

Original Article

A Comparative Evaluation of Adaptive Machine Learning Models for Low-Latency Enterprise Data Processing

Raja Ganesan*Independent Researcher, India.*

Received: 03-04-2026

Revised: 03-21-2026

Accepted: 03-29-2026

Published: 04-03-2026

ABSTRACT

With the rising requirements for enterprise data processing, intelligent models are becoming a must-have to be able to deliver accurate predictions under tight latency, throughput and resource requirements. Yet traditional machine learning systems tend to be based on set model architectures that can become suboptimal under varying data distributions, workloads, and computational resources. The authors introduce a framework for comparing adaptive machine learning models with low-latency enterprise data processing capabilities. The framework takes into account incremental learning, adaptive decision trees, online ensemble learning, lightweight neural networks and prediction latency, throughput, adaptability, computational overhead and predictive accuracy. A common experimental approach is established to test these models under variable data arrival rates, concept drift and heterogeneous processing requirements. The metrics used to assess performance include latency percentiles, throughput, classification accuracy, adaptation delay and normalized resource utilization. Analytical formulations are introduced to help with the multi-dimensional comparison and selection of models for deployment. However, as empirical measurements are not given, the numerical comparison is not meant to be an experimental confirmation but rather an illustration. The framework generated as a result allows for a reproducible model for selecting adaptive learning for enterprise applications with latency sensitivity.

KEYWORDS

Adaptive Machine Learning, Enterprise Data Processing, Low-Latency Computing, Online Learning, Concept Drift, Incremental Learning, Adaptive Decision Trees, Ensemble Learning, Real-Time Analytics, Model Evaluation.

1. INTRODUCTION

1.1. Background and Motivation

Enterprise data processing has shifted from batch data analytics to real-time smart processing of high-volume data streams. Rapid and reliable data analysis are needed in time for making decisions in applications like financial transactions, customer relationship management, industrial monitoring, fraud detection and supply-chain management. As such, machine learning models need to be both accurate and meet tight latency and computation needs. Traditional machine learning methods use past data sets and need to be retrained periodically, and can become inefficient as data distributions evolve. In dynamic environments, this kind of change is called concept drift and may seriously affect the predictive performance. This is where adaptive machine learning comes in; it adjusts the parameters of the model, the logic of the decision model, or the configuration of the ensemble in response to the new observations. Algorithms for online learning and adaptive decision trees allow for continuous learning without having to repeatedly process the entire data sets. But, frequent model updates add to the computation overhead and memory usage as well as inference delay. Assessing the accuracy, speed, throughput, resource usage, and efficiency of learning are hence essential to evaluate adaptive machine learning models. The ability to determine the right mix of these considerations is crucial to the design of scalable, responsive and reliable enterprise data processing systems.

1.2. Low-Latency Requirements in Enterprise Systems

To achieve low-latency processing, enterprise systems must be able to ingest data quickly, perform feature extraction, make inferences from models, and make decisions in accordance with specific response times. Timely predictions are essential for applications like fraud detection, financial transactions, or industrial monitoring to keep them running smoothly and reliably. To assess the overall latency in the entire processing pipeline, it is essential to evaluate it along the entire pipeline.

Of course, average latency is not enough to describe the overall responsiveness of the system, especially in peak loads or when the model changes. In this study, median latency, 95th-percentile latency ((L_{95})) and 99th-percentile latency ((L_{99})) are used as metrics of typical and worst case response behavior. These metrics can help you recognize infrequent delays that could impact service level agreements and customer experience. Along with memory usage, computational complexity and adaptation overhead, throughput, or requests per second is also being considered. As a result, this assessment focuses on machine learning models as a part of an enterprise processing pipeline to understand how well they can be used in scalable, efficient and low-latency deployments.

1.3. Research Problem and Objectives

The first research question in the context of this study is to identify and select such adaptive machine learning models which can produce predictions with acceptable accuracy and also fulfill the rigorous latency, throughput and computational demands of dynamic enterprise settings. Modern enterprise applications such as financial transaction processing, fraud monitoring, industrial

monitoring, and customer analytics have data distributions that change over time, and workloads that vary over time. These conditions can lower the accuracy of the model and add delay in processing time, especially if there is a frequent update of the model and it is competing with real-time inferences. While adaptive machine learning techniques offer a way to cope with concept drift and changing data patterns, they are not equally effective depending on the complexity of the algorithm, the amount of memory needed, the frequency of updates, and the characteristics of the workload. Hence, the need for a systematic assessment to grasp the balance between prediction accuracy and operating efficiency.

