Islamic Azad University , Shiraz Branch
نشریه تحلیل مدارها، داده ها و سامانه ها
Journal of Circuits, Data and Systems Analysis
sanad.iau.ir/journal/jcdsa

A novel and Intelligent Ensemble Framework for Real-Time Detection and Adaptation to Concept Drift in Data Streams Using Incremental Decision Trees

Hadi Tarazodar[1], Karamollah Bagherifard[*2], Samad Nejatian[3], Hamid Parvin[4], Razieh Malekhosseini[5]

[1] Department of computer Engineering ,Yas.C., Islamic Azad University ,Yasuj, Iran
Hadi.Tarazodar@iau.ac.ir
[2] Department of computer Engineering ,Yas.C., Islamic Azad University ,Yasuj, Iran
RKa.bagherifard@iau.ac.ir
[3] Department of Electrical Engineering ,Yas.C., Islamic Azad University ,Yasuj, Iran
Sa.nejatian@iau.ac.ir
[4] Department of computer Engineering , NoM.C., Islamic Azad University , Noorabad Mamasani, Iran
Parvin@iaut.ac.ir
5 Department of computer Engineering ,Yas.C., Islamic Azad University ,Yasuj, Iran
Malekhoseini.r@iau.ac.ir

Abstract: Learning from real-time data has been increasingly considered over the past decade. The change in data distribution in online learning, known as concept drift, reduces the accuracy of learning models and makes them ineffective in future predictions. This research aims to design and develop a novel ensemble incremental decision tree algorithm that is capable of detecting concept drift and automatically adapting to changes in data distribution. To achieve this goal, a new architecture of ensemble incremental decision tree is presented that uses an adaptive probabilistic sampling strategy to continuously monitor the pattern of data changes and automatically and in real time performs structural updates in the decision tree. Unlike traditional methods that respond reactively to changes, this approach has an active monitoring mechanism that enables early detection of concept drift by tracking changes in the model error function. In this way, the proposed model is able to maintain high accuracy even in streaming data scenarios with irregular changes. Extensive experiments were conducted on the dataset and the results show that the proposed method performs better than existing methods in several evaluation criteria including accuracy, recall, and precision.

Keywords: Concept Drift, Data Stream, Incremental Decision Tree, Hoeffding, Ensemble Learning

JCDSA, Vol. 3, No. 2, Summer 2025	Online ISSN: 2981-1295	Journal Homepage: https://sanad.iau.ir/en/Journal/jcdsa
Received: 2025-07-09	Accepted: 2025-09-11	Published: 2025-09-21

CITATION

Tarazodar, H., et. al., "A novel and Intelligent Ensemble Framework for Real-Time Detection and Adaptation to Concept Drift in Data Streams Using Incremental Decision Trees", Journal of Circuits, Data and Systems Analysis (JCDSA), Vol. 3, No. 2, pp. 57-72, 2025.
DOI: 10.82526/jcdsa.2025.1211382

* Corresponding author

Extended Abstract

1- Introduction

The rise of real-time data streams in various domains such as finance, healthcare, cybersecurity, and e-commerce has emphasized the importance of adaptive machine learning models. Traditional models, which rely on stationary data distributions, are often inadequate in these settings due to concept drift — a phenomenon where the statistical properties of the target variable change over time. This drift can be sudden, gradual, recurring, or incremental and severely impacts the predictive performance of static models. Existing methods often rely on reactive strategies that adapt only after detecting a significant loss in performance, which may not be optimal in high-speed or highly dynamic data environments.To address these challenges, this paper proposes a novel ensemble-based incremental decision tree architecture designed for both concept drift detection and real-time adaptation. The model enhances traditional Hoeffding Trees by integrating multithreaded learning, adaptive sampling, and active drift monitoring. It aims to maintain high accuracy and robustness while processing data streams with evolving patterns and limited storage capacity.

2- Methodology

The proposed algorithm is structured into five integrated phases, aiming to provide accurate, adaptive, and efficient learning in streaming environments. In the first phase, multithreaded learning is implemented using an ensemble of incremental Hoeffding Trees (HTs), where each thread processes data independently to achieve low-latency updates. The second phase focuses on managing leaf nodes and their counters, which track feature-value-label statistics. These counters enable real-time decision-making, and structural updates are triggered when localized error deviations are detected.The third phase introduces ensemble consensus and adaptive probabilistic sampling. Final predictions are obtained through majority voting across trees, with each tree trained on different data subsets selected via a dynamic sampling strategy. This promotes diversity among models and enhances generalization. The fourth phase incorporates a hybrid concept drift adaptation mechanism, combining proactive monitoring of error trends with reactive structural adjustments. Subtrees exhibiting performance degradation are retrained or replaced automatically to restore accuracy.The fifth phase implements a sliding sample buffer that stores recent data points for efficient reuse. This buffer allows rapid retraining of affected submodels when drift occurs, improving recovery speed without overloading memory resources. Altogether, the five-phase methodology offers a robust, scalable framework capable of maintaining high predictive performance and real-time adaptability in dynamic data stream environments.

3- Results and discussion

The proposed ensemble-based incremental decision tree model was evaluated on widely used benchmark data stream datasets, including SEA, Hyperplane, Electricity, and CovType. These datasets cover various types of concept drift scenarios such as sudden, gradual, and recurring drifts. The model consistently outperformed baseline methods like HAT, FIMT-DD, and ARF-Reg across key metrics including classification accuracy, precision, recall, and drift adaptation latency. The results showed that the proposed approach maintains high predictive performance even as data distributions evolve over time. The multithreaded implementation contributed significantly to runtime efficiency, reducing sample processing time by 30–50% compared to single-threaded models. Adaptive probabilistic sampling improved data diversity within the ensemble, enhancing generalization and robustness. Ensemble majority voting also proved effective in mitigating the impact of noise and irrelevant features, delivering stable predictions under challenging stream conditions. Another key advantage was the sample buffer mechanism, which facilitated quick retraining of underperforming trees without requiring full model resets. This enabled faster recovery from concept drift and maintained model responsiveness in real-time applications. Overall, the results confirm that the integration of parallel learning, adaptive sampling, and drift-aware model updates provides a scalable and resilient solution for data stream classification in dynamic environments

4- Conclusion

This paper proposed an adaptive, ensemble-based incremental decision tree model for effective concept drift detection and adaptation in data streams. The integration of multithreaded learning, adaptive sampling, and buffer-driven retraining led to improved accuracy, fast response to drift, and efficient resource usage. The model proved suitable for real-time, evolving environments such as fraud detection and sensor networks. Future extensions may explore hybrid models and broader data types.

Islamic Azad University , Shiraz Branch
نشریه تحلیل مدارها، داده ها و سامانه ها
Journal of Circuits, Data and Systems Analysis
sanad.iau.ir/journal/jcdsa

یک چارچوب گروهی نوین و هوشمند برای شناسایی و انطباق بلادرنگ با رانش مفهومی در جریان داده‌ها با استفاده از درخت تصمیم‌گیری افزایشی

هادی ترازودار[۱]، کرم اله باقری فرد[۲*]، صمد نجاتیان[۳]، حمید پروین[۴]، راضیه ملک حسینی[۵]

۱- گروه مهندسی کامپیوتر، واحد یاسوج، دانشگاه آزاد اسلامی، یاسوج، ایران (Hadi.Tarazodar@iau.ac.ir)
۲- گروه مهندسی کامپیوتر، واحد یاسوج، دانشگاه آزاد اسلامی، یاسوج، ایران (Ka.bagherifard@iau.ac.ir)
۳- گروه مهندسی برق، واحد یاسوج، دانشگاه آزاد اسلامی، یاسوج، ایران (Sa.nejatian@iau.ac.ir)
۴- گروه مهندسی کامپیوتر، واحد نورآباد ممسنی، دانشگاه آزاد اسلامی، نورآباد ممسنی ، ایران (Parvin@iaut.ac.ir)
۵- گروه مهندسی کامپیوتر، واحد یاسوج، دانشگاه آزاد اسلامی، یاسوج، ایران (Malekhoseini.r@iau.ac.ir)

چکیده: یادگیری از داده‌های بلادرنگ از دهه گذشته به طور فزاینده‌ای موردتوجه قرار گرفته است تغییر در توزیع داده‌ها در یادگیری آنلاین که بنام رانش مفهوم شناخته می‌شود باعث کاهش دقت مدل‌های یادگیری و ناکارآمدی آن‌ها در پیش‌بینی‌های آینده می‌شود. این تحقیق باهدف طراحی و توسعه یک الگوریتم درخت تصمیم‌گیری افزایشی گروهی نوین ارائه شده است که قادر به شناسایی رانش مفهومی و انطباق خودکار با تغییرات توزیع داده‌ها باشد. برای نیل به این هدف، یک معماری جدید از درخت تصمیم‌گیری افزایشی گروهی ارائه شده است که با بهره‌گیری از یک استراتژی نمونه‌برداری احتمالی تطبیقی، الگوی تغییرات داده‌ها را به‌صورت مداوم پایش کرده و به‌روزرسانی‌های ساختاری در درخت تصمیم را به طور خودکار و بلادرنگ انجام می‌دهد. این رویکرد برخلاف روش‌های سنتی که واکنشی به تغییرات پاسخ می‌دهند، دارای یک مکانیزم نظارت فعال است که از طریق رهگیری تغییرات تابع خطای مدل، امکان شناسایی زودهنگام رانش مفهومی را فراهم می‌کند. به‌این‌ترتیب، مدل پیشنهادی قادر است در سناریوهای داده‌های جریانی با تغییرات نامنظم نیز دقت بالایی را حفظ کند. آزمایش‌های گسترده‌ای روی مجموعه‌داده‌ها انجام شد و نتایج نشان می‌دهد که روش پیشنهادی در چندین معیار ارزیابی از جمله دقت، حساسیت و صحت عملکرد بهتری نسبت به روش‌های موجود دارد.

واژه های کلیدی: رانش مفهومی، جریان داده، درخت تصمیم‌گیری افزایشی، هافدینگ، یادگیری گروهی

نوع مقاله: پژوهشی
DOI: 10.82526/jcdsa.2025.1211382
تاریخ ارسال مقاله: ۱۴۰۴/۰۴/۱۸ تاریخ پذیرش مقاله: ۱۴۰۴/۰۶/۲۰ تاریخ چاپ مقاله: ۱۴۰۴/۰۶/۳۱

۱- مقدمه

یادگیری ماشین، به‌عنوان یکی از زیرشاخه‌های مهم علوم کامپیوتر، طی سال‌های اخیر پیشرفت چشمگیری را تجربه کرده است و در کاربردهای متعددی از تشخیص الگوهای پیچیده تا سامانه‌های تصمیم‌گیری مبتنی بر داده مورداستفاده قرار گرفته است [۱]. در حوزه یادگیری ماشین، مدل‌ها به‌طورکلی برای تخمین تابع هدف $\hat{f}:\mathbb{X}\to\mathbb{Y}$ طراحی شده‌اند که در آن $\mathbb{X}$ فضای ویژگی‌ها و $\mathbb{Y}$ فضای برچسب‌ها است. در شرایط ایده‌آل، توزیع داده‌های آموزشی و تست یکسان در نظر گرفته می‌شود، اما در محیط‌های واقعی، این توزیع در طول زمان دستخوش تغییر می‌شود که این پدیده تحت عنوان رانش مفهوم شناخته می‌شود [۲]. تحقیقات موجود روش‌ها و الگوریتم‌های مختلفی را با هدف تشخیص رانش مفهوم و تسهیل انطباق مدل معرفی کرده‌اند. در این میان، الگوریتم‌های مبتنی بر کران هافدینگ [۳] گام‌های قابل توجهی در تنظیمات یادگیری آنلاین برداشته‌اند. با این وجود، یک خلأ آشکار در حوزه مدل‌های رگرسیون مبتنی بر درخت، از جمله درخت‌های رگرسیون و درخت‌های مدل با تشخیص رانش باقی می‌ماند. علاوه بر این، کار تجربی محدودی قلمرو مدل‌های چند هدفه آنلاین را در زمینه جریان‌های داده بررسی کرده است. برای پرداختن به این شکاف‌های مهم و ارائه یک رویکرد جدید برای تشخیص رانش مفهومی و انطباق در جریان‌های داده، یک الگوریتم درخت تصمیم ابتکاری و افزایشی ارائه می‌کنیم. این الگوریتم به طور خاص برای یادگیری درخت‌های رگرسیون و درخت‌های مدل از جریان‌های داده پویا طراحی شده است که با جذب داده‌ها با سرعت بالا و پتانسیل ورود داده‌های نامحدود مشخص می‌شود. رویکرد ما چندین عنصر پیش‌گام را معرفی می‌کند که آن را از روش‌های موجود متمایز می‌کند.

