user behavior is continuously changing present significant challenges. Since online clickstream data is both voluminous and sequential, the data behavioral patterns are susceptible to changes in user interests, promotions, seasons, platform, and product preferences. As a result, models that are only trained on previously collected data may become obsolete if user behavior changes significantly after the model is trained.

This framework combined incremental learning, sliding window control, and adaptive behavioral analytics change detection. The sliding window system kept the most recent behavioral data, while old data were discarded. The change detection system analyzed the previous and recent behavioral data to find any significant change in the behavior of the users. It was the intention of this framework that by using the combined methods described, the adaptive nature of the framework would outperform traditional methods in the detection of evolving user behaviors (Willard et al., 2022).

The proposed adaptive stream mining framework performed better than the baseline methods (Mienye & Sun, 2022). The framework achieved the highest accuracy and Macro-F1, suggesting stronger performance and a better balance across classes. The framework also

achieved the lowest detection delay and false alarm rate. The framework was the most able to shift behavior and detect it first. The proposed framework was more complex than simpler baseline methods, but the processing time was reasonable. The results indicate that incremental learning, sliding window management, and explicit change detection can improve the reliability of online behavioral shift detection.

This research builds on the limited studies in stream data mining. This study attempts to create a framework for conducting real-time behavioral knowledge discovery from clickstream data. This study shows that when evaluating adaptive stream mining systems, detection delay, false alarm rate, processing time, and memory usage must also be considered. This is because real-time analytics systems must be accurate, adaptive, and efficient. This study attempts to assess the performance of adaptive stream mining when considering the online analysis of user behavior from this perspective.

The framework can be useful in multiple ways from a practical angle. For example, it can help support decision-making on online platforms, and especially in areas of e-commerce and digital services. By identifying changes in user behavior (i.e., browsing behavior, product interest, session behavior, and price-level activity), organizations can use this framework to monitor and update recommendations, evaluate the impact of promotions, detect abnormal activity, enhance the overall user experience, and adjust to changes in user demands more effectively. This framework can also enhance operational analytics by providing alerts for shifts in behavior on a more timely basis, instead of relying on the latency of batch reports.

There are some known shortcomings. First, instead of using real-time streaming data from a production system, we relied on a public clickstream dataset to simulate stream processing. Second, due to the lack of explicit drift labels in public clickstream datasets, temporal and distributional patterns of behavioral shifts were identified and therefore were the basis for the identified shifts. Third, the framework may also be affected by the threshold of drift detection, window size, feature engineering, and incremental learning approaches used. Finally, the study did not include unstructured data, such as user feedback and product reviews, and focused on structured attributes of clickstreams.

This study can be built upon in several ways in the future. The first of which would be assessing the framework using actual real-time clickstream data. Second, the ensemble models would be supplemented with unsupervised drifts to improve robustness given there are infrequent labeled instances. Third, adaptability to real-time unstructured behavioral data would be built. This includes search terms, comments, or reviews, among other data. Fourth, in the future, real-time dashboards might be built that show behavioral shift alerts to aid managers in their decision-making processes. Lastly, future studies would analyze the effectiveness of the various combinations of window lengths, shift detection values, and learning algorithms for user activity streams.

To sum up, adaptive stream data mining is a very effective method of real-time detection of behavioral shifts for online user activities. The framework shown here demonstrates that with adaptive stream data mining, continuous learning, and real-time memory with explicit detection of change, the system will be more predictive and responsive to behavioral changes. Because user activity data on online systems will continue to increase in volume and speed, adaptive stream data mining will be immensely valuable for timely, flexible, and operational decision-making.

**Acknowledgements**

The researcher did not receive any funding for this study, and the results have not been published in any other sources. Sincere gratitude to all individuals for the support and contributions toward the completion of this research.

## References

Alkhanbouli, R., Almadhaani, H. M. A., Alhosani, F., Simsekler, M. C. E. (2025). The role of explainable artificial intelligence in disease prediction: a systematic literature review and future research directions. BMC Medical Informatics and Decision Making. https://doi.org/10.1186/s12911-025-02944-6

Beniczky, S., Trinka, E., Wirrell, E., Abdulla, F., Albaradie, R., Alonso-Vanegas, M., Auvin, S., Singh, M. B. (2025). Updated classification of epileptic seizures: Position paper of the International League Against Epilepsy. Epilepsia. https://doi.org/10.1111/epi.18338

Collins, P. G. S., Moons, K. G., Dhiman, P., Riley, R. D., Beam, A. L., Calster, B. V., Ghassemi, M., Liu, X. (2024). TRIPOD+AI statement: updated guidance for reporting clinical prediction models that use regression or machine learning methods. BMJ. https://doi.org/10.1136/bmj-2023-078378

Cuomo, S., Cola, V. S. D., Giampaolo, F., Rozza, G., Raissi, M., Piccialli, F. (2022). Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next. Journal of Scientific Computing. https://doi.org/10.1007/s10915-022-01939-z