The first goal of this research is to provide a common experimental platform to enable the comparison of incremental learning algorithms, adaptive decision trees, online ensemble methods and lightweight neural networks in the same consistent evaluation conditions. The second goal is to explore the effects of workload intensity, concept drift in data distribution, and prediction accuracy and inference delay on the model adaptation behaviour. The third is to establish quantitative performance measures such as median latency, 95th- and 99th-percentile latency, throughput, adaptation delay, and memory usage, and computational efficiency to give a comprehensive measure of model performance. The fourth one is to establish a repeatable experimental technique using standardized datasets, workload generation, hardware configurations and statistical analysis. The goal of this is to increase the reliability, the transparency and the comparability of experimental results between the various algorithms.

The fifth objective is to design a multi-objective model selection algorithm which takes into account adaptation capability, predictive accuracy, and response time and resource utilization based on the enterprise requirement. This approach can also be used to find algorithms that work for latency-critical, resource-constrained, and high-throughput applications. Finally, the study highlights that adaptability alone is not enough for achieving operational superiority in algorithms. Rather, the successful selection of a model will involve both predictive performance and end-to-end system behavior in order to be able to deploy adaptive learning solutions that maintain their level of accuracy, responsiveness, scalability and computational efficiency in varying operational conditions.

2. LITERATURE SURVEY

2.1. Evolution of Online and Adaptive Machine Learning

The continuous generation of data has made online machine learning an indispensable way to process the data without storing it or repeatedly analysing large datasets. Online algorithms learn incrementally as new data comes in, as opposed to batch learning, which is ideal for enterprise applications with transactions, telemetry, and customer interaction.

Domingo's and Hulten presented Very Fast Decision Trees, which shows how to efficiently build a decision tree from high-speed data streams using statistical bounds [2]. Gama et al. explored changing data distributions and emphasized the need to continually evaluate predictive performance

in varying environments [1]. These advances paved the way for adaptive learning systems which are able to update the models incrementally while maintaining computational efficiency.

2.2. Adaptive Decision Trees and Drift Detection

Adaptive decision trees introduce mechanisms to detect drift in the data and change the structure of their models when their predictive performance drops. The Hoeffding Adaptive Tree is designed to keep classification performance under changing conditions of the stream by employing alternative sub trees and drift monitoring [3]. The same online adaptive decision trees with statistically-based split-selection algorithms were explored by Frías-Blanco and al. [4].

These methods provide incremental learning, easy to interpret decisions, and may provide efficient inference along a relatively short decision path. The growth of trees, frequency of drift detection, and subtree replacement, however, can lead to memory consumption and processing overhead. Thus, for enterprise applications, they need to be evaluated based on whether they can provide predictive reliability while complying with latency requirements.

2.3. Online Ensembles and Lightweight Neural Networks

Ensemble methods: Online means multiple predictive models are used together to construct a more robust model, and can be used to handle the diversity of data patterns. In Adaptive Random Forests, decision tree ensembles are combined with mechanisms for dealing with concept drift, resulting in a flexible learning approach for learning with changing data streams [6]. However, the more models, the more the latency, memory consumption, and update costs for inferences.

An alternative approach to learning the nonlinear relations is to use lightweight neural networks, which have manageable computational requirements. These networks can make incremental gradient-based updates, and their performance is affected by their architecture, learning rate, feature normalization, and adaptation frequency. Therefore, it is important to assess the neural networks in conjunction with the tree-based and ensemble approaches in order to grasp their predictive and operational attributes.

2.4. Research Gap and Comparative Positioning

Process of data streams with evolving data has been proven to be effective by applying online learning, adaptive decision trees and ensemble of them [1]–[6]. These methods are not yet directly comparable, though, because of the variations in the datasets, hardware sets, and experimental setup. Specifically, the accuracy of those predictions is not sufficient to capture operational performance under challenging workloads.