سنگ بنای نوآوری ما در تشخیص فعال رانش مفهوم نهفته است. به‌جای اینکه صرفاً به اشتباهات پیش‌بینی افزایش‌یافته واکنشی نشان

* نویسنده مسئول

دهیم، یک استراتژی جدید پیشنهاد می‌کنیم. ما به طور مداوم کیفیت و عملکرد زیر درخت‌های جداگانه در درخت تصمیم را با ردیابی تحول خطای آنها نظارت می‌کنیم. این امکان تشخیص زودهنگام تغییرات در تابع هدف اساسی را فراهم می‌کند و باعث سازگاری به‌موقع در ساختار مدل می‌شود. کار ما فراتر از تشخیص رانش مفهومی است. ما یک استراتژی نمونه‌گیری تعریف‌شده احتمالی را برای بهبود فرآیند یادگیری معرفی می‌کنیم و آن را در گرفتن اطلاعات مرتبط از جریان‌های داده کارآمدتر می‌کنیم. علاوه بر این، ما یک روش خودکار پیشرفته را ارائه می‌کنیم که قادر به مدیریت برازنده توزیع‌های داده‌های غیر ثابت - یک اتفاق رایج در جریان‌های داده پویا است. از طریق آزمایش و ارزیابی جامع، ما عملکرد برتر الگوریتم پیشنهادی خود را به نمایش می‌گذاریم. ما کارایی آن را از نظر دقت پیش‌بینی، امتیاز فیشر، صحت، اندازه مدل و قابلیت‌های تشخیص تغییر نشان می‌دهیم. در انجام این کار، ما نه تنها یک رقیب قدرتمند برای طبقه‌بندی‌کننده‌های جریان موجود ارائه می‌کنیم، بلکه با مقابله مستقیم با چالش فراگیر رانش مفهومی در جریان‌های داده، سهم قابل‌توجهی در زمینه در حال تکامل یادگیری ماشینی داریم.

بقیه این مقاله به شرح زیر سازماندهی شده است: بخش دوم پیش‌زمینه و کارهای مرتبط ارائه می‌دهد. بخش سوم روش پیشنهادی را توضیح می‌دهد .بخش چهار نتایج ارزیابی را ارائه می‌دهد و به دنبال آن بحث و مقایسه نتایج ارائه می‌شود. در نهایت، بخش پنجم مقاله نتیجه‌گیری و توصیه‌هایی برای تحقیقات آتی ارائه می‌دهد.

۲- پیش زمینه و کارهای مرتبط

رانش مفهوم زمانی رخ می‌دهد که توزیع شرطی داده‌ها $P(Y|X)$ تغییر کند، به‌طوری که مدل پیش‌بینی‌گر دیگر نتواند با دقت اولیه خود عمل کند [۴]. این مسئله در سیستم‌های نظارتی، تشخیص تقلب، بازارهای مالی و بسیاری از حوزه‌های دیگر چالش‌برانگیز است. رانش مفهوم ممکن است ناگهانی، تدریجی، بازگشتی یا افزایشی باشد که هر یک، استراتژی‌های خاصی را برای تطبیق مدل‌ها می‌طلبد [۵]. مقاله حاضر به ارائه رویکردی مبتنی بر درخت تصمیم تطبیقی می‌پردازد که توانایی شناسایی و جبران رانش مفهوم را در محیط‌های پویا دارد. به‌منظور مدیریت رانش مفهوم، لازم است ابتدا انواع آن به‌درستی تشخیص داده شود. بر اساس مطالعات گذشته [۵ و۶]، رانش مفهوم به چهار دستهٔ کلی تقسیم می‌شود:

۱. رانش مفهوم ناگهانی[1]: تغییرات ناگهانی و غیرمنتظره در توزیع داده‌ها، به‌طوری که مدل قبلی بلافاصله ناکارآمد می‌شود:

$$P_t(Y|X) = P_{t-1}(Y|X. P_{t+\epsilon}(Y|X) \neq P_t(Y|X) \quad (۱)$$

۲. رانش مفهوم تدریجی[2]: تغییرات آرام و پیوسته در توزیع داده‌ها که نیاز به به‌روزرسانی تدریجی مدل دارد:

$$P_t(Y|X) \approx P_{t+\epsilon}(Y|X) \; \forall \epsilon \cdot \quad (۲)$$

۳. رانش مفهوم بازگشتی[3]: تغییراتی که پس از یک بازه زمانی به وضعیت قبلی خود بازمی‌گردند، مانند الگوهای فصلی در داده‌های مالی.

۴. رانش مفهوم افزایشی[4]: تغییراتی که به‌آرامی رخ‌داده ولی در بلندمدت منجر به تفاوت‌های قابل‌توجهی در داده‌ها می‌شوند.

مدل پیشنهادی بر مبنای درخت تصمیم تطبیقی طراحی شده است که قابلیت شناسایی رانش مفهوم و به‌روزرسانی ساختار خود را در مواجهه با تغییرات داده‌ای دارد. به‌طورکلی، یک درخت تصمیم را می‌توان به‌صورت مجموعه‌ای از توابع شاخص به‌صورت زیر بیان کرد:

$$\hat{f}(X) = \sum_{j=1}^{M} \mathbb{I}(X \in R_j) f_j \quad (۳)$$

که در آن R_j نشان‌دهنده‌ی نواحی تصمیم‌گیری، f_j مقادیر برآوردشده در هر ناحیه و M تعداد گره‌های درخت است. در صورت وقوع رانش مفهوم، مجموعه‌ی $\{R_j\}$ و یا مقادیر f_j تغییر خواهند کرد. تابع هزینه برای مدیریت این تغییرات به‌صورت زیر تعریف می‌شود:

$$\mathcal{L}(f\hat{}) = \sum_{i=1}^{N} \ell\ \left(y_i . f\hat{}(x_i)\right) + \lambda \sum_{j=1}^{M} \Omega\ \left(R_j\right) \quad (۴)$$

که در آن:

- $\ell(y.\hat{y})$ تابع زیان، مانند زیان مربعات خطا است،
- $\Omega(R_j)$ معیار پیچیدگی ساختار درخت،
- λ وزن کنترل‌کننده تنظیم مدل است.

برای مقابله با این جنبه‌های پیچیده رانش، تشخیص تغییر به‌عنوان یک رویکرد فراگیر و ضروری ظاهر می‌شود. این مستلزم نظارت دقیق بر جریان‌های داده‌های ورودی است. پس از تشخیص یک تغییر، کالیبراسیون مجدد مدل‌ها با هدایت اصل تشخیص تغییر انجام می‌شود. تکنیک‌های مختلف شناسایی این جابه‌جایی‌ها را تسهیل می‌کنند که شامل ردیابی دقیق نرخ خطا و کنار هم قراردادن آنها با سطوح آستانه از پیش تعریف‌شده است. یک نرخ خطا که یک آستانه تعیین شده را نقض می‌کند به‌عنوان یک سیگنال واضح از توزیع داده‌های آشفته عمل می‌کند و مدل را به تطبیق براین‌اساس اشاره می‌کند.

انطباق‌پذیری در زمینه جریان داده‌ها به‌عنوان یک تلاش ظریف جلوه می‌کند که نیاز به تشخیص دقیق و کاهش رانش مفهوم دارد. در حوزه مدیریت رانش مفهوم، تکنیک تشخیص نقش برجسته‌ای را بر عهده می‌گیرد و انطباق برازنده مدل‌های یادگیری ماشین را با چشم‌انداز داده در حال تحول تسهیل می‌کند. اهمیت آن در توانایی آن برای تشخیص تغییرات اساسی در جریان داده است و در نتیجه به فرایند تنظیم مدل کمک می‌کند. پس از شناسایی، این تغییرات مکانیسم تطبیقی مدل را آغاز می‌کند. در قلمرو رانش مفهوم جریان داده، رویکردهای مختلفی در ادبیات موردبررسی قرار گرفته است، از جمله روش‌های طبقه‌بندی مبتنی بر پنجره، مبتنی بر وزن و گروه [۱۱]. برخی از استراتژی‌های جدید، ساختارهای شبکه عصبی را برای مدیریت رانش مفهومی تطبیق می‌دهند، مانند شبکه‌های عصبی فازی در حال تحول عمیق (DEVFNN) [۱۲]، شبکه با ظرفیت تکامل‌یافته پویا (NADINE)

[1]Sudden Drift
[2]Gradual Drift
[3]Recurring Drift
[4]Incremental Drift

[۱۳]) و شبکه عصبی بازگشتی خود تکاملی چندلایه (MUSE-RNN) [۱۴]. بر خلاف این رویکردهای مبتنی بر عصبی، راه‌حل پیشنهادی به طور خاص برای یادگیری درخت تصمیم طراحی شده است. علاوه بر این، روش‌هایی برای پرداختن به پیچیدگی محاسباتی طراحی شده‌اند، مانند شبکه کمک معلم مقیاس‌پذیر تحت نظارت ضعیف (WeScatterNet) [۱۵]. WeScatterNet با استفاده از قابلیت‌های محاسباتی توزیع‌شده پلتفرم آپاچی اسپارک، هم نمونه‌های برچسب‌دار پراکنده و هم جریان‌های داده در مقیاس بزرگ را به طور مؤثر مدیریت می‌کند. بااین‌حال، این امر از طریق غنی‌سازی برچسب نمونه‌های داده با برچسب جزئی به دست می‌یابد، درحالی‌که روش ما منحصراً بر داده‌های برچسب‌دار متکی است.

کومورنیچاک و همکاران [۱۶] گروه تشخیص رانش آماری (sdde)، یک روش جدید برای تشخیص رانش مفهومی را پیشنهاد می‌کند. این روش از اندازه‌های رانش و اندازه‌های رانش متغیر حاشیه‌ای شرطی استفاده می‌کند که توسط مجموعه‌ای از تشخیص‌دهنده‌ها، که اعضای آن بر زیرفضاهای تصادفی ویژگی‌های جریان تمرکز می‌کنند، تحلیل می‌شوند. یک روش تشخیص رانش گروهی (GDDM)[1] برای جریان‌های داده متعدد توسط یو و همکاران [۱۷] معرفی شد. ایده روش از روش تشخیص رانش مبتنی بر نرخ خطا برای یک جریان داده ارث گرفته شده است، به‌عنوان‌مثال، نرخ خطا به‌جای خود داده، متغیر ورودی GDDM است تا تفاوت در تعداد و مقیاس ویژگی‌ها را نادیده بگیرد. در عوض، تفاوت این است که متغیرهای ورودی در GDDM چندمتغیره هستند؛ زیرا نرخ خطای همه جریان‌های داده را هم‌زمان در نظر گرفته شده است. علاوه بر این، برای نادیده‌گرفتن توزیع اساسی جریان‌های داده و همبستگی جریان‌های داده، یک آمار آزمایشی جدید را معرفی کرده است. یک چارچوب نیمه نظارتی به نام CPSSDS[2] توسط تنها و همکاران [۱۸] معرفی شد که از یک طبقه‌بندی افزایشی به‌عنوان یادگیرنده پایه و یک چارچوب خود یادگیرنده برای رسیدگی به کمبود نمونه‌های برچسب‌گذاری شده استفاده می‌کند. آزمون کولموگروف - اسمیرنوف برای تشخیص رانش مفهوم با مقایسه خروجی‌های پیش‌بینی منسجم برای دو دنباله از تکه‌های داده اتخاذ شده است.

یک روش آنلاین نظارت شده مبتنی بر دسته‌ای به نام شبکه پشته سریع و عمیق متوالی آنلاین (OSFDSN)[3] توسط داسیلوا و سیارلی [۱۹] معرفی شد. در این روش شبکه پشته عمیق سریع (FDSN) را به‌عنوان گروهی از شبکه‌های عصبی پیش‌خور تک‌لایه (SLFN) در نظر گرفته که در آن خروجی شبکه، خروجی جدیدترین SLFN است. FDSN متوالی آنلاین (OSFDSN) مشابه FDSN است، اما هر یک از ماژول‌های SLFN آن سهم وزنی در خروجی شبکه دارند. این وزن‌ها به‌صورت پویا و بر اساس جدیدترین داده‌ها محاسبه می‌شوند. روش دیگر مدل تشخیص رانش مفهومی مبتنی بر شبکه کانولوشنال زمانی دوجهته و یادگیری گروه (CD-BTMSE)[4] [۲۰] است. ساختار CD-BTMSE شش یادگیرنده پایه مناسب را برای حل مشکلات بیش از حد، توانایی تعمیم ضعیف و استحکام ضعیف مدل‌های تشخیص رانش مفهومی مبتنی بر یادگیری گروهی انتخاب می‌کند، همچنین از مدل شبکه کانولوشنال موقتی دوجهته (BiTCN)[5] برای بهبود دقت تشخیص استفاده می‌کند. رانش مفهوم از طریق درنظرگرفتن ویژگی‌های زمانی داده‌ها و همچنین معناشناسی دوطرفه در فرایند تشخیص. در همان زمان، از مدل یادگیری گروه برای حل مشکل دقت پایین تشخیص رانش مفهومی ناشی از نرخ خطای تعمیم نسبتاً بالا و توانایی تعمیم ضعیف روش‌های مبتنی بر یادگیری گروهی موجود استفاده می‌کند. اروورا و همکاران [۲۱]. یک رویکرد جدید - گروه انتخابی با استفاده از یادگیری انتقالی (SETL) پیشنهاد کردند که توانایی تطبیق مفهوم جدید داده‌ها را دارد. این رویکرد از یادگیری انتقالی و یک طرح رأی‌گیری اکثریت وزنی برای بهینه‌سازی منابع استفاده می‌کند. همچنین بر مسائلی مانند انتقال منفی و بیش‌برازش که ممکن است در طول فرآیند یادگیری انتقالی رخ دهد، غلبه می‌کند. دنگ و همکاران [۲۲] یک مدل یادگیری گروهی جدید به نام یادگیری گروهی وزن‌دهی شده با نمونه مبتنی بر تصمیم سه‌جانبه (IWE-TWD) پیشنهاد کردند. در IWE-TWD، از یک استراتژی تقسیم و غلبه برای مدیریت رانش نامشخص و انتخاب یادگیرنده‌های پایه استفاده می‌شود. خوشه‌بندی چگالی به‌صورت پویا مناطق چگالی را برای قفل‌کردن محدوده رانش می‌سازد. تصمیم سه‌جانبه برای تخمین اینکه آیا توزیع منطقه تغییر می‌کند یا خیر، اتخاذ می‌شود و نمونه با احتمال تغییر توزیع منطقه وزن‌دهی می‌شود. تنوع بین یادگیرنده‌های پایه نیز با تصمیم سه‌جانبه تعیین می‌شود.

