МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЧЕРНІГІВСЬКА ПОЛІТЕХНІКА»
WIN-WIN EDIH



Co-funded by
the European Union



Співфінансується
Європейським Союзом



ВІЛЬНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ В ОСВІТІ, НАУЦІ ТА БІЗНЕСІ

FREE/OPEN SOURCE SOFTWARE IN EDUCATION, SCIENCE & BUSINESS (FOSS 2026)

Матеріали II Міжнародної науково-практичної конференції
01–02 жовтня 2026 р.
м. Чернігів

Чернігів 2026

УДК 004.49:[37+001+334.722]
В41

Рекомендовано до друку вченою радою Національного університету «Чернігівська політехніка» (протокол № 12 від 02.11.2026 р.)

Вільне програмне забезпечення в освіті, науці та бізнесі : матеріали ІІ Міжнар. В41 наук.-практ. конф. (м. Чернігів, 01–02 жовтня 2026 р.) / відп. за вип. Д. Б. Мехед. Чернігів : НУ «Чернігівська політехніка», 2026. 156 с.

ISBN 978-617-8584-09-2

Збірник містить матеріали ІІ Міжнародної науково-практичної конференції «Вільне програмне забезпечення в освіті, науці та бізнесі» (FOSS 2026), що відбулася онлайн 01–02 жовтня 2026 року в Національному університеті «Чернігівська політехніка». Розглянуто питання використання вільного та відкритого програмного забезпечення в освіті, науці, бізнесі, ІТ-інфраструктурі, кібербезпеці, аналізі даних і штучному інтелекті. Матеріали подано українською та англійською мовами.

Для науковців, викладачів, здобувачів освіти, фахівців ІТ-галузі та всіх, хто цікавиться вільним програмним забезпеченням.

Матеріали подано в авторській редакції. Відповідальність за зміст, достовірність фактів, цитат і посилань несуть автори.

Конференцію проведено в межах проєкту WIN-WIN EDIH (грантова угода № 101191647), співфінансованого Європейським Союзом у межах програми Digital Europe. Висловлені погляди та думки належать виключно автору(ам) і не обов'язково відображають позицію Європейського Союзу чи Європейської Комісії.

Відповідальний за випуск: Мехед Дмитро Борисович

ISBN 978-617-8584-09-2

© НУ «Чернігівська політехніка», 2026

Чопорова О. В., Чопоров С. В. ВИКОРИСТАННЯ 3D-МОДЕЛЮВАННЯ ТА АЛГОРИТМІВ НЕЙРОННИХ МЕРЕЖ В НАВЧАЛЬНИХ ПРОЄКТАХ ІНЖЕНЕРНО-ТЕХНІЧНОГО СПРЯМУВАННЯ / USE OF 3D MODELING AND NEURAL NETWORK ALGORITHMS IN EDUCATIONAL PROJECTS OF ENGINEERING AND TECHNICAL DIRECTION 82

Чуб М. В. ВИЯВЛЕННЯ ТА ПРОТИДІЯ АДАПТИВНИМ ВЕБ-АТАКАМ ЗА ДОПОМОГОЮ МОДЕЛЕЙ ГЛИБОКОГО НАВЧАННЯ 86

Шевченко Ю. О. ПОРІВНЯЛЬНЕ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ POSTGRESQL ТА INFLUXDB ДЛЯ ЗБЕРІГАННЯ ДАНИХ ФІЗИЧНОЇ АКТИВНОСТІ / COMPARATIVE STUDY OF POSTGRESQL AND INFLUXDB PERFORMANCE FOR STORING PHYSICAL ACTIVITY DATA 89

Al-Akwaa K. H. M. A. OPEN-SOURCE DATA SCIENCE FOR SUSTAINABLE DEVELOPMENT: A REPRODUCIBLE ANALYTICAL FRAMEWORK 93

Borysenko O. A. AN ARCHITECTURAL APPROACH TO LLM-BASED ROOT CAUSE ANALYSIS FOR CONTAINER ORCHESTRATION INCIDENTS 100

Burmaka I. LOCAL, OPEN, AND ENERGY-EFFICIENT: HARNESSING X86 NPUS FOR EVERYDAY AI TASKS 102

Byvoino T. P., Byvoino P. G. SIMULATION MODELING AND STATISTICAL ANALYSIS OF THE EFFICIENCY OF AN EMERGENCY MEDICAL SERVICE STATION 106

Didusenko O. A., Bula Y. I., Shymchuk G. V. REPRODUCIBLE FOSS FRAMEWORK FOR RESILIENCE EVALUATION OF CONTAINERIZED DISTRIBUTED SYSTEMS 109

Didusenko O. A., Bula Y. I., Shymchuk G. V. REPRODUCIBLE FOSS TESTBED FOR FAULT-TOLERANT MULTIPATH ROUTING IN IP NETWORKS 112