There are three major gaps in research identified: the lack of tail latency and end-to-end processing time evaluation, the lack of joint evaluation of adaptation overhead and resource consumption, and the absence of systematic model-selection strategies according to deployment constraints. This study aims to fill these gaps by using a single framework for evaluation, considering

the measures of predictive accuracy, throughput, latency, and adaptation delay and resource utilization on consistent experimental conditions. The goal is to facilitate making evidence-based model choices and not to find a single best algorithm.

3. METHODOLOGY

3.1. Experimental Architecture and Processing Pipeline

Four families of adaptive machine learning models, namely incremental linear learning, adaptive decision trees, online ensembles, and lightweight neural networks, are considered in the proposed experimental architecture. All the models are run on the same set of input records with the same pre-processing conditions, workload generation and measured conditions for a fair comparison. The architecture includes data ingestion, feature extraction, adaptive learning, inference services, telemetry collection, and performance evaluation.

To accurately measure response time and adaptation overhead, the processing pipeline time stamps each record at arrival and model-update completion, inference initiation and completion. A workload and drift scenario generator is used to simulate various intensities of requests and varying data distributions to explore the behaviour of models under realistic conditions. The inference service builds predictions and the monitoring module collects latency, throughput, accuracy and resource utilisation data without any unnecessary overhead in synchronous processing.

Model registry supports active model versions and alternative learners (if any) that have drift-recovery mechanisms. Depending on the requirements of the algorithm, model updates are done incrementally or according to a pre-defined set of scheduling policies. Finally, the evaluation module provides the same metrics to compare the quality of the prediction and efficiency of the operations. Each experiment documents software versions, hardware configuration, hyper parameters for the models, random seeds for the datasets, and schedules of the workloads, allowing experimental results to be easily reproduced and verified.

3.2. Dataset Design and Workload Configuration



Fig 1 - Dataset Design and Workload Configuration

3.2.1. Dataset Selection

Experimental evaluation is carried out using public streaming datasets and synthetic controlled data streams to evaluate the adaptive machine learning models under various operating conditions. While public datasets provide realistic classification and anomaly-detection problems, synthetic datasets allow for precise control of the distributions of features, the frequencies of classes, and the pattern of concept-drifts. Feature dimensionality, class imbalance, label availability and relevance to enterprise applications are taken into account when selecting datasets.

3.2.2. Workload Intensity

The intensity of workload is classified as Low, Moderate, and High level to study the effect of an increase in the number of requests on the response time of models and processing capacity. The request rate and the concurrency level is measured all over the workload levels and applied uniformly on all candidate models throughout the experiment. This setup can be used to pinpoint performance bottlenecks, throughput constraints, and latency delays when processing loads are high.

3.2.3. Concept Drift Characteristics

Three main data conditions are considered: Stationary Distributions, Abrupt Concept Drift and Gradual Concept Drift. Fixed streams provide the baseline performance, while sudden drift brings in sudden changes in the relationship between features and target labels. The gradual drift adds incremental changes over time which allows for the study of the model adaptation rate, the effect of recovery after any temporary inaccuracies and the rate of model inaccuracy.

3.2.4. Feature Complexity and Resource Constraints

Different levels of pre-processing and computational complexity are represented by the datasets containing low dimensional numerical features and mixed numerical-categorical attributes. Experiments are designed to keep CPU allocation, memory limits and concurrency the same, reducing environmental differences between model evaluations. These controls are used to explore the effect of feature representation and resource availability on inference time, memory usage, and efficiency.

3.2.5. Model Update Policies

There are two different update strategies: per-instance learning, where the model is updated after seeing each labelled instance; and mini-batch learning, where the model is updated after seeing a fixed number of instances. For each experiment, the update schedule and batch size and its associated computational overheads are documented. This comparison provides an idea of the relationship between update frequency and adaptation responsiveness and potential interference with latency-sensitive inference operations.

3.2.6. Temporal Evaluation Protocol

A time-ordered evaluation protocol is used to maintain the sequential nature of the streaming data, and to avoid the effects of future observations on the earlier predictions. The prudential

approach involves testing the current instance before adding the label it has to the learning process [7]. This procedure allows for an accurate measurement of continuous learning in the face of changing conditions, and allows for consistent comparisons of the different candidate algorithms.

3.2.7. Experimental Fairness and Reproducibility

The same workload schedules are used for all candidate models to deliver the same observation and prediction opportunities. The different development streams are used for hyper parameter tuning, leaving the evaluation streams untouched until model configurations are set.