۲-۱- الگوریتم درخت هافدینگ

الگوریتم درخت هافدینگ یک رویکرد یادگیری ماشینی است که برای ساخت کارآمد درختان تصمیم در زمینه جریان داده طراحی شده است [۲۳،۲۴،۲۵]. بر خلاف الگوریتم‌های درخت تصمیم مرسوم که نیاز به عبور چندگانه از کل مجموعه داده دارند، الگوریتم درخت هافدینگ نمونه‌های داده را به‌صورت تدریجی، یک‌به‌یک پردازش می‌کند و به‌تدریج ساختار درخت تصمیم را جمع‌آوری می‌کند. این معیار از یک متریک آماری به نام کران هافدینگ برای تعیین حداقل مقدار موارد موردنیاز برای تصمیم‌گیری قابل‌اعتماد هنگام تقسیم‌بندی یک گره در درخت استفاده می‌کند [۲۳]. این استراتژی انطباق‌پذیری سریع الگوریتم را با تغییرات در توزیع داده تضمین می‌کند و آن را به‌ویژه برای سناریوهایی که با ورود داده‌های پیوسته و با حجم بالا مشخص می‌شوند مناسب می‌کند [۲۵]. در طرف دیگر طیف مربوط به مدیریت داده های جریانی، ما با خانواده الگوریتم های افزایشی مواجه می‌شویم که درخت هافدینگ به عنوان نماینده برجسته‌ای ایستاده است. الگوریتم درختی اصلی

[1] group drift detection method
[2] Conformal prediction for semi-supervised classification on data streams
[3] Online Sequential Fast Deep Stacked Network
[4] Concept Drift detection based on Bidirectional Temporal convolutional network and Multi-Stacking Ensemble
[5] Bidirectional Temporal Convolutional Network

هافدینگ اصول متدلوژی درخت تصمیم را گسترش می‌دهد، یک رویکرد طبقه‌بندی غیرپارامتریک که به‌خاطر سرعت ارزیابی سریع و مهارت آن در مدیریت ویژگی‌های انواع مختلف مشهور است [۲۳]. بااین‌وجود، یک محدودیت ذاتی الگوریتم درخت هافدینگ در این فرض آشکار می‌شود که توزیع داده‌ها در طول زمان بدون تغییر باقی می‌ماند، سناریویی که به‌ندرت در کاربردهای عملی با آن مواجه می‌شویم. برای مقابله با این چالش، همان محققان یک الگوریتم پیشرفته به نام درخت تصمیم‌گیری بسیار سریع مفهومی (CVFDT) را معرفی کردند. CVFDT دارای یک پنجره با اندازه ثابت برای شناسایی گره‌هایی در ساختار درخت تصمیم است که در حال کهنگی (aging) هستند و ممکن است نیاز به به‌روزرسانی داشته باشند، بنابراین الگوریتم را قادر می‌سازد تا با تغییر توزیع داده‌ها منطبق شود [۲۶].

دو الگوریتم مجزا، J48 و HAT، کاربردهایی در یادگیری ماشین و داده کاوی پیدا می کنند. J48 یک الگوریتم درخت تصمیم است که برای کارهای طبقه بندی و رگرسیون استفاده می شود و ویژگی هایی مانند تفسیرپذیری، انتخاب ویژگی، هرس، تطبیق پذیری و قابلیت های گروهی را ارائه می دهد. در مقابل، درخت تطبیقی هافدینگ (HAT) در یادگیری افزایشی در جریان‌های داده، تطبیق با رانش مفهومی، اطمینان از کارایی حافظه، مقیاس‌پذیری و پردازش بلادرنگ از طریق کران هافدینگ تخصص دارد. در حالیکه J48 با داده‌های دسته‌ای مناسب است، HAT در محیط‌های داده‌ای پویا و در حال تحول برتری دارد، و انتخاب آن به داده‌ها و الزامات یادگیری خاص بستگی دارد [۲۶]. وینبرگ و همکاران [۲۷] چارچوب EnHAT را معرفی کرده‌اند که الگوریتم‌های J48 و HAT را ترکیب می‌کند. در این زمینه خاص، J48، که در جاوا منبع باز پیاده‌سازی شده و ریشه در درخت تصمیم C4 دارد، به دلیل مهارت خود در طبقه‌بندی داده‌ها به رسمیت شناخته شده است. در مقابل، HAT یک الگوریتم درخت تطبیقی را مجسم می‌کند که نه تنها اصول روش درخت هافدینگ را در بر می‌گیرد، بلکه الگوریتم ADWIN را نیز برای تسهیل تشخیص تغییرات در جریان داده‌ها ادغام می‌کند. الگوریتم درخت هافدینگ با استفاده استراتژیک از کران هافدینگ برای تعیین اندازه مورد نیاز نمونه‌های آموزشی لازم برای دستیابی به آستانه‌های اطمینان از پیش تعریف‌شده، نقش مرکزی را در بهینه‌سازی ساخت درخت تصمیم ایفا می‌کند. تکنیک گروهی شامل آموزش مدل‌های متعدد بر روی مجموعه‌های داده مختلف و پیش‌بینی با استفاده از یک مدل با بهترین عملکرد یا ترکیبی از مدل‌های با عملکرد برتر است. شایان ذکر است که هر مدل در مجموعه را می‌توان با استفاده از یک الگوریتم متفاوت ایجاد کرد. به عنوان مثال، یک رویکرد گروهی از یک مدل SVM استفاده می کند، همانطور که در [۲۸] ذکر شد روش دیگری که به نام WEAP-I شناخته می‌شود، دو تکنیک گروهی را ترکیب می‌کند: گروه وزنی (WE) که مدل های طبقه بندی فردی را از تکه های داده‌های مختلف تولید می‌کند و از وزن مدل برای ایجاد یک مدل پیش‌بینی واحد استفاده می‌کند. میانگین احتمال (AP)، که طبقه‌بندی‌کننده‌های متعدد را در جدیدترین تکه داده آموزش می‌دهد و به همان اندازه آنها را در یک مدل یکپارچه جمع می‌کند. استفاده از میانگین متحرک وزنی نمایی (EWMA) [۲۹] و ADWIN [۲۶] به‌عنوان تشخیص‌دهنده‌های رانش از جمله روش‌هایی برای معرفی مکانیسم فراموشی به درخت هافدینگ است. ADWIN با ارائه ضمانت‌های عملکرد در مورد میزان خطای به‌دست‌آمده متمایز است و هر دو روش دقت بهبودیافته و کاهش مصرف حافظه را در مقایسه با CVFDT نشان می‌دهند. بااین‌حال، اذعان به این نکته ضروری است که افزودنی‌هایی شامل EWMA و ADWIN به قیمت افزایش سربار محاسباتی بر حسب میانگین زمان پردازش برای هر نمونه ورودی است . این یک مصالحه بین کارایی محاسباتی و حفظ حافظه است که یکی از ملاحظات مهم در حوزه تحلیل داده‌های جریانی است. الگوریتم‌های درخت رگرسیون مبتنی بر هافدینگ و الگوریتم‌های درخت مدل موفقیت در پرداختن به چالش یادگیری مدل‌های رگرسیون از جریان‌های داده پیوسته را نشان داده‌اند. این الگوریتم‌ها باتکیه‌بر اصول روش‌های مبتنی بر هافدینگ که در ابتدا برای طبقه‌بندی توسعه داده شدند، از تخمین‌های احتمالی بر اساس کران هافدینگ استفاده می‌کنند. چندین الگوریتم قابل‌توجه در این دسته عبارتند از FIMT، FIRT-DD و FIMT-DD هستند که در [۹و۳۰] معرفی شد. FIMT یک الگوریتم آنلاین است که به طور خاص برای ساخت درخت‌های مدل خطی از جریان‌های داده ثابت طراحی شده است و از ثبات آماری تصمیمات انتخاب تقسیم‌بندی از طریق استفاده از کران‌های چرنوف اطمینان حاصل می‌کند. نسخه توسعه‌یافته، FIMT-DD، FIMT را با ترکیب قابلیت‌های تشخیص تغییر برای انطباق با جریان‌های داده‌های متغیر با زمان و شناسایی مؤثر تغییرات زمانی، افزایش می‌دهد. FIMT-DD که بر اساس کران هافدینگ است، به‌عنوان یک مرجع حیاتی برای ارائه مفاهیم اصلی در این زمینه عمل می‌کند. قابل‌ذکر است، فرایند استنتاج استقرایی در FIMT-DD بسیار شبیه به روش‌های استنتاج یافت شده در الگوریتم‌های یادگیری درخت تصمیم که ریشه در هافدینگ دارند، با دو تمایز قابل‌توجه است. اولین تمایز کلیدی در رویکرد به‌دست‌آوردن تخمین‌های احتمال از طریق مقایسه توزیع‌های کاندید نهفته است، درحالی‌که تفاوت کلیدی دوم شامل استنتاج افزایشی مدل‌های خطی اضافی در برگ‌های درخت است [۳۰].

درختان مدل سریع و افزایشی (FIMT-DD) [۳۱] درخت‌های رگرسیون افزایشی را مشابه درختان هافدینگ می‌سازد، یعنی FIMT-DD با درختی خالی شروع می‌شود آمار برگ‌ها را از رسیدن به داده‌ها تا رسیدن به دوره مهلت نگه می‌دارد، به‌طوری‌که ویژگی‌ها بر اساس واریانس آنها در مقایسه با متغیر هدف برای تصمیم‌گیری برای تقسیم‌ها رتبه‌بندی شوند، و اگر دو بهترین رتبه حداقل با کران هافدینگ با هم تفاوت داشته باشند [۳]، برگ‌ها تقسیم می‌شوند. FIMT-DD مانند دیگر درختان تصمیم افزایشی، تشخیص رانش مفهومی و انطباق را با تنظیم مجدد دوره‌ای زیر شاخه‌های درخت که در آن افزایش واریانس قابل‌توجهی مشاهده می‌شود، انجام می‌دهد. در جستجوی بهبود عملکرد پیش‌بینی‌کننده درختان رگرسیون افزایشی، یک رویکرد رایج این است که مشابه مدل‌های طبقه‌بندی گروهی چندین درخت را در کنار هم قرار

دهیم، [۳۲]. ایکونوموفسکا و همکاران [۳۳]گروه‌های جنگل تصادفی آنلاین (ORF)[1] و online bagging (OBag) را پیشنهاد کردند که از FIMT-DD به‌عنوان یادگیرنده پایه استفاده می‌کنند. بر اساس آزمایش‌های تجربی، نویسندگان به این نتیجه رسیدند که ORTO-A (درخت‌های گزینه آنلاین با میانگین‌گیری) عملکرد بهتری از OBag و ORF از نظر میانگین مربعات خطا (MSE) دارد. گومز و همکاران [۳۴] رگرسیون جنگل تصادفی تطبیقی (ARF-Reg)[2] را پیشنهاد کردند که اقتباسی از طبقه‌بندی‌کننده جریان داده ARF[3] است [۳۲]. ARF-Reg جنگلی از درختان FIMT-DD را به‌عنوان ORF می‌سازد، تفاوت اصلی بین هر دو الگوریتم این است که ARF-Reg از یک نمونه از الگوریتم ADWIN در هر درخت برای تشخیص رانش‌های مفهومی استفاده می‌کند. حتی اگر شباهت‌هایی بین مدل‌های طبقه‌بندی گروهی و رگرسیون وجود دارد، تفاوت‌های مهمی نیز وجود دارد، به‌عنوان‌مثال، چگونه پیش‌بینی‌ها ترکیب می‌شوند و چگونه تنوع استنتاج می‌شود.

در [۳۵]، نویسندگان به بررسی الگوریتم‌های درخت تصمیم‌گیری برای داده‌های جریانی، چالش‌ها و پیشرفت‌های اخیر در یادگیری درخت تصمیم‌گیری افزایشی، تکنیک‌های پیشرفته برای پردازش سریع و به‌روزرسانی درخت‌های تصمیم و مقایسه الگوریتم‌های مختلف درخت تصمیم‌گیری در محیط‌های جریان داده‌ها پرداخته اند. در [۳۶]، نویسندگان به بررسی درخت‌های تصمیم افزایشی برای پردازش داده‌های جریان بزرگ‌مقیاس پرداخته‌اند. این تحقیق تمرکز ویژه‌ای بر الگوریتم‌های درخت تصمیم‌گیری که برای محیط‌های جریان داده با حجم زیاد و ویژگی‌های متغیر طراحی شده‌اند، دارد. وو و همکاران [۳۷] مقاله‌ای که الگوریتم‌های درخت تصمیم‌گیری سازگار در زمان واقعی را برای جریان داده‌های در حال تحول معرفی می‌کند. این مقاله روش‌هایی را برای شناسایی تغییرات در توزیع داده‌ها و سازگاری درخت‌های تصمیم با تغییرات سریع و پیوسته در داده‌های جریان ارائه می‌دهد. نویسندگان با استفاده از الگوریتم‌هایی مانند کران هافدینگ و تشخیص رانش مفهومی، روش‌های جدیدی برای به‌روزرسانی مدل‌ها در زمان واقعی معرفی می‌کنند. لیو و همکاران [۳۸] یک مدل درخت هافدینگ تطبیقی مبتنی بر آنتروپی تفاضلی و آنتروپی نسبی (AHT-DERE) را برای تشخیص رانش مفهوم پیشنهاد کردند. این مدل یک استراتژی دو مرحله‌ای را اتخاذ می‌کند: الف) یک روش تشخیص رانش مبتنی بر آنتروپی. ب) یک روش تنظیم پویا مبتنی بر آنتروپی نسبی.