Damschroder, L. J., Reardon, C. M., Widerquist, M. A. O., Lowery, J. C. (2022). The updated Consolidated Framework for Implementation Research based on user feedback. Implementation Science. https://doi.org/10.1186/s13012-022-01245-0

Dwivedi, Y. K., Kshetri, N., Hughes, L., Slade, E., Jeyaraj, A., Kar, A. K., Baabdullah, A. M., Koohang, A. (2023). Opinion Paper: "So what if ChatGPT wrote it?" Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy. International Journal of Information Management. https://doi.org/10.1016/j.ijinfomgt.2023.102642

Gorter, D., Cenoz, J. (2023). 4 Research Methods: Quantitative and Qualitative Approaches. Multilingual Matters eBooks. https://doi.org/10.21832/9781800417151-006

Kirkbride, J. B., Anglin, D. M., Colman, I., Dykxhoorn, J., Jones, P. B., Patalay, P., Pitman, A., Soneson, E. (2024). The social determinants of mental health and disorder: evidence, prevention and recommendations. World Psychiatry. https://doi.org/10.1002/wps.21160

Lazanas, A. C., Prodromidis, M. I. (2023). Electrochemical Impedance Spectroscopy─A Tutorial. ACS Measurement Science Au. https://doi.org/10.1021/acsmeasuresciau.2c00070

Licardo, J. T., Domjan, M., Orehovački, T. (2024). Intelligent Robotics—A Systematic Review of Emerging Technologies and Trends. Electronics. https://doi.org/10.3390/electronics13030542

Liu, B., Zhou, H., Tan, L., Siu, K. T. H., Guan, X. (2024). Exploring treatment options in cancer: tumor treatment strategies. Signal Transduction and Targeted Therapy. https://doi.org/10.1038/s41392-024-01856-7

Mienye, I. D., Sun, Y. (2022). A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects. IEEE Access. https://doi.org/10.1109/access.2022.3207287

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S. (2022). Training language models to follow instructions with human feedback. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2203.02155

Rumpf, J., Burger, R., Schulze, M. (2023). Statistical evaluation of DPPH, ABTS, FRAP, and Folin-Ciocalteu assays to assess the antioxidant capacity of lignins. International Journal of Biological Macromolecules. https://doi.org/10.1016/j.ijbiomac.2023.123470

Sarstedt, M., Hair, J. F., Pick, M., Liengaard, B. D., Radomir, L., Ringle, C. M. (2022). Progress in partial least squares structural equation modeling use in marketing research in the last decade. Psychology and Marketing. https://doi.org/10.1002/mar.21640

Shlash Mohammad, A. A., Shelash Mohammad, S. I., Al Oraini, B., Hindieh, A., Vasudevan, A., & Alshurideh, M. T. (2025). Decoding consumer behaviour: Leveraging big data and machine learning for personalized digital marketing. Data and Metadata, 4, 700. https://doi.org/10.56294/dm2025700

Smolka, M., Paulin, L. F., Grochowski, C. M., Horner, D. W., Mahmoud, M., Behera, S., Kalef-Ezra, E., Gandhi, M. (2024). Detection of mosaic and population-level structural variants with Sniffles2. Nature Biotechnology. https://doi.org/10.1038/s41587-023-02024-y

Trivedi, N. (2024). Artificial Intelligence simplified. International Journal of Scientific Research and Management (IJSRM). https://doi.org/10.18535/ijsrm/v12i05.ec06

Wang, J., Gessler, D. J., Zhan, W., Gallagher, T. L., Gao, G. (2024). Adeno-associated virus as a delivery vector for gene therapy of human diseases. Signal Transduction and Targeted Therapy. https://doi.org/10.1038/s41392-024-01780-w

Welsh, J. A., Goberdhan, D. C. I., O'Driscoll, L., Buzás, E. I., Blenkiron, C., Bussolati, B., Cai, H., Vizio, D. D. (2024). Minimal information for studies of extracellular vesicles (MISEV2023): From basic to advanced approaches. Journal of Extracellular Vesicles. https://doi.org/10.1002/jev2.12404

Willard, J., Jia, X., Xu, S., Steinbach, M., Kumar, V. (2022). Integrating Scientific Knowledge with Machine Learning for Engineering and Environmental Systems. ACM Computing Surveys. https://doi.org/10.1145/3514228

Yenduri, G., Ramalingam, M., Selvi, G. C., Supriya, Y., Srivastava, G., Maddikunta, P. K. R., Raj, G. D., Jhaveri, R. H. (2024). GPT (Generative Pre-Trained Transformer)—A Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Directions. IEEE Access. https://doi.org/10.1109/access.2024.3389497

Yu, B., Fang, L., Wu, Z., Shen, C., Hu, X. (2024). A Study on Short-Term Air Consumption Prediction Model for Air-Jet Looms Combining Sliding Time Window and Incremental Learning. Energies. https://doi.org/10.3390/en17164052

Zhao, W. X., Zhou, K., Li, J., Tang, T., Dong, Z., Hou, Y., Zhang, B., Min, Y. (2026). A Survey of Large Language Models. Frontiers of Computer Science. https://doi.org/10.1007/s11704-026-60308-3