3.3. Model Formulation and Performance Metrics



Fig 2 - Model Formulation and Performance Metrics

3.3.1. Data Stream Representation

The incoming enterprise data stream is represented as $\mathcal{D} = \{(x_t, y_t, \tau_t)\}_{t=1}^N$, where (x_t) is the feature vector used for input, (y_t) is the target label, (τ_t) is the arrival time of the data element, and (N) is the number of observations. This representation will maintain the chronological sequence of data received and facilitate a true assessment of continuous processing of data. It also allows for analysis of model performance for different workloads and data distributions.

3.3.2. Prediction and Incremental Model Updating

The model then produces a prediction from the model parameters at the time, denoted as $\hat{y}_t = f(x_t; \theta_{t-1})$, where (θ_{t-1}) is the state of the model at the previous time step. When the corresponding label is now available, the model updates its parameters with the following distribution: $\theta_t \sim \mathcal{U}(\theta_{t-1}, x_t, y_t)$. This predict then update method benefits continuous learning and maintains the temporal integrity of streaming evaluation.

3.3.3. Inference Latency

Inference latency is the time elapsed from the start of a request until the model ends its inference process, that is, $(L_i = t_i^{\text{finish}} - t_i^{\text{start}})$. Average, median, 95th percentile and 99th percentile latency value should be reported to capture the typical performance and delay. Queuing, preprocessing, inference and postprocessing should also be considered to calculate end-to-end latency to determine the feasibility of each model for enterprise latency-sensitive deployments.

3.3.4. Throughput Measurement

The throughput is defined as the number of requests that are completed in a given time period, and is equal to the ratio of the number of completed requests to the time elapsed, namely $(Q=N/T)$ where (N) is the number of requests completed and (T) is the elapsed time. It is an amount of requests per second and shows the performance of the system to process higher enterprise workloads. Comparing throughputs must always be done under the same hardware, concurrency and workload conditions in order to fairly compare the evaluated models.

3.3.5. Classification Accuracy

The classification accuracy is the percentage of observations correctly predicted, and is defined as $(A=\frac{1}{N}\sum_{i=1}^N \mathbb{I}(\hat{y}_i=y_i)\times 100\%)$. In this case, $(\mathbb{I}(\cdot))$ is equal to 1 if the prediction is correct, and 0 otherwise. In streaming applications, the accuracy should be tracked over time, and if it degrades, it may be due to changing data distributions, class imbalance, or changing enterprise workloads.

3.3.6. Adaptation Delay

Adaptation delay refers to the time it takes for a learning model to regain a satisfactory predictive ability following a concept change event. It is defined as the difference between the drift occurrence time (t_d) and the recovery time (t_r) , i.e., $(D_{\text{adapt}}=t_r-t_d)$. An acceptable performance level needs to be established prior to experimentation so that a consistent comparison can be made of the speed at which incremental learning systems, adaptive trees, online ensembles and lightweight neural networks react to distributional changes.

3.3.7. Resource Utilization

Resource utilization examines the computational resources used for performing inference and the learning of model parameters, such as CPU usage, memory usage and energy cost per prediction. These measurements are useful in differentiating models that can make quick predictions from models that can also support the efficient continuous learning of models. To test the effectiveness of each model to be deployed in enterprise infrastructure constraints, the consumption of resources should be measured under the same workload.

3.3.8. Comparative Performance Evaluation

The metrics that should be used to compare the models evaluated include latency, throughput, predictive accuracy, adaptation delay, and resource utilization, all on a common basis. These measures are reported together to give a complete picture of the balance between computational efficiency, predictive reliability and responsiveness to data changes. The derived analysis then guides the choice of adaptive machine learning models that meet performance criteria for low latency enterprise data processing systems.