۳- روش پیشنهادی

روش پیشنهادی در این مقاله به شناسایی و مدیریت یادگیری از جریان‌های داده می‌پردازد که یک فرایند پویا و بی‌نهایت است جریان‌های داده به طور مداوم نمونه‌ها را به‌صورت بی‌پایان ارائه می‌دهند که به طور بالقوه به طور نامحدود گسترش می‌یابند. محققان مختلف ویژگی‌های کلیدی که یک الگوریتم جریان داده باید داشته باشد را شناسایی کرده‌اند. این ویژگی‌ها عبارت‌اند از:

۱. الگوریتم باید قادر به پردازش تدریجی داده‌ها در زمان رسیدن باشد، به‌طوری‌که به طور مؤثر و سریع بتواند به داده‌های ورودی جدید واکنش نشان دهد.
۲. مدل باید به طور تدریجی و بدون نیاز به بازسازی کل مدل اطلاعات جدید را ترکیب کند که این ویژگی به‌ویژه در مواجهه با تغییرات و رانش‌های مفهومی داده‌ها ضروری است.
۳. الگوریتم باید سرعت همگرایی بالایی داشته باشد تا مدل بتواند به‌سرعت با تغییرات در توزیع داده‌ها منطبق شود.
٤. زمان پردازش برای هر نمونه ورودی باید محدود باشد تا پردازش بلادرنگ و بدون تأخیر برای هر داده جدید امکان‌پذیر باشد.
٥. در حالت ایده‌آل، مصرف منابع (CPU و حافظه) باید ثابت و مستقل از تعداد نمونه‌های پردازش‌شده باقی بماند، که این ویژگی مقیاس‌پذیری و کارایی الگوریتم را تضمین می‌کند.
۶. الگوریتم باید قادر به تشخیص و انطباق با تغییرات در توزیع داده‌ها باشد، به‌ویژه زمانی که تغییرات در توزیع داده‌ها به طور مداوم در حال وقوع است.

در این مقاله، روش پیشنهادی مبتنی بر "درخت‌های تصمیم‌گیری افزایشی" است. این درخت‌ها یکی از روش‌های شناخته‌شده برای پردازش جریان داده‌ها هستند. درخت‌های تصمیم‌گیری افزایشی به‌ویژه برای الگوریتم‌های قدرتمندتر مانند مجموع طبقه‌بندی‌کننده‌های مرکزی استفاده می‌شوند. برای بیان ریاضی مسئله یادگیری از جریان‌های داده با رانش مفهومی، می‌توان از مدل‌های ریاضی مرتبط با تغییرات توزیع داده‌ها و عملکرد مدل‌های یادگیری استفاده کرد. در ادامه، فرمول‌ها و روابط ریاضی برای بیان اجزای مختلف مسئله آمده است. روش پیشنهادی برای یادگیری از جریان‌های داده شامل پنج مرحله اصلی است.

۳-۱- فاز ۱: یادگیری اجماع چند نخی با تاخیر کم برای جریان های داده پویا

این بخش به تقویت درخت هافدینگ (HT) برای یادگیری کارآمد از جریان‌های داده پویا اختصاص دارد. HT یک درخت تصمیم‌گیری افزایشی است که به دلیل توانایی آن برای انطباق سریع و دقیق با جریان های داده شناخته شده است مفهوم اصلی HT بر این ایده استوار است که تجزیه و تحلیل یک زیرمجموعه کوچک و تصادفی از داده‌ها که با S نشان داده می‌شوند، اغلب برای تصمیم‌گیری آگاهانه در مورد گسترش مدل کافی است. این مفهوم به نابرابری هافدینگ بستگی دارد، که تضمینی احتمالی را فراهم می کند که میانگین نمونه مشاهده شده یک متغیر نزدیک به میانگین واقعی آن با احتمال زیاد است. به طور خاص،

[1] Online Random Forest
[2] Adaptive random forest-Regression
[3] Adaptive random forest

برای وظایف طبقه‌بندی باینری[1]، کران هافدینگ را می توان به صورت زیر بیان کرد:

$$P(|\bar{X} - \mu| \geq \varepsilon) \leq 2e^{-2\varepsilon^2 n} \quad (۵)$$

جایی که P احتمال، $\bar{X}$ میانگین نمونه برچسب‌های کلاس باینری است، μ میانگین واقعی (۰.۵ برای طبقه‌بندی باینری)، ε کران هافدینگ (یک مقدار مثبت کوچک)، n تعداد نمونه‌ها در زیر مجموعه S است. این نابرابری نشان می‌دهد که با احتمال زیاد، نرخ خطای مشاهده‌شده یک گره در درخت نزدیک به میزان خطای واقعی آن است. در این زمینه، هدف اولیه تعیین اندازه بهینه زیرمجموعه است که هزینه محاسباتی را به حداقل می‌رساند و درعین‌حال یادگیری دقیق را تضمین می‌کند. این اغلب به‌عنوان "معیار تقسیم" نامیده می‌شود. یکی از روش‌های رایج انتخابی است که کران هافدینگ را به حداقل می‌رساند و اطمینان حاصل می‌کند که زیرمجموعه انتخابی به‌اندازه کافی بزرگ است تا تصمیمات آماری مهمی در مورد تقسیم شود. این معیار از نظر ریاضی به‌صورت زیر بیان می‌شود:

$$|S| = max\left(\frac{4R^2 \ln(2/\delta)}{\varepsilon^2}.1\right) \quad (۶)$$

جایی که R محدوده احتمالات کلاس (۰.۵ برای طبقه‌بندی باینری) است، δ پارامتر اطمینان است که احتمال میزان خطای واقعی را در محدوده هافدینگ تعیین می‌کند. این رویکرد از به‌کارگیری کروهی از طبقه‌بندی‌های افزایشی HT حمایت می‌کند که در مجموع به پیش‌بینی نهایی کمک می‌کنند. در این مقاله، گروهی از طریق ترکیبی از درختان رگرسیون افزایشی تشکیل شده است .

اهداف کلیدی این مرحله شامل پیشرفت‌های الگوریتمی و پیاده‌سازی HT و ترکیبات آن است. هدف افزایش توان عملیاتی و کاهش مصرف منابع در عین حفظ یا افزایش دقت پیش‌بینی است. هدف کلی افزایش کارایی با کاهش تأخیر از طریق اجرای یک مجموعه چند نخی از درختان رگرسیون است. این رویکرد نوآورانه از پتانسیل طراحی حافظه پنهان[2] (کش) بهینه استفاده می‌کند، از قابلیت‌های بردار SIMD[3] (دستورالعمل واحد، داده‌های چندگانه) در واحدهای عملکردی بهره‌برداری می‌کند و از هسته‌های پردازنده‌های متعدد استفاده می‌کند. درحالی‌که سال‌های اخیر شاهد کاوش در موازی‌سازی درخت‌های تصمیم‌گیری و مجموعه‌های طبقه‌بندی بوده‌ایم، راه‌حل‌های موجود در برآوردن نیازهای دقیق جریان‌های داده بلادرنگ کوتاهی می‌کنند. این به‌دقت طراحی شده است تا از سازگاری حافظه پنهان اطمینان حاصل کند، و یک زیر درخت باینری اولیه با یک عمق خاص را در یک خط کش واحد پردازنده L1 فشرده می‌کند. هنگامی که یک گره توسط پردازنده درخواست می‌شود، یک خط کش از L1 شامل یک زیردرخت کامل بازیابی می‌کند. این رویکرد استراتژیک دسترسی به حافظه اصلی[4] را به حداقل می‌رساند و استفاده از حافظه کش را به حداکثر می‌رساند. به شبه‌کد فاز ۱ روش پیشنهادی در شکل (۱) نشان داده شده است.

Algorithm 1: Phase 1 - Low Latency Multithreaded Consensus Learning

Input:
DataStream: Input data stream
Hoeffding Trees: List of incremental Hoeffding Trees
Number of Threads: **N**

Output:
Final Prediction: Consensus prediction

Initialize **N** worker threads

For each worker thread i from **1** to **N** do:
Create a **sample buffer** $_i$
End For
while DataStream is not empty do:
For each worker thread i from **1** to **N** do:
Read next data sample **Data** $_i$ from DataStream
Add **Data** $_i$ to **sample buffer** $_i$
End For
End while
For each Hoeffding Tree **Learner** $_j$ in Hoeffding Trees do:
Train **Learner** $_j$ asynchronously on **sample buffer** $_i$
End For
Wait for all threads to finish training

Initialize Predictions array **Predictions** $_j$ for each **Learner** $_j$

For each **Learner** $_j$ in Hoeffding Trees do:
Predictions $_j$ = Predict(**Learner** $_j$, **Data** $_i$) for each **Data** $_i$ in on **sample buffer** $_i$

End For
FinalPrediction = CombinePredictions (Predictions $_1$, Predictions $_{2,\dots,}$ Predictions $_L$)

Output FinalPrediction

End Algorith

شکل (۱): شبه کد برای فاز ۱ روش پیشنهادی

۳-۲- فاز ۲: برگ‌ها و شمارنده‌ها

در این مرحله، به عبارات جبری حاکم بر برگ‌ها و شمارنده‌ها در رویکرد نوآورانه خود برای یادگیری جریان داده پویا می‌پردازیم. این مؤلفه‌ها در دستیابی به یادگیری مدل کارآمد و دقیق مؤثر هستند. نمادهای ریاضی رسمی را بدین صورت معرفی کنیم:

L : تعداد کل برچسب‌ها، جایی که $0 \leq j < L$

N : تعداد کل ویژگی که در آن $0 \leq i < N$

V_0: تعداد کل مقادیر ویژگی ممکن، که در آن $0 \leq k < V_0$

هر شمارنده مربوط به یک j برچسب، i ویژگی و k مقدار ویژگی خاص است. هدف آن محاسبه وقوع ویژگی i با فرض مقدار k درون برچسب jاست. به طور رسمی، این می‌تواند به‌صورت زیر بیان شود:

$$counter(j.i.k) = \sum(x_{ij} = k) \quad (۷)$$

در اینجا، عملگر $\sum$ جمع‌بندی را نشان می‌دهد و x_{ij} ویژگی i را برای برچسب j در مجموعه داده نشان می‌دهد. عبارت $\sum(x_{ij} = k)$ تعداد کل رخدادها را محاسبه می‌کند که در آن ویژگی i مقدار k درون برچسب j را در نظر می‌گیرد. به‌علاوه، مفهوم شمارش صفات کل را

[1] Binary classification
[2] Cache
[3] Single instruction, multiple data
[4] RAM

معرفی می‌کنیم که به‌صورت $total(j.i)$ است. این تعداد کل ویژگی‌های مشاهده شده i برای یک برچسب j را محاسبه می‌کند. از نظر ریاضی، ما این را به‌صورت زیر نشان می‌دهیم:

$$total(j.i) = \sum(x_{ij}) \quad (۸)$$

در (۸)، $\sum(x_{ij})$ تعداد کل رخدادهای ویژگی i در برچسب j را محاسبه می‌کند. استراتژی سازماندهی حافظه ما برای شمارشگرهای ویژگی شامل ذخیره‌سازی متوالی برای هر برچسب است که هر شمارنده تعداد بیت خاصی را اشغال می‌کند. تکنیک بهینه‌سازی *SIMD* امکان عملیات موازی روی این شمارنده‌ها را فراهم می‌کند. به‌عنوان‌مثال، در رجیسترهای *SIMD*، می‌توان تا ۸ شمارنده را در هر رجیستر جای داد و عملیات را همزمان اجرا کرد. به طور خلاصه، فاز ۲ حول برگه‌ها و شمارنده‌های مدل ما می‌چرخد، که برای دستیابی به کارایی محاسباتی و دقت در یادگیری از جریان‌های داده‌های پویا بسیار مهم هستند. شبه کد فاز ۲ روش پیشنهادی در شکل (۲) نشان داده شده است.

۳-۳- فاز ۳: اجماع درختان رگرسیون افزایشی

در این مرحله، به وظیفه حیاتی دستیابی به اجماع در میان درختان رگرسیون افزایشی برای تشکیل یک طبقه‌بندی دقیق و قابل‌تفسیر می‌پردازیم. هدف ما مهار قدرت چندین درخت و ادغام نتایج آنها در یک مدل پیش‌بینی منسجم و قوی است. ماهیت این مرحله در توسعه الگوریتمی نهفته است که خروجی‌های درختان رگرسیون افزایشی را به طور مؤثر ترکیب می‌کند. همان‌طور که در شکل (۳) نشان داده شده است، این فرایند ایجاد گروه شامل یک مکانیسم تصمیم‌گیری گروهی است. مجموع درختان رگرسیون افزایشی برای تولید نتایج با هم همکاری می‌کنند که متعاقباً با استفاده از یک تابع توافق جمع می‌شوند. در نهایت، خروجی گروهی از طریق طرح رای‌گیری اکثریت تعیین می‌شود. فرایند طبقه‌بندی با عبور از ساختار درخت تصمیم، از ریشه شروع می‌شود و به سمت گره‌های برگ پیش می‌رود. در هر گره میانی، یک تصمیم تقسیم گرفته می‌شود که ما را به سمت چپ یا راست هدایت می‌کند.