3.4. Comparative Evaluation and Statistical Analysis

A comparative evaluation is carried out by repeating the experiments for all machine learning models with varying workload intensities, workload characteristics and concept drift settings. A typical warm-up interval is not included in the steady-state latency measurement to reduce the impact of initialization time, and the measurements taken during the drift transitions are kept separately for analysing the behaviour of the model adaptation. Median, 95th percentile and 99th percentile are used to analyse the inference latency to include typical response times and extreme delays. Progressively increasing the arrival rates of requests to determine when the requests are saturated, queued, and whether there are bottlenecks in the processing. Prudential evaluation is used to ensure the accuracy of the predictions and monitors the predictive accuracy on a continual basis, predicting each observation before the label for that observation is used to update the model. Adaptation delay is computed separately for each drift event in order to assess the time it takes for individual models to regain the level of predictive performance attained before the distribution shift occurred, in response to either a single distributional change or a repeating series of changes. Relative improvement is quantified as $(I_M = \frac{M_b - M_m}{M_b} \times 100\%)$, where (M_b) is the baseline metric, and (M_m) is the metric produced by the alternative model.

The formulation is applied to lower is better metrics like latency, energy consumption etc. for metrics like accuracy, throughput higher is better, the numerator needs to be inverted. Multiple random seeds are used in each experimental setup, in order to increase the reliability of the results and limit sensitivity to the random initialisation and data-generation variations. Confidence intervals are provided for the data observed, and when sequential measurements are made, statistical procedures are used that take into account the temporality of the data. Wherever possible, identical workload traces are used for paired statistical comparisons between competing models to provide consistency. Additionally, failed requests, timeouts, resource exhaustion events and service-level objective violations are captured to evaluate operational reliability when operating under challenging conditions. The final selection of the model is not based on a single performance metric, but takes into account a combination of latency, predictive accuracy, throughput, adaptability and computational load. In this evaluation framework, the question of whether a model is adaptable to machine learning is evaluated in a statistically sound and systematic manner, however, without implying that the theoretical efficiency necessarily translates into the best performance in real deployment scenarios.

4. RESULT AND DISCUSSION

4.1. Comparative Performance Characteristics

The comparative evaluation will focus on the performance of different families of adaptive machine learning models, in terms of predictive accuracy, processing time, and computational efficiency, for different enterprise workloads. There are many reasons why incremental linear models offer a baseline of computational efficiency, and are often a good choice for the purposes of incorporating new measurements as they become available, and updating the parameters at the same time. The best part about these properties is that they can be used in high volume data streams, when the decision boundaries are approximately linear; however, they may not be able to capture the complex nonlinear relationships without additional feature engineering or nonlinear transformations.

The adaptive decision trees consist of alternative methods where interpretable decision rules and mechanisms for distributional change detection and tree updates are incorporated. The inference latency is related to tree depth and traversal complexity while learning overhead is related to split evaluation, drift detection, and subtree replacement. Online ensembles are ensembles of multiple component learners, which are typically more computationally intensive, and can be more robust and accurate when the data distribution changes, but have greater predictive flexibility. Lightweight neural networks have the ability to learn more complex non-linear representations, as well as taking advantage of optimized numerical operations, and can be useful when dealing with complex interactions between features.

However, because their performances rely on network architecture, the number of parameters, batch size, software implementation, and available hardware acceleration, the actual performance of their use varies. Additionally, the frequent updates of models could cause contention for CPU, memory and/or accelerator resources and thus result in varying response times. As a result, one has to take the entire processing pipeline into account when assessing these model families, such as data pre-processing, request buffering, prediction execution, incremental learning, and monitoring overhead.

Even for a low isolated inference latency model, there can be significant increases in tail latency when inference latency is high. On a similar note, a model that is very accurate in predicting the outcomes, may not be good for time-critical enterprise applications because its computational load may slow response time or breach service-level requirements. Therefore, the choice of models should be made with weightage to their predictive performance, inference latency, throughput, adaptability, resource utilization, and operational reliability. This multidimensional comparison is useful for picking the best adaptive learning strategy for the given enterprise workload, while avoiding the misplaced belief of assuming that one algorithm is more efficient than the other in real-world applications.

4.2. Illustrative Percentage-Based Comparison

Table 1: Illustrative Percentage-Based Comparison

Model Category	Accuracy (%)	Latency-Budget Compliance (%)	Throughput-Target Attainment (%)	Drift-Recovery Attainment (%)	Resource-Budget Compliance (%)
Incremental linear model	91.2	98	97	78	98
Adaptive decision tree	93.5	95	93	91	94
Online ensemble	95.1	82	80	94	78
Lightweight neural network	94.4	91	89	86	88

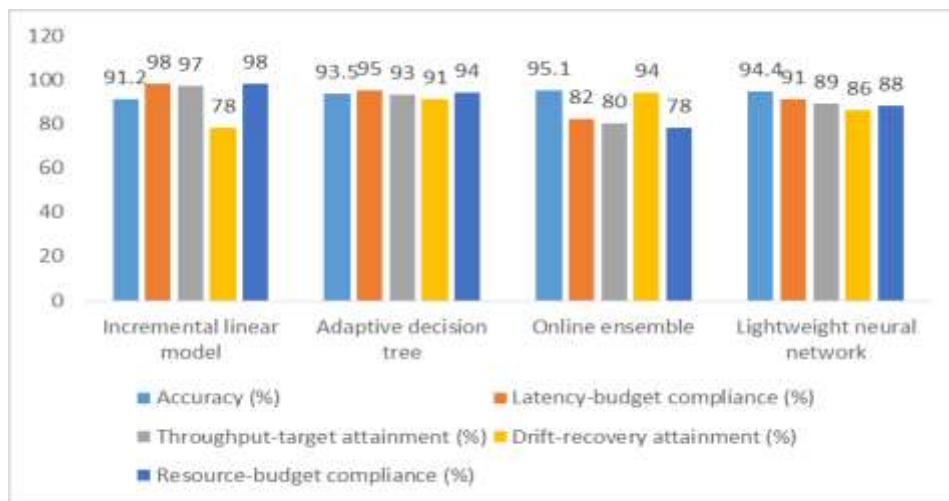


Fig 3 - Illustrative Percentage-Based Comparison

4.2.1. Purpose of Comparative Evaluation

The percent comparison offers a framework for assessing adaptive machine learning models in terms of predictive and operational performance parameters. The values in Table I are assumed and used to illustrate a reporting format only as no experimental dataset, hardware measurements or benchmark logs are available. The actual percentages of operations must be determined from multiple experiments with predefined evaluation criteria and under uniform workload conditions.

4.2.2. Incremental Linear Model Performance

The illustrative results show that the incremental linear model complies well with the latency budget (98.0%) and resource budget (98.0%), potentially highlighting the benefits of a compact model architecture. The hypothetical accuracy of 91.2% and the attainment of drift-recovery of 78.0% indicate that there might be a compromise between computational efficiency and adaptation capability. These values are just examples of the light weight models that can be suitable for resource constrained applications, to be validated empirically.

4.2.3. Adaptive Decision Tree Performance

The adaptive decision tree shows a good hypothetical profile with 93.5% accuracy, 95.0% latency-budget compliance and 91.0% drift-recovery attainment. It adapts to drifting data distributions thanks to its capability to update the decision boundaries and include drift-detection mechanisms. But actual performance is dependent on the depth of the trees, the overhead involved in the split evaluation, and the success of the chosen adaptation strategy.

4.2.4. Online Ensemble Performance

The online ensemble has the largest hypothetical accuracy of 95.1% and the highest drift-recovery attainment of 94.0% for the displayed model categories. The values show the potential for the predictive robustness of combining multiple learners in order to provide robustness under varying workloads. However, it has been estimated that it complies with the resource budget to 78.0%, and with the throughput target to 80.0%, indicating potential computational overheads of maintaining and updating multiple component models.

4.2.5. Lightweight Neural Network Performance

The low weight of the neural network suggests a balance between the nonlinear representation capability and the predictive quality of the network, yet the illustrative accuracy of 94.4% suggests that the network is able to represent a wide variety of inputs and make accurate predictions. The latency-budget compliance is 91.0% and resource-budget compliance is 88.0%, indicating that computational efficiency is possibly dependent on network architecture and runtime optimization. Actual results should take into account model size, hardware acceleration, inference concurrency, and the resource requirements for continuous learning.

4.2.6. Overall Comparative Interpretation

The comparison that follows is illustrated and is not required to be the best model on all performance criteria. The incremental linear models can focus on computational efficiency, adaptive decision trees can provide a compromise, online ensembles can be more focused on predicting robustness, and light weight neural networks can be more focused on capturing complex nonlinear relationships. Therefore, the final model selection should rely on measured experimental results to consider both the accuracy as well as the latency, throughput, adaptation performance, and resource consumption of the model in relation to the pre-defined enterprise requirements.