برای ارزیابی کیفیت تصمیمات طبقه‌بندی در هر گره، یک تابع زیان را معرفی می‌کنیم که به طور رسمی به‌صورت زیر تعریف می‌شود: اجازه دهید $D = x.y$ درحالی‌که $X = [x_1.x_2.\dots.x_n]$ و $Y = [y_1.y_2.\dots.y_n]$. در اینجا، x_i بردار ویژگی را نشان می‌دهد و y_i نشان‌دهنده برچسب در گره فعلی است که با m نشان داده می‌شود.کاهش خطا $\Delta(y_i)$ با کم کردن ارزیابی تابع زیان پس از تقسیم از ارزیابی تابع زیان قبل از تقسیم محاسبه می‌شود. این را می‌توان به صورت زیر بیان کرد:

$$\Delta(y_i) = \mathcal{L}(y_i) - \left[\frac{n_l}{n}\mathcal{L}(y_l) + \frac{n_r}{n}\mathcal{L}(y_r)\right] \quad (۹)$$

که $\Delta(y_i)$ نشان‌دهنده کاهش خطا هنگام تقسیم در گره m است، $\mathcal{L}(0)$ نشان‌دهنده تابع است و n_l و n_r نشان‌دهنده تعداد نمونه‌هایی است که به‌ترتیب به شاخه‌های چپ و راست هدایت می‌شوند.

Algorithm2: Phase 2 - Leaves and Counters

Input:

L: Total number of labels
N Total number of attributes
V_0 : Total number of possible attribute values

Output:

Leaf Counters: Counters for each label, attribute, and attribute value
Total Attribute Counts: Total counts of attributes for each label

Initialize Leaf Counters and Total Attribute Counts arrays

For each label j from 0 to L-1 do:
  For each attribute i from 0 to N-1 do:
    For each attribute value k from 0 to V_0-1 do:

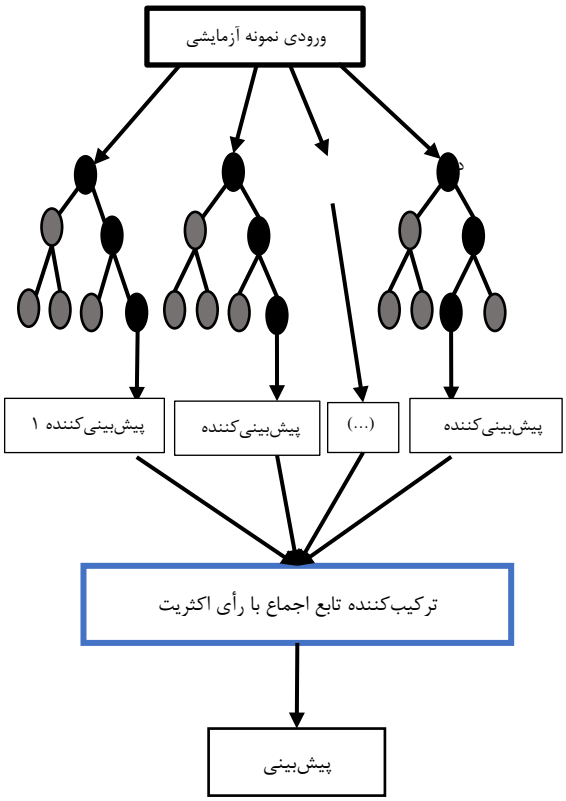
Leaf Counters [j][i][k]=0 // Initialize counters to zero

For each label j from 0 to L-1 do:
  For each attribute i from 0 to N-1 do:
    For each data sample D in the dataset do:
      If D.Lable [i] == j then:
Total Attribute Counts [j] [i] + = 1

For each attribute value K from 0 to V_0-1 do:

If D.attributes [i] == k then:
Leaf Counters [j] [i] [k] + = 1
End Algorithm

شکل (۲) : شبه کد فاز دوم از مرحله پیشنهادی



شکل (۳): مکانیسم اجماع درختان تصمیم‌گیری رگرسیون در روش پیشنهادی

این مرحله علاوه بر توضیح فرایند ایجاد اجماع، مفهوم دستیابی به تنوع در گروه درختان را معرفی می‌کند. رویکرد ما به‌جای تکیه بر زیر-مجموعه‌های متنوع نمونه‌های تولید شده توسط روش‌های نمونه‌گیری، بر ایجاد تنوع در خود گروه تمرکز دارد. این تنوع شامل اعضایی شامل درختان رگرسیون متمایز است که به‌صورت پویا و تطبیقی در زمان واقعی بر اساس ویژگی‌های در حال تحول درخت‌های رگرسیون افزایشی تولید می‌شوند. تنوع طبقه‌بندی‌کننده‌ها در گروه تضمین می‌کند که هر طبقه‌بندی‌کننده پیش‌بینی‌های منحصربه‌فردی را ارائه می‌دهد. به طور معمول، ترکیب چنین طبقه‌بندی‌کننده‌های متنوعی منجر به بهبود عملکرد کلی می‌شود. یک روش متداول برای جمع‌آوری پیش‌بینی‌های اعضای گروه از طریق اکثریت آرا است. در ساده‌ترین شکل، این شامل یک سیستم دموکراتیک است که در آن تصمیم نهایی بر اساس اجماع اکثریت است. بااین‌حال، رویکرد ما با تخصیص وزن‌های مختلف به طبقه‌بندی‌کننده‌های منفرد در گروه فراتر می‌رود و هر کدام را به‌عنوان یک خبره مجزا در نظر می‌گیریم. این مفهوم سنتی رأی اکثریت را گسترش می‌دهد و فرایند تصمیم‌گیری دقیق‌تری را ارائه می‌دهد.

کار [۴۰] مزایای رأی اکثریت وزنی را نشان داد. آنها روش آبشاری[1] را به‌عنوان افزایشی برای رأی‌گیری اکثریت وزنی معرفی کردند. بااین‌حال، این روش‌ها یک گروه بسته را در نظر می‌گیرند که در آن هر یک از اعضای گروه دارای دانش قبلی از همه کلاس‌ها هستند. مشکل "شروع سرد"[2] زمانی پدیدار می‌شود که یک عضو جدید گروه با دانش کلاسی که قبلاً برای اعضای موجود شناخته نشده بود به آن ملحق شود. اکثریت آرای اعضای فعلی ممکن است غالب باشد که منجر به پیش‌بینی‌های نادرست می‌شود تا زمانی که طبقه‌بندی‌کننده‌ها با دانش قبلی از طبقه جدید نفوذ کافی را جمع کنند. برای پرداختن به این موضوع در سناریوهای باز، ما یک رویکرد جدید رأی اکثریت وزنی را پیشنهاد می‌کنیم. روش گروهی پیشنهادی ما به طور خاص با مشکل شروع سرد زمانی که اکثر طبقه‌بندی‌کننده‌ها در گروه فاقد دانش یک کلاس خاص هستند، مقابله می‌کند. الگوریتم ۳ و ۴ که به ترتیب در شکل ۵ و ۶ ارائه شده است، شبه کد این روش ابتکاری را تشریح می‌کند. در این رویکرد، ما از معیار ماهالانوبیس[3] برای اندازه‌گیری فاصله بین نمونه‌های آزمایشی و مرکز توزیع نقطه داده در هر منطقه از مجموعه داده اصلی استفاده می‌کنیم. اگر یک نقطه داده آزمایشی در همان منطقه از فضای داده به‌عنوان طبقه‌بندی‌کننده آموزش‌دیده باشد، در نظر گرفته می‌شود که اطمینان بیشتری در پیش‌بینی‌های خود دارد. در نتیجه، فرایند طبقه‌بندی وزن بیشتری را به منطقه با کمترین فاصله ماهالانوبیس تا نقطه داده آزمون اختصاص می‌دهد. روش پیشنهادی ما با ذخیره ماتریس‌های میانگین و کوواریانس[4] مجموعه آموزشی، کارایی محاسباتی را افزایش می‌دهد که کارآمدتر از ذخیره کل مجموعه‌داده است. پارامتر ε به‌عنوان کسری عمل می‌کند که کاهش وزن را کنترل می‌کند. به طور خلاصه، فاز ۳ بر دستیابی به اجماع در میان درختان رگرسیون افزایشی، توضیح در مورد فرآیند ایجاد اجماع و معرفی استراتژی‌های نوآورانه برای افزایش تنوع، سازگاری و دقت در یادگیری جریان داده‌های پویا تمرکز دارد.

Algorithm 3: Phase 3- Incremental Regression Tree Weighted Majority Vote System
Input: Initialize all weights $w_i = 1$.
Output: A final predictive decision.
For each round:

Given a set of predictions $w_1, w_2, \ldots, w_n$ by experts.

Calculate a new prediction P_i using multiple incremental regression trees (i.w).
Update the history of predictions.

Penalize each mistake made by an expert as follows:
$w_i \leftarrow (1-\varepsilon)\cdot w_i$

End

این الگوریتم فرایند رأی اکثریت وزنی برای یادگیری گروهی را با درختان رگرسیون افزایشی ترسیم می‌کند. در هر دور، پیش‌بینی‌های چند خبره (درخت) را با درنظرگرفتن وزن‌های مربوطه ترکیب می‌کند. وزن‌ها برای جریمه‌کردن اشتباهات و کنترل تأثیر آنها بر تصمیم نهایی تنظیم می‌شوند، جایی که یک کسر از پیش تعریف شده است. پیش‌بینی نهایی از طریق این فرایند تکراری به دست می‌آید.

شکل (۴) : شبه کد برای سیستم رأی اکثریت وزنی درخت رگرسیون افزایشی

Algorithm 4: Multiple Incremental Regression Tree (i, W)

Input: i is the index of a committee member, W is a committee of members.

Output: A probability for the confidence weight of the i -th member.

1. **For** each prediction of the committee with data p :

Compute the Mahalanobis distance D_i from p to the center C_i of the data distribution using the following equation:

$$D_i = \sqrt{(p - C_i)^T \cdot \Sigma_i^{-1} \cdot (p - C_i)}$$

Where, D_i is the Mahalanobis distance for member i , p is the data point for which the distance is being calculated, C_i is the center of the data distribution for member i , Σ_i^{-1} is the inverse of the covariance matrix for member i .

Calculate the inverse Mahalanobis distance h_i for member i :

$$h_i = 1/D_i$$

2. Compute the posterior distribution of the weight $p[R \mid h_i]$ for member i . This can be done using a probability distribution function appropriate for your specific application.

در این الگوریتم، وزن اطمینان را برای عضو i- اجماع درختان رگرسیون افزایشی بر اساس فاصله ماهالانوبیس بین p نقطه داده و مرکز توزیع داده برای آن عضو محاسبه می‌کنیم. سپس فاصله معکوس ماهالانوبیس محاسبه می‌شود که سطح اطمینان را منعکس می‌کند. وزن اطمینان نهایی با مدل سازی توزیع پسین بر اساس این اطمینان به دست می آید.

شکل (۵) : شبه کد درخت رگرسیون افزایشی چندگانه

[1] Cascade
[2] Cold start
[3] Mahalanobis distance
[4] Covariance

۳-۴- فاز ۴: یادگیری گروهی نخی چندگانه

روش ما برای استفاده از قدرت پردازنده‌های چندهسته‌ای مدرن طراحی شده است و درعین‌حال به رانش مفهوم و بهینه‌سازی ترکیب رأی‌گیری می‌پردازد:

۱-استراتژی های مدیریت رانش مفهومی: در رویکرد خود، ما از استراتژی بازنشانی درختی پویا برای رسیدگی به رانش مفهومی استفاده می‌کنیم. هنگامی که تغییر قابل توجهی در توزیع داده ها شناسایی شد، درخت تصمیم را بازنشانی می‌کنیم. این به صورت ریاضی به صورت زیر نمایش داده می شود:

$$If\ Drift\ Detected \Rightarrow Reset\ Tree \quad (۱۰)$$

۲-ترکیب رأی‌گیری:

در طراحی خود، ما یک رویکرد ساده را برای ترکیب رأی‌گیری اتخاذ می‌کنیم که در آن آرای یادگیرندگان تکی با استفاده از وزن‌های مساوی برای همه یادگیرندگان ترکیب می‌شوند. از نظر ریاضی، این به‌صورت زیر بیان می‌شود:

$$Final\ Decision = \frac{1}{L}\sum_{i=1}^{L} Opinion\ of\ learner_i \quad (۱۱)$$

شبه کد فاز ۴ روش پیشنهادی در شکل (۶) نشان داده شده است.

اجرای چند نخی پیشنهادی که در شکل (۷) نشان داده شده است، از مجموع N نخ برای پردازش کارآمد جریان‌های داده استفاده می‌کند.

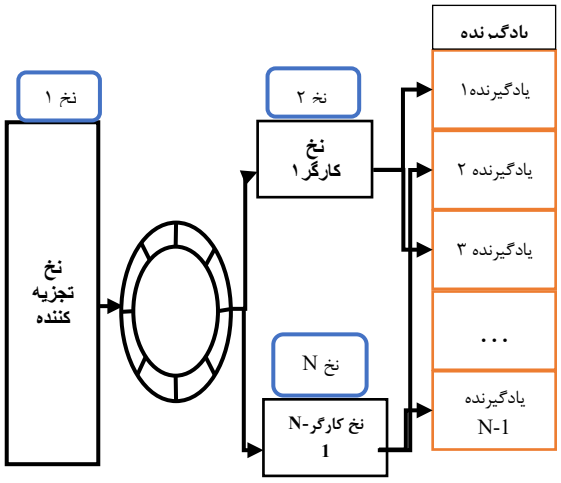
۱-نخ ۱ (نخ تجزیه‌کننده[1] داده): این نخ وظیفه تجزیه ویژگی‌های هر نمونه ورودی و ایجاد یک بافر نمونه را بر عهده دارد. وظایف اصلی آن شامل پیش‌پردازش داده ها و آماده‌سازی داده‌های ورودی برای پردازش بیشتر است.