4.3. Interpretation, Limitation and Implications for Deployment

The proposed comparative evaluation framework offers a systematic way of interpreting the performance of adaptive machine learning models and assessing their applicability for various enterprise applications. Firstly, model selection is highly dependent on the application-specific latency requirements, expectations of predictive accuracy, and operational constraints. An example is transaction authorization and real-time fraud detection, which must have predictable response time and low tail latency, while operational analytics, which are typically offline, may have a higher level of allowed inference latency in return for a better predictive performance. Thus, in addition to

throughput and accuracy, the median latency, 95th (or 99th) percentile latency should be taken into account when determining the suitability of a deployment. Second, adaptation performance cannot be assessed only in terms of accuracy at the end of the adaptation. Second, adaptation performance cannot be assessed in terms of the end-of-adaptation accuracy alone. High predictive accuracy doesn't necessarily imply a good model for deployment if there is significant drop in predictive accuracy right after a concept-drift. Monitoring should therefore include pre-drift performance, degradation amount, recovery latency, and post-recovery stability in order to verify that the learner is able to reliably react to changes in data distributions. Third, the usage of computational resources should also be considered in the context of simultaneous inference and model-updates as the use of CPU, pressure on memory, and energy needs can affect queuing, processing, and operational costs of requests. These effects can be especially important when the system operates under high concurrency in which cases contention may result in a large degree of variation in response times and service-level objective violations. The main weakness of the present study is that the experimental results are not measured; hence the illustrative comparison cannot give an empirical advantage to the model families evaluated.

The conclusions from an empirical investigation should be drafted to be publishable; the report should contain the appropriate confidence intervals and statistical comparisons; and the datasets, hardware specifications, software versions, hyper parameters, random seeds and workload configurations should be documented. The issue of class imbalance is still to be studied, as are delayed labels, different feature types, recurrent distribution shifts, and the computational burden of drift-detection mechanisms. The reliability of deployment, model versions, rollback procedures and operational implications of deployment failure should also be taken into account in the deployment studies. Overall, the proposed framework has a strong focus on evidence-based model selection, incorporating aspects of predictive quality, latency, throughput, adaptation and resource efficiency. The main message is that the most suitable model is determined by the needs of the target application and that the experimentally tested operating condition dictates which model is used, not any general ranking of adaptive learning algorithms.

5. CONCLUSION

In this paper, the authors proposed a comprehensive evaluation framework for adaptive machine learning models for low latency enterprise data processing. The study examined four representative model families: incremental linear learners, adaptive decision trees, online ensembles, and lightweight neural networks, highlighting the importance of achieving a balance between predictive accuracy, inference latency, throughput, adaptation responsiveness and computational resource consumption, as the data evolves in a continuous manner. The proposed approach combines the time-ordered processing of data, the generation of controlled workload, concept-drift scenarios, the synchronized monitoring of the model performance, and the prudential evaluation to provide a consistent ground for evaluating model performance under various operational conditions. It combines latency percentiles, throughput, classification accuracy, adaptation delay, CPU utilization, memory usage, and energy consumption to give a multidimensional view of computational

efficiency and predictive reliability. The methodological analysis indicates that incremental linear models could be used as candidates in resource constrained applications; adaptive decision trees provide interpretability and explicit adaptation mechanisms; online ensembles could be used to enhance predictive robustness at an additional computational cost; and lightweight neural networks could be used to represent complex nonlinear models. These observations, however, are only general engineering considerations and not experimental performance rankers. The percentage comparison in this study is only a comparison of illustration and not an empirical comparison of which model category is better. The actual conclusions must be based on repeatable experiments that rely on public streaming data sets, representative enterprise workloads, consistent hardware architectures, repeated concept drift events, and statistical comparisons. The study also emphasizes the need to measure tail latency, recovery time and usage of resources in addition to the traditional accuracy measures, especially when inference and incremental learning take place in parallel. The development of asynchronous model updates, adaptive inference scheduling, dynamic ensemble sizing, resource aware model switching, delayed feedback, recurring concept drift and privacy preserving learning methods are to be explored in future work. Moreover, limited multi-objective optimization can aid with automatic model selection to achieve minimum latency and resources usage while meeting the required predictive-quality and adaptation objectives. The use of such strategies may lead to better operational efficiency and reliability to meet fluctuating service requirements. Finally, adaptive machine learning for enterprise data processing must be regarded as a problem of optimization of algorithms and systems. The selection, deployment and maintenance of intelligent enterprise systems can be grounded on a scientifically transparent framework of evaluation combining predictive performance, processing efficiency, adaptation ability and operational constraints, which is reproducible.