۲-نخ ۲ تا N (نخ کارگر[2]): این نخها به موازات اجرای گروهی یادگیرندگان کار می‌کنند. هر نخ کارگر پردازش آیتم‌ها را از بافر نمونه انجام می‌دهد. تعداد نخها که با N مشخص می‌شوند، بر اساس تعداد هسته‌های پردازنده موجود یا تعداد نخ‌های سخت‌افزاری پشتیبانی‌شده توسط پردازنده، با در نظر گرفتن بیش از حد نخها در صورت فعال بودن تعیین می‌شود. روش پیشنهادی از یک رویکرد یادگیری گروهی برای معرفی تنوع از طریق تصادفی‌سازی داده‌ها استفاده می‌کند. این پیش‌بینی یادگیرندگان را با استفاده از طرح رای‌اکثریت ترکیب می‌کند.

در الگوریتم، هر یادگیرنده در گروه بر روی زیرمجموعه‌ای از مجموعه داده اصلی آموزش می بیند. این زیرمجموعه با استفاده از تکنیک نمونه‌گیری تکراری ایجاد می‌شود. در شکل (۷)، طرح اجماع یادگیرندگان احتمالاً به مکانیزمی اشاره دارد که در آن چندین یادگیرنده یا مدل برای اتخاذ تصمیمات یا پیش بینی های جمعی با یکدیگر همکاری می‌کنند. با توجه به چند نخی، شکل ممکن است یک رویکرد پردازش موازی را نشان دهد که در آن نخ‌ها یا فرآیندهای مختلف وظایف مجزایی را به طور همزمان انجام می‌دهند و به طور بالقوه کارایی محاسباتی و توان عملیاتی را بهبود می‌بخشند. توضیح اینکه چگونه چند نخی در روش پیشنهادی استفاده می‌شود، مانند موازی‌سازی پردازش داده ها یا وظایف آموزش مدل، وضوح اجرای آن را فراهم می‌کند.

Algorithm 5: Phase 4- Multi-threaded Ensemble Learning
Input:
N: Number of threads
Ensemble: List of L learners
DataStream: Input data stream
Output:
Final Prediction: Ensemble prediction
Initialize N worker threads
For each worker thread I from 1 to N do:
Create a sample buffer $_i$
End For
while DataStream is not empty do:
For each worker thread i from 1 to N do:
Read next data sample Data $_i$ from DataStream
Add Data $_i$ to samble Buffer $_i$
End For
End while
For each **Learner** $_j$ from 1 to L do:
Train **Learner** $_j$ asynchronously on samble Buffer $_i$
Loss = Calculate Loss (**Learner** $_j$, samble Buffer $_j$) // Loss calculation
End For
Wait for all threads to finish training
Initialize Predictions array **Predictions** $_j$ for each **Learner** $_j$
For each **Learner** $_j$ from 1 to L do:
Predictions $_j$ = **Predict (Learner** $_j$ **,** Data $_i$ **)**
FinalPrediction = CombinePredictions (Predictions $_1$, Predictions $_{2,...,}$ Predictions $_L$)
Output FinalPrediction
End For
End Algorithm

شکل (۶) : شبه کد برای فاز ۴ از روش پیشنهادی



شکل (۷) : طراحی اجماع یادگیرندگان و چند نخی

۳-۵- فاز ۵: بافر نمونه

بافر نمونه، یک جزء حیاتی در طراحی اجماع چند نخی، نقشی محوری در بهینه‌سازی پردازش جریان‌های داده پویا ایفا می‌کند. طراحی نمونه بافر ما با الهام از معماری LMAX ، یک بافر حلقه معروف با تأخیر کم، بر اشتراک‌گذاری داده‌ها به طور مؤثر در سراسر نخ‌ها، کاهش اختلاف و افزایش مقیاس‌پذیری تمرکز دارد. در معماری LMAX ، هر نخ دارای یک شماره دنباله منحصربه‌فرد است که برای دسترسی به بافر حلقه

[1] Parser

[2] worker

استفاده می‌شود. برای به‌حداقل‌رساندن مشاجره نوشتن، طرح LMAX از اصل "نویسنده تک" پیروی می‌کند، عملیات اتمی برای دستیابی به اعداد دنباله استفاده می‌شود و حداقل یک معناشناسی پیشرفت را تضمین می‌کند که اغلب در ساختارهای داده بدون قفل مشاهده می‌شود.

طراحی نمونه بافر: شکل (۸) طرح نمونه بافر را نشان می دهد که از بافر حلقه LMAX مدل شده است. Head اشاره‌گر آخرین عنصر حلقه را نشان می‌دهد و تنها در صورتی که شرط $(Head - Tail) < \#samples$ برقرار باشد توسط نخ تجزیه‌کننده داده نوشته می‌شود. این نشان می‌دهد که می توان یک عنصر جدید به بافر اضافه کرد. هر نخ کارگر که با $Worker_j$ نشان داده می‌شود، شماره دنباله $LastProcessed_j$ خود را حفظ می‌کند، که نشان‌دهنده آخرین نمونه پردازش شده توسط آن کارگر است. نخ تجزیه بافر سراسری با استفاده از کمترین مقدار $LastProcessed_j$ در میان همه کارگران، «Tail»را تعیین می‌کند و دسترسی همگام را تضمین می‌کند.

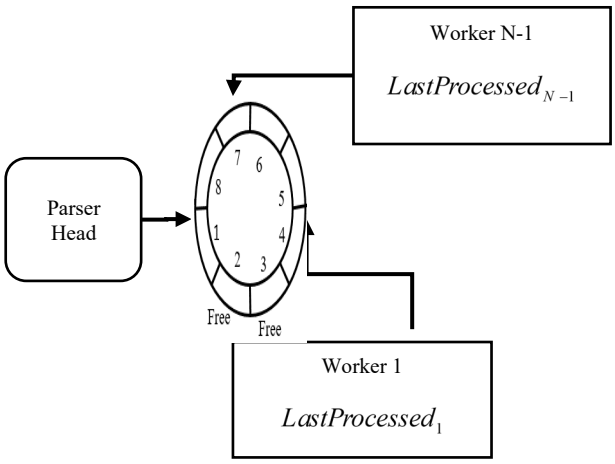
پردازش دسته‌ای: برای به‌حداقل‌رساندن سربار ناشی از عملیات اتمی، طراحی ما به نخ‌های کارگر اجازه می‌دهد تا نمونه‌ها را از بافر حلقه به‌صورت دسته‌ای واکشی کنند. اندازه دسته، بسته به مقادیر $LastProcessed_j$ و $Head$ برای هر کارگر متفاوت است.

در شکل (۸) طراحی بافر نمونه احتمالاً ساختار یا مکانیزم داده‌ای را برای ذخیره و مدیریت نمونه‌ها یا نمونه‌های داده نشان می‌دهد. شفاف‌سازی می تواند مستلزم جزئیات نحوه پر شدن، نگهداری و استفاده از بافر نمونه در الگوریتم باشد.

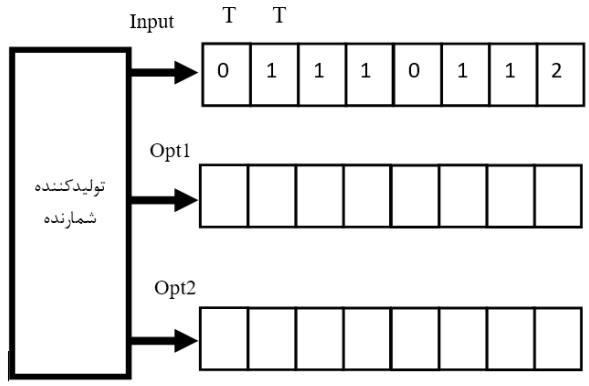
پردازش و ترکیب: ترکیب در روش اجماع پیشنهادی ما مسئول نمونه‌برداری مجدد از موارد بافر، انجام استنتاج تصادفی HT ، و بازنشانی یادگیرندگان در هنگام شناسایی رانش است. به هر نخ کارگر تعدادی یادگیرنده اختصاص می‌یابد که به‌صورت ایستا توزیع می‌شوند و اطمینان حاصل می‌شود که عدم تعادل بار از طریق تصادفی‌سازی در نمونه‌ها و ساخت HT کاهش می‌یابد.

ساختار بافر: هر ورودی در بافر حلقه نمونه ورودی را ذخیره می‌کند و برای هر کارگر یک بافر برای ذخیره خروجی طبقه‌بندی‌کننده‌های اختصاص‌داده‌شده اختصاص داده می‌شود. برای به‌حداقل‌رساندن دسترسی به این بافر، هر کارگر به‌صورت محلی خروجی‌های یادگیرندگان اختصاص‌داده‌شده خود را برای هر نمونه ترکیب می‌کند. هنگامی که تمام یادگیرندگان اختصاص‌داده‌شده به پایان رسید، کارگر نتیجه ترکیب را در بافر می‌نویسد.

شمارنده: دو نوع از تابع شمارنده پیاده‌سازی شده است (شکل (۹)). در $opt1$ ، برچسب هر نماد در جریان ورودی تعداد نمادهایی است که از آخرین صفر (و ۰ هنگام ظاهر شدن نماد صفر) ظاهر شده اند. در $opt2$، برچسب تنها زمانی که یک صفر در جریان ورودی ظاهر می‌شود با ۰ متفاوت است، در این مورد تعداد نمادها از زمان ظهور صفر قبلی برمی‌گردد. جریان ورودی یک توالی تصادفی از نماد صفر و یک است که به‌دنبال یک توزیع طبیعی تولید می‌شود. شبه کد فاز ۵ روش پیشنهادی در شکل (۱۰) نشان داده شده است.



شکل (۸): طراحی بافر نمونه



شکل (۹): تابع تولیدکننده شمارنده

Algorithm 6: Phase 5- Sample Buffer Design

Input:
Number of worker threads: N
Maximum number of samples in the buffer: # samples
Worker thread ID: ID

Output:
Sample buffer operations for each worker

Initialize Head to 0
Initialize LastProcessed [ID] to 0 to 0

while true do:
if (Head-Tail) < # samples then:
Read next data sample **Data** $_i$ from DataStream
Add **Data** $_i$ to Ring Buffer at position **Head**
Head ++
if (Head - LastProcessed [ID]) >=BatchSize then:
Process Batch of Data from LastProcessed [ID] to Head
Update LastProcessed [ID] to head
Perform Random HT Inference on the Batch
Reset Learner on Drift Detection

if Drift Detected then:
Reset Learner and Update LastProcessed [ID] accordingly
End Algorithm

شکل (۱۰) : شبه کد برای فاز ۵ از روش پیشنهادی

۴- نتایج

برای ارزیابی الگوریتم ما از چند مجموعه‌داده مصنوعی و واقعی استفاده می‌کنیم: مجموعه‌داده مصنوعی فریدمن، *Losc* و *Lexp* برای معیار ارزیابی خطا، دقت با استفاده از معیارهای زیر اندازه‌گیری می‌شود: خطای نسبی (*RE*)، ریشه خطای مربع نسبی (*RRSE*) و ضریب همبستگی (*CC*). ما تعداد کل گره‌ها و تعداد گره‌های در حال رشد را بیشتر اندازه‌گیری می‌کنیم. هنگام یادگیری تحت رانش مفهومی، تعداد هشدارهای نادرست، تشخیص اشتباه و تأخیرها را اندازه‌گیری می‌کنیم.

۴-۱- نتایج ارزیابی در سناریو۱

کیفیت مدل‌ها: ازآنجایی‌که ۱۰ مجموعه داده واقعی نسبتاً کوچک هستند، ارزیابی با استفاده از روش اعتبارسنجی متقاطع ده لایه استاندارد انجام شد که در آن لایه‌های یکسان برای همه الگوریتم‌ها استفاده شد. برای انجام یک تحلیل منصفانه و ارزیابی اثر مدل‌های خطی در برگ‌ها، دو مقایسه جداگانه انجام شد. چهار نوع اصلی الگوریتم ما با خود و با سایر الگوریتم‌ها مقایسه شد: نوع اصلی *FIMT-DD* شامل مدل‌های خطی در برگ‌ها و تشخیص رانش است. *FIRT-DD* مدل خطی در برگ‌ها ندارد، *FIMT* تشخیص رانش ندارد و *FIRT* مدل خطی در برگ‌ها و تشخیص رانش ندارد. ابتدا، *FIRT* و *M5'* با گزینه انتخاب رگرسیون *(M5'RT)* مقایسه شد. دوم، *FIMT* با *M5'* با گزینه درخت رگرسیون *(M5'RT)*، روش رگرسیون *LR*، الگوریتم تجاری *CUBIST* و چهار الگوریتم برای یادگیری درختان مدل ارائه شده در دو نوع یادگیرنده دسته‌ای (*RD* دسته‌ای و *RA* دسته‌ای) و دو یادگیرنده افزایشی (*RD* آنلاین و *RA* آنلاین) مقایسه شدند.

ما همچنین از دو نوع *FIMT* استفاده کردیم: *FIMT-const* با نرخ یادگیری ثابت و *FIMT-Decent* با کاهش نرخ یادگیری. نتایج عملکرد یادگیرندگان مدل درختی به طور میانگین در ۱۰ مجموعه داده در جدول (۲) نشان داده شده است. *FIMT-Decent* و *FIMT-const* دقتی مشابه *LR* دارند، در حالیکه بقیه یادگیرندگان بهتر هستند. همچنین نشان داد که بین روش‌های *FIMT* و *RA* آنلاین و دسته‌بندی *RA* و *RD* آنلاین تفاوت معناداری وجود ندارد. تفاوت معنا‌داری در دقت بین *M5'RT*، *CUBIST* و دسته‌بندی *RD* در مقایسه با *FIMT* و *LR* وجود داشت، در حالی که تفاوت معناداری در مقایسه با دسته‌بندی *RA*، *RA* آنلاین و *RD* آنلاین وجود نداشت. همچنین مشاهده شد که مدل‌های خطی در برگ‌ها به ندرت دقت درخت رگرسیون افزایشی را در این مجموعه داده‌های واقعی کوچک بهبود می‌بخشد. این می‌تواند به این دلیل باشد که سطح رگرسیون برای این مجموعه داده ها هموار نیست. بزرگ‌ترین مدل‌ها به طور متوسط توسط *M5'RT* و دسته‌بندی *RD* تولید می‌شوند، یادگیرندگان آنلاین نسبت به همتایان کلاس خود دقت کمتری دارند. روش پیشنهادی مزیت سرعت قابل‌توجهی نسبت به تمامی یادگیرندگان دارد. مزیت دیگر روش پیشنهادی این است که می تواند با حافظه محدود یاد بگیرد، در حالی که سایر الگوریتم‌ها این گزینه را ارائه نمی‌دهند.

جدول (۱): مجموعه داده ۱

مجموعه‌داده	تعداد رکوردها	تعداد متغیر عددی	تعداد متغیر طبقه بندی	تعداد کل طبقه بندی
ABALONE	4.98E3	۸	۱	۳
AILERONS	1.38E4	۴۱	۰	۰
CAL_HOUSING	2.05E4	۹	۰	۰
ELEVATORS	1.66E4	۱۹	۰	۰
HOUSE_8L	2.28E4	۹	۰	۰
HOUSE_16H	2.28E4	۱۷	۰	۰
MV_DELVE	4.10E4	۸	۳	۷
POL	1.56E4	۴۹	۰	۰
WIND	6.57E3	۱۳	۲	۴۳
WINEQUALITY	5.30E3	۱۲	۰	۰
FRIED	1.00E6	۱۰	۰	۰
Lexp	۳۰۰۰۰۰	۵	-	-
Losc	۳۰۰۰۰۰	۵	-	-
Expo	۱۱۶۰۰۰۰۰۰	۱۳	-	-

جدول (۲): نتایج ارزیابی متقابل ۱۰ لایه به‌طور میانگین در مجموعه داده‌های واقعی

Algorithm	RE%	RRSE%	Leaves	Time (s)	CC
M5'RT	42.10	46.09	76.71	29.54	0.85
LR	60.48	63.01	1.00	2.59	0.74
CUBIST	48.75	-	20.67	1.02	0.75
FIMT_Const	60.21	104.01	35.54	0.42	070
FIMT_Decent	59.40	74.11	35.54	0.42	0.73
BachRD	41.84	45.27	55.58	6.13	0.85
BachRA	50.22	53.39	22.85	2.24	0.81
OnlineRD	49.77	53.26	9.98	32.67	0.80
OnlineRA	48.18	52.03	12.78	3.08	0.82
Proposed	**24.63**	**26.85**	**38.12**	**12.25**	**0.97**

جدول (۳): نتایج ارزیابی Holdout به طور میانگین در k۳۰۰/k۱۰۰۰ مجموعه داده ساختگی

Algorithm	RE%	RRSE%	Leaves	Time (s)	CC
FIRT	0.16	0.19	2452.23	24.75	0.98
CUBIST	0.13	-	37.50	104.79	0.98
FIM'T Const	0.11	0.14	2452.23	27.11	0.98
FIMT-Decent	0.11	0.14	2452.23	26.93	0.98
LR	0.46	0.51	1.00	2468.61	0.83
BatchRD	0.08	0.10	27286.30	5234.85	0.00
BatchRA	0.71	0.69	56.97	2316.03	0.51
OnlineRD	0.10	0.13	6579.50	3099.82	0.98
OnlineRA	0.70	0.68	57.77	2360.56	0.53
Proposed	0.20	0.25	42.75	1400.36	0.99

برای مسائل ساختگی ما ۱۰ مجموعه داده تصادفی، هر کدام با ۱ میلیون نمونه، با مجموعه آزمایشی مجزا از ۳۰۰ *k* نمونه تولید کردیم. همه الگوریتم‌ها با استفاده از لایه‌های مشابه آموزش و آزمایش شدند. نتایج جدول ۳ نشان می‌دهد که الگوریتم‌های افزایشی می‌توانند به دقت بهتری نسبت به الگوریتم‌های *CUBIST*، *LR* و *RA* دسته‌ای دست یابند. مدل‌های خطی در برگ‌ها دقت *FIMT* را بهبود می‌بخشد زیرا اکنون در حال مدل‌سازی سطوح هموار هستیم. درختان مدل *FIMT-const* و *FIMT-Decent* دقت مشابهی با *RD* آنلاین دارند. اما از نظر اندازه و زمان یادگیری حداقل ۱۰۰ برابر کوچک‌تر هستند.

نمای انحراف - واریانس: یک ابزار بسیار مفید برای ارزیابی کیفیت مدل‌ها، تحلیل انحراف - واریانس میانگین مربعات خطا است. مولفه انحراف خطا نشانه توانایی ذاتی روش در مدل‌سازی پدیده مورد مطالعه و مستقل از مجموعه آموزشی است. مؤلفه واریانس خطا مستقل از مقدار واقعی متغیر پیش‌بینی‌شده است و تغییرپذیری پیش‌بینی‌ها را با توجه به مجموعه‌های آموزشی مختلف اندازه‌گیری می‌کند. آزمایش‌ها برای همه مجموعه‌های داده واقعی در همان لایه‌های مورد استفاده در ارزیابی متقابل و برای مجموعه داده‌های ساختگی در همان مجموعه‌های آموزشی و همان مجموعه آزمایشی انجام شد.

روش آزمایش به شرح زیر است: ما ۱۰ مجموعه آموزشی مستقل را آموزش می‌دهیم و پیش‌بینی مدل‌های مربوطه را در مجموعه آزمایشی ثبت می‌کنیم. سپس از آن پیش‌بینی‌ها برای محاسبه انحراف و واریانس استفاده می‌کنیم. جدول (۴) نمای انحراف واریانس مدل‌های آموخته-شده توسط *FIRT*، *FIMT-const* ، *RD* آنلاین، و روش پیشنهادی را نشان می‌دهد. نتایج نشان می‌دهد که تغییر پذیری (تنوع) کمتر آن‌ها را در پیش‌بینی‌ها با توجه به مجموعه‌های آموزشی مختلف توجیه می‌کند. این نشان می‌دهد که درختان ساخته شده به صورت تدریجی پایدارتر هستند و کمتر به انتخاب داده‌های آموزشی وابسته هستند.

۴-۲- نتایج ارزیابی در سناریو ۲

در این بخش، از هشت مجموعه داده شامل چهار مجموعه داده واقعی و چهار مجموعه داده مصنوعی استفاده شده است، که مشخصات آن‌ها در جدول (۵) آمده است. در میان مجموعه داده‌های مصنوعی، مجموعه داده SEA به عنوان پایه اصلی مورد استفاده قرار گرفته است. همچنین، برای گسترش مجموعه داده‌ها، سه نسخه تکمیلی از SEA با نام‌های SEA-1، SEA-2 و SEA-3 تولید شده‌اند که ویژگی‌های اصلی مجموعه داده SEA را حفظ کرده‌اند. شکل (۱۱) مقایسه جامعی از نتایج دقت بین روش پیشنهادی و مدل‌ها را با استفاده از بهترین درخت‌ها با اندازه گروهی ۵ در طیف متنوعی از مجموعه‌های داده و اندازه‌های مجموعه داده ارائه می‌کند. تجزیه‌وتحلیل این شکل چندین مشاهدات قابل توجه به دست می‌دهد.

نمودار به طور واضح نشان می‌دهد که روش پیشنهادی به طور مداوم از مدل گروهی با اندازه گروهی ۵ و الگوریتم EnHAT از نظر دقت بهتر عمل می‌کند. شکل (۱۲) نشان‌دهنده برتری مداوم روش پیشنهادی از نظر دقت در مقایسه با مدل گروهی با اندازه گروه ۱۰ است. این روش در مجموعه داده‌ها و اندازه‌های مختلف عملکرد پایداری دارد و دقت بالایی را حفظ می‌کند. نتایج نشان می‌دهد که روش پیشنهادی در طبقه‌بندی داده‌های جریانی حتی با وجود تغییر الگوها نیز عملکرد قابل اعتمادی دارد. این یافته‌ها بر استحکام و کاربرد عملی بالای روش پیشنهادی تأکید می‌کنند.

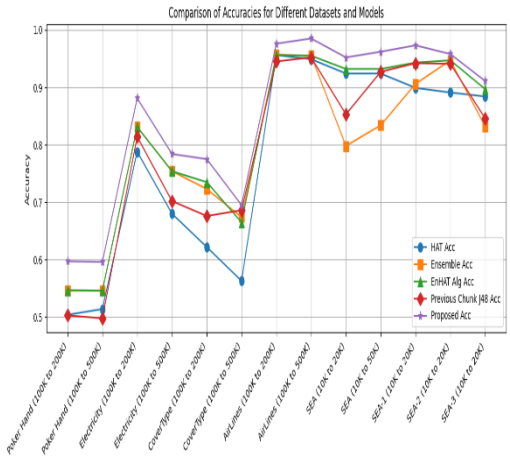
شکل (۱۳) عملکرد روش پیشنهادی را از نظر دقت، حساسیت و معیار فیشر با سایر روش‌ها مقایسه می‌کند. نتایج نشان می‌دهد که روش پیشنهادی در بیشتر مجموعه داده‌ها دقت رقابتی و عملکردی پایدار و قابل مقایسه یا بهتر از الگوریتم‌هایی مانند CUBIST، OnlineRD، FIRT_Const و FIRT دارد. همچنین، میزان پوشش بالا نشان‌دهنده توانایی آن در شناسایی مؤثر نمونه‌های مرتبط است. امتیاز فیشر نیز تعادل مناسب بین دقت و حساسیت را نشان می‌دهد و عملکرد روش پیشنهادی را در مقایسه با سایر روش‌ها برتر یا هم‌تراز نشان می‌دهد. شکل (۱۴) نشان‌دهنده مقایسه روش پیشنهادی با سایر روش‌ها از نظر مقیاس‌پذیری است. نتایج حاکی از برتری مداوم الگوریتم پیشنهادی در معیارهایی مانند دقت، حساسیت و معیار فیشر در مجموعه داده‌های مختلف است. این الگوریتم توانایی بالایی در مدیریت جریان‌های داده پویا و متغیر دارد. همچنین، از نظر سرعت پردازش و مصرف حافظه عملکرد بهتری داشته و برای کاربردهای بلادرنگ بسیار مناسب است

جدول (۴): تجزیه و تحلیل انحراف - واریانس میانگین مربعات خطا برای مجموعه داده‌های واقعی و مصنوعی

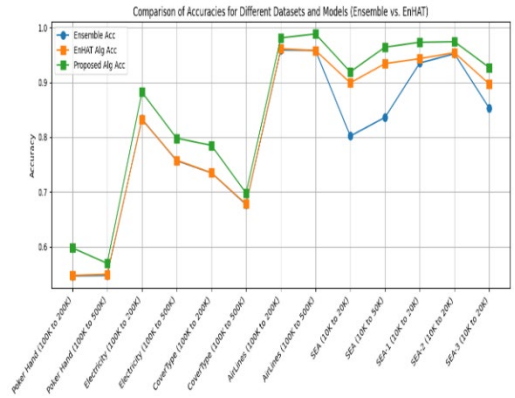
Dataset	FIRT		FIMT_Const		OnlineRD		Proposed	
	Bias	Variance	Bias	Variance	Bias	Variance	Bias	Variance
Abalone	1.19 E+01	0.41 E+01	1.22 @40 1	0.39 E+01	1.17 E+01	0.53 E40 1	0.40 E+01	0.32 E+01
Ailerons	0.00 E400	0,00 E400	1.00 E-06	4.00 E.06	0.00 E+00	0.00 E400	0.00 E+00	0.00 E+00
Mv_delve	9.80 E+01	9.68 E+01	9.83 E+01	9.83 E+01	9.79 E+01	9.71 E+01	8.96 E+01	8.63 E+01
Wind	4.26 E+01	2.64 E+01	4.29 E+01	3.59 E+01	4.20 E+01	3.20 E+01	1.56 E+01	2.98 E+01
Cal_housing	1.27 E+10	7.51 E+09	1.25 E+10	7.75 E+09	1.25 E+10	7.93 E+09	5.15 E+09	0.98 E+08
House_8L	2.58 E+09	1.42 E+09	2.59 E+09	1.55 E+09	2.59 E+09	1.35 E+09	0.96 E+08	1.87 E+09
House_16H	2.61 E+09	1.09 E+09	2.61 E+09	1.21 E+09	2.61 E+09	0.93 E+09	0.90 E+09	1.23 F+09
Elevators	0.38 E-04	0.19 E-04	0.59 E-04	1.98 E-04	0.37 E-04	0.36 E-04	0.15 E-03	0.32 E-03
Pol	1.58 E+03	1.42 E+03	1.58 E+03	1.50 E+03	1.60 E+03	0.96 E+03	0.78 E+03	0.92 E+06
Wine quality	0.72 E+00	0.21 E+00	0.71 E+00	0.31 E+00	0.69 E+00	0.24 E+00	0.12 E+02	0.43 8+01
Fried	1.62 B+00	9.20 E-01	1.40 E+00	4.30 E-01	9.80 E-01	1.30 E-02	1.15 E-03	1.32 E-03
Lexp	2.75 E-02	4.88 E-02	5.57 E-04	1.90 E-03	1.25 E-03	1.05 E-03	1.08 E+03	1.12 E+06
Losc	1.02 E-01	2.18 E-02	9.61 E-02	1.95 E-02	1.90 E-01	1.20 E-05	1.10 E+02	1.13 8+01

جدول (۵): مجموعه داده‌های جریانی مورد استفاده برای ارزیابی تجربی.