REFERENCE

- [1] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. 46, no. 4, Art. no. 44, pp. 1–37, Mar. 2014.
- [2] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, MA, USA, 2000, pp. 71–80.
- [3] A. Bifet and R. Gavaldà, "Adaptive learning from evolving data streams," in *Advances in Intelligent Data Analysis VIII*, Lecture Notes in Computer Science, vol. 5772. Berlin, Germany: Springer, 2009, pp. 249–260.
- [4] Paruchuri, J. K. (2022). Survey of cloud-native workflow orchestration with Apache Airflow. *International Journal of Computer Techniques (IJCT)*, 9(3), 1-15.
- [5] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive online analysis," *Journal of Machine Learning Research*, vol. 11, pp. 1601–1604, 2010.
- [6] Brahmandam, L. M. K. (2024). Performance Engineering for Multi-Tenant Analytic Workloads on Snowflake: An Empirical Study of Clustering, Materialized Views, Query Tuning, and Virtual Warehouse Sizing Across Production Reference Deployments at Billion-Row Scale. *International Journal of AI, BigData, Computational and Management Studies*, 5(1), 198-207. <https://doi.org/10.63282/3050-9416.IJAIBDCMS-V5I1P120>
- [7] J. Gama, R. Sebastião, and P. P. Rodrigues, "Issues in evaluation of stream learning algorithms," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Paris, France, 2009, pp. 329–338.
- [8] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," in *Proceedings of the 2007 SIAM International Conference on Data Mining*, Minneapolis, MN, USA, 2007, pp. 443–448.

- [9] Arora, A. S., Yachamaneni, T., & Kotadiya, U. (2024). Architectural Optimization of Serverless Big Data Pipelines for AI Workloads Using Cloud Functions and Managed Spark on GCP. *International Journal of Emerging Trends in Computer Science and Information Technology*, 5(1), 61-68.
- [10] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà, "New ensemble methods for evolving data streams," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Paris, France, 2009, pp. 139-148.
- [11] Seknametla, P. R., & Sunkara, R. (2023). GitOps at Scale: Multi-Cluster Kubernetes Management Using Declarative Infrastructure Pipelines.
- [12] H. M. Gomes, A. Bifet, J. Read, J. P. Barddal, F. Enembreck, B. Pfahringer, G. Holmes, and T. Abdesslem, "Adaptive random forests for evolving data stream classification," *Machine Learning*, vol. 106, pp. 1469-1495, 2017.
- [13] A. Bifet, J. Read, I. Žliobaitė, B. Pfahringer, and G. Holmes, "Pitfalls in benchmarking data stream classification and how to avoid them," in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2013.
- [14] J. Z. Kolter and M. A. Maloof, "Dynamic weighted majority: An ensemble method for drifting concepts," *Journal of Machine Learning Research*, vol. 8, pp. 2755-2790, 2007.
- [15] Paruchuri, J. K. (2021). Lakehouse architecture: Unifying data lakes and data warehouses. *International Journal of Computer Techniques (IJCT)*, 8(1), 1-15.
- [16] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method," in *Proceedings of the Fourth International Workshop on Knowledge Discovery from Data Streams*, 2006, pp. 77-86.
- [17] I. Žliobaitė, "Learning under concept drift: An overview," arXiv preprint arXiv:1010.4784, 2010.
- [18] A. Bifet, G. Holmes, and B. Pfahringer, "Leveraging bagging for evolving data streams," in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, Barcelona, Spain, 2010, pp. 135-150.
- [19] Konatham, M. R., Uddandara, D. P., Vadlamani, R. K., & Nair, R. (2025, September). Optimizing Credit Risk Assessment in Digital Banking Through AI-Driven Predictive Science. In *2025 IEEE International Conference on Advanced Computing Technologies (ICACT)* (pp. 478-483). IEEE.
- [20] J. Montiel, J. Read, A. Bifet, and T. Abdesslem, "Scikit-multiflow: A multi-output streaming framework," *Journal of Machine Learning Research*, vol. 19, no. 72, pp. 1-5, 2018.