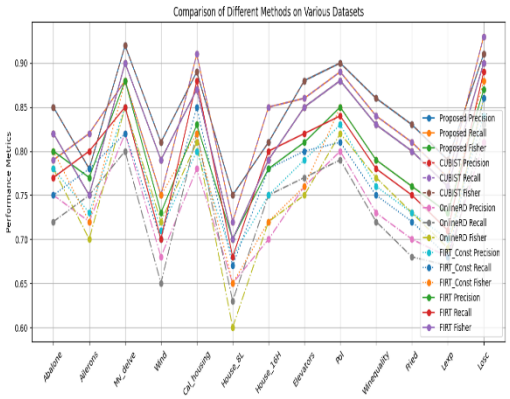
ID	UCI Dataset Name	Samples	Attributes	Classes
DS1	Poker Hand	1,025,010	11	9
DS2	Electricity	45,312	9	2
DS3	CoverType	581,012	54	7
DS4	AirLines	539,383	18	2
DS5	SEA Concepts	60,000	3	3



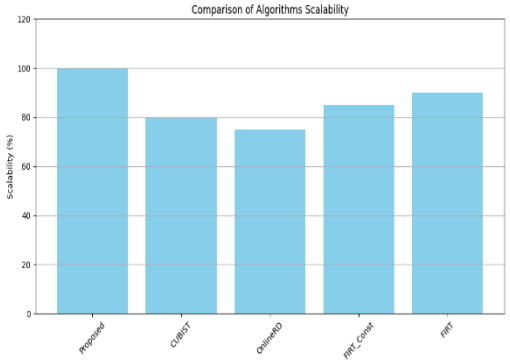
شکل (۱۱): مقایسه بین دقت روش پیشنهادی و مدل‌سازی بهترین درختان با اندازه گروه ۵



شکل (۱۲): مقایسه بین دقت روش پیشنهادی و مدل‌سازی بهترین درختان با اندازه گروه ۱۰



شکل (۱۳): مقایسه روش‌های مختلف در مجموعه داده های مختلف



شکل (۱۴) : فشرده‌سازی مقیاس‌پذیری الگوریتم

۵- نتیجه

در این مقاله، یک الگوریتم افزایشی برای استخراج مدل‌های باکیفیت از مجموعه داده‌های کوچک و نویزدار پیشنهاد شده است. این روش با کاهش نیاز به داده‌های زیاد، مدل‌هایی مشابه مدل‌های مبتنی بر کل داده تولید می‌کند. الگوریتم پیشنهادی شامل پنج فاز است و نتایج شبیه‌سازی، برتری آن را نسبت به روش‌های دیگر نشان می‌دهد. این الگوریتم در طبقه‌بندی داده‌های جریانی از نظر دقت، زمان یادگیری و مقیاس‌پذیری با سایر الگوریتم‌ها مقایسه شده و نتایج حاکی از عملکرد برتر آن، به‌ویژه در شرایط رانش مفهومی و تغییرات داده، هستند.

۱. دقت الگوریتم پیشنهادی در مجموعه داده‌های مختلف از جمله داده‌های جریانی با رانش مفهوم، و داده‌های با نویز به‌طور قابل‌توجهی بالاتر از دیگر الگوریتم‌ها از جمله Online RD و FIRT است. به‌ویژه، در مواجهه با رانش‌های ناگهانی یا تدریجی
۲. زمان یادگیری الگوریتم پیشنهادی نسبت به دیگر الگوریتم‌های موجود، زمان یادگیری کوتاه‌تری داشت.
۳. مقیاس‌پذیری الگوریتم پیشنهادی در مقایسه با دیگر الگوریتم‌ها در مدیریت منابع محاسباتی و حافظه مؤثرتر عمل کرده است.
٤. توانایی الگوریتم پیشنهادی در مدیریت داده‌های غیرثابت از نظر شناسایی و انطباق سریع با این تغییرات.
٥. روش پیشنهادی در مقایسه با دیگر الگوریتم‌ها توانایی بیشتری در شناسایی و کنار گذاشتن نویز و داده‌های متناقض نشان داد.

موارد زیر در کارهای آتی نیز پیشنهاد می‌شود :

۱. پیش‌بینی و شبیه‌سازی رانش‌های پیچیده‌تر با استفاده از الگوریتم‌های مبتنی بر شبکه‌های عصبی و مدل‌های پیچیده‌تر
۲. استفاده از داده‌های مقیاس بزرگ که داده‌ها را به‌صورت موازی و با استفاده از پردازنده‌های گرافیکی و توزیع‌شده پردازش کنند.
۳. افزایش کارایی مدل‌ها در تعاملات پیچیده داده‌ها با توسعه مدل‌هایی که قادر به شبیه‌سازی تعاملات پیچیده داده‌ها باشند مانند روش‌های هوش مصنوعی و الگوریتم‌های مبتنی بر گراف
٤. پیشرفت در تکنیک‌های شناسایی رانش‌های زمان‌بر و دقیق با استفاده از روش‌های شناسایی

مراجع

[1] Quintana, D., Suárez-Cetrulo, L., & Cervantes, A. (2022) "A survey on machine learning for recurring concept drifting data streams." Expert Systems with Applications, 118934. [DOI: 10.1016/j.eswa.2022.118934]

[2] Žliobaitė, R. (2019). Vyresnio amžiaus žmonių informacijos apdorojimo greičio, atminties ir vykdomųjų funkcijų sąsajos su subjektyviais kognityviniais nusiskundimais ir depresiškumu (Doctoral dissertation, Vilniaus universitetas.).

[3] Hoeffding, W. (1994). Probability inequalities for sums of bounded random variables. The collected works of Wassily Hoeffding, 409-426.

[4] Gama, J., P. Medas, G. Castillo, and P. Rodrigues (2004). Learning with drift detection. In SBIA Brazilian Symposium on Artificial Intelligence, pp. 286–295. Springer

for Concept Drift Adaptation. *Tsinghua Science and Technology*, *30*(5), 2029-2047.

[22] Kumar, A., Kaur, P., & Sharma, P. (2015). A survey on Hoeffding tree stream data classification algorithms. CPUH-Res. J, 1(2), 28-32.

[23] Banar, F., Tabatabaei, A., & Saleh, M. (2023, May). Stream Data Classification with Hoeffding Tree: An Ensemble Learning Approach. In 2023 9th International Conference on Web Research (ICWR) (pp. 208-213).

[24] Svoboda R et al (2023) A natural gas consumption forecasting system for continual learning scenarios based on Hoeffding trees with change point detection mechanism. arXiv preprint. arXiv:2309

[25] Gonçalves Jr, P. M., de Carvalho Santos, S. G., Barros, R. S., & Vieira, D. C. (2014). A comparative study on concept drift detectors. Expert Systems with Applications, 41(18), 8144-8156.

[26] Weinberg, A. I., & Last, M. (2023). Enhat—synergy of a tree-based ensemble with hoeffding adaptive tree for dynamic data streams mining. *Information Fusion*, *89*, 397-404.

[27] Ouyang, Z., Zhou, M., Wang, T., & Wu, Q. (2009, November). Mining concept-drifting and noisy data streams using ensemble classifiers. In 2009 International Conference on Artificial Intelligence and Computational Intelligence (Vol. 4, pp. 360-364). IEEE

[28] Lucas, J. M., & Saccucci, M. S. (1990). Exponentially weighted moving average control schemes: properties and enhancements. Technometrics, 32(1), 1-12.

[29] Ikonomovska, E., & Gama, J. (2008, October). Learning model trees from data streams. In International Conference on Discovery Science (pp. 52-63). Berlin, Heidelberg: Springer Berlin Heidelberg.

[30] Ikonomovska E, Gama J, Džeroski S. (2011).Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery* 2011, *23*: 128–168

[31] Gomes, H. M., Barddal, J. P., Enembreck, F., & Bifet, A. (2017). A survey on ensemble learning for data stream classification. ACM Computing Surveys (CSUR), 50(2), 1-36.

[32] Ikonomovska, E., Gama, J., & Džeroski, S. (2015). Online tree-based ensembles and option trees for regression on evolving data streams. Neurocomputing, 150, 458-470

[33] Gomes, H. M., Barddal, J. P., Ferreira, L. E. B., & Bifet, A. (2018, April). Adaptive random forests for data stream regression. In *ESANN*.

[34] Kumar, M., Khan, S. A., Bhatia, A., Sharma, V., & Jain, P. (2023, February). Machine learning algorithms: A conceptual review. In 2023 1st International Conference on Intelligent Computing and Research Trends (ICRT) (pp. 1-7). IEEE.

[35] Zhong, Y., Zhou, J., Li, P., & Gong, J. (2023). Dynamically evolving deep neural networks with continuous online learning. Information Sciences, 646, 119411.

[36] Wu, Y., Liu, L., Yu, Y., Chen, G., & Hu, J. (2023). An Adaptive Ensemble Framework for Addressing Concept Drift in IoT Data Streams. Authorea Preprints.

[37] Liu, Wenzheng, et al. "An Adaptive Hoeffding Tree Model Based on Differential Entropy and Relative Entropy for Concept Drift Detection." 2024 International Joint Conference on Neural Networks (IJCNN). IEEE, 2024.

[38] Gama J, Rocha R, Medas P.(2003). Accurate decision trees for mining high-speed data streams. In: ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. Washington DC: ACM; 2003, 523–528.

[39] Littlestone, N., & Warmuth, M. K. (1994). The weighted majority algorithm. Information and computation, 108(2), 212-261.

[5] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., & Zhang, G. (2018). Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering*, *31*(12), 2346-2363.

[6] Amin, M., Al-Obeidat, F., Tubaishat, A., Shah, B., Anwar, S., & Tanveer, T. A. (2023). Cyber security and beyond: Detecting malware and concept drift in AI-based sensor data streams using statistical techniques. Computers and Electrical Engineering, 108, 108702.

[7] Ko, A. H., Sabourin, R., & Britto Jr, A. S. (2008). From dynamic classifier selection to dynamic ensemble selection. Pattern recognition, 41(5), 1718-1731.

[8] Ikonomovska, E., Gama, J., Sebastião, R., & Gjorgjevik, D. (2009). Regression trees from data streams with drift detection. In Discovery Science: 12th International Conference, DS 2009, Porto, Portugal, October 3-5, 2009 12 (pp. 121-135). Springer Berlin Heidelberg.

[9] Bifet, A., & Gavalda, R. (2009). Adaptive learning from evolving data streams. In Advances in Intelligent Data Analysis VIII: 8th International Symposium on Intelligent Data Analysis, IDA 2009, Lyon, France, August 31-September 2, 2009. Proceedings 8 (pp. 249-260). Springer Berlin Heidelberg.

[10] Xu, Y., Xu, R., Yan, W., & Ardis, P. (2017, May). Concept drift learning with alternating learners. In 2017 International Joint Conference on Neural Networks (IJCNN) (pp. 2104-2111). IEEE.

[11] Pratama, M., Ashfahani, A., & Hady, A. (2019, December). Weakly supervised deep learning approach in streaming environments. In 2019 IEEE International Conference on Big Data (Big Data) (pp. 1195-1202). IEEE

[12] Pratama, M., Pedrycz, W., & Webb, G. I. (2019). An incremental construction of deep neuro fuzzy system for continual learning of nonstationary data streams. IEEE Transactions on Fuzzy Systems, 28(7), 1315-1328.

[13] Das, M., Pratama, M., Savitri, S., & Zhang, J. (2019, November). Muse-rnn: A multilayer self-evolving recurrent neural network for data stream classification. In 2019 IEEE International Conference on Data Mining (ICDM) (pp. 110-119). IEEE.

[14] Pratama, M., Za'in, C., Lughofer, E., Pardede, E., & Rahayu, D. A. (2021). Scalable teacher forcing network for semi-supervised large scale data streams. Information Sciences, 576, 407-431.

[15] Komorniczak, J., Zyblewski, P., & Ksieniewicz, P. (2022). Statistical drift detection ensemble for batch processing of data streams. *Knowledge-Based Systems*, *252*, 109380.

[16] Yu, H., Liu, W., Lu, J., Wen, Y., Luo, X., & Zhang, G. (2023). Detecting group concept drift from multiple data streams. *Pattern Recognition*, *134*, 109113.

[17] Tanha, J., Samadi, N., Abdi, Y., & Razzaghi-Asl, N. (2022). CPSSDS: Conformal prediction for semi-supervised classification on data streams. *Information Sciences*, *584*, 212-234.

[18] da Silva, B. L. S., & Ciarelli, P. M. (2024). A fast online stacked regressor to handle concept drifts. Engineering Applications of Artificial Intelligence, 131, 107757.

[19] Cai, S., Zhao, Y., Hu, Y., Wu, J., Wu, J., Zhang, G., ... & Sosu, R. N. A. (2024). CD-BTMSE: A Concept Drift detection model based on Bidirectional Temporal Convolutional Network and Multi-Stacking Ensemble learning. Knowledge-Based Systems, 294, 111681.

[20] Arora, S., Rani, R., & Saxena, N. (2024). SETL: a transfer learning based dynamic ensemble classifier for concept drift detection in streaming data. *Cluster Computing*, *27*(3), 3417-3432.

[21] Deng, D., Shen, W., Deng, Z., Li, T., & Liu, A. (2025). An Ensemble Learning Model Based on Three-Way Decision