Ebirim V., Jadama A. F. R. RESILIENT DIGITAL EDUCATION IN DISRUPTED ENVIRONMENTS: THE CCDS APPROACH 115

Fraya DIGITAL TRANSFORMATION OF BUSINESS THROUGH FREE AND OPEN SOURCE SOFTWARE: OPPORTUNITIES, BARRIERS, AND STRATEGIC PATHWAYS 119

Kala P. RUN MONGODB YOUR WAY: A PRACTICAL OPEN-SOURCE STACK FOR SELF-MANAGED DEPLOYMENTS 121

Lukin O. A MODERN C++ KMIP CLIENT BASED ON A REUSABLE PROTOCOL LIBRARY 124

Mekhed D. B. DEVELOPING STUDENTS' CYBER HYGIENE SKILLS THROUGH THE USE OF FREE SPECIALIZED SOFTWARE 127

Nahornyi B. I., Lysenok D. V., Savenko A. O. AN OPEN BROWSER-BASED CO-SIMULATOR OF STM32 FIRMWARE AND ELECTRICAL CIRCUITS AS A FREE ALTERNATIVE TO PROTEUS 130

Romanenko M. V. FOSS APPROACH IN MOBILE DEVELOPMENT: CREATING AN OPEN-SOURCE ANDROID APPLICATION WITH THE INTEGRATION OF READY-MADE LIBRARIES FOR STYLISTIC RECOMMENDATIONS 134

Shevchenko N. A., Kibitkin D. S., Shymchuk G. V. REPRODUCIBLE OPEN-SOURCE BENCHMARK OF CONCEPT-DRIFT DETECTORS FOR STREAMING REGRESSION 137

Shevchenko N. A., Kibitkin D. S., Shymchuk G. V. INTERPRETABLE OPEN-SOURCE MACHINE LEARNING FOR NATURAL GAS CONSUMPTION FORECASTING WITH WEATHER AND LAG FEATURES 140

Shivashankar C. ARCHITECTING DATABASE DEPLOYMENTS ON KUBERNETES: PATTERNS, TRADE-OFFS AND EXPERIENCES 143

Shymchuk G. V., Yatskiv P. O. DETECTION OF ANOMALOUS STATES IN IP NETWORKS USING MACHINE LEARNING AND OPEN-SOURCE SOFTWARE TOOLS 146

Shymchuk G. V., Yatskiv P. O. OPEN-SOURCE SOFTWARE ENVIRONMENT FOR MODELING FAULT-TOLERANT MULTIPATH ROUTING IN IP NETWORKS 149

Silva F. HOSTING SOMEONE ELSE'S APP: IFRAME CHOREOGRAPHY IN PMM 152

Šupić A. ZERO-DOWNTIME MIGRATION WITH PERCONA CLUSTERSYNC FOR MONGODB 154

Shevchenko N. A., Kibitkin D. S., Shymchuk G. V.
Ternopil Ivan Puluj National Technical University, Department of Computer Science, Ternopil, Ukraine
Corresponding author: Grigorii Shymchuk, e-mail: gorych@gmail.com, ORCID: 0000-0003-2362-7386

REPRODUCIBLE OPEN-SOURCE BENCHMARK OF CONCEPT-DRIFT DETECTORS FOR STREAMING REGRESSION

Abstract. A reproducible open-source benchmark for detecting concept drift in regression-error streams is presented. The benchmark evaluates three lightweight detectors suitable for online monitoring: Page-Hinkley, one-sided CUSUM, and a Kolmogorov-Smirnov sliding-window detector. Controlled streams contained 2,600 observations with a change point at observation 1,300. Sudden and gradual mean shifts were generated, and each scenario was repeated 100 times with fixed pseudo-random seeds. Page-Hinkley produced the shortest mean detection delay for both sudden and gradual drift (18.0 and 114.8 observations, respectively) but generated 5 false alarms in 100 no-drift runs. CUSUM was slightly slower for sudden drift (20.1 observations) and substantially slower for gradual drift (176.7 observations) but produced no false alarms in the no-drift benchmark. The KS-window detector required 34.4 observations for sudden drift and 167.6 for gradual drift and generated 4 false alarms. The results illustrate the practical trade-off between responsiveness and false-alarm control and provide a compact FOSS methodology that can be integrated with streaming-regression models.

Keywords: concept drift; data streams; online learning; Page-Hinkley; CUSUM; Kolmogorov-Smirnov test; regression monitoring; open-source software.

Introduction

Machine-learning models deployed on data streams are often trained under the assumption that the statistical relation between inputs and targets remains sufficiently stable. In practice, seasonal changes, operating regimes, sensor replacement, market behavior, or changes in the physical process can modify the distribution observed after deployment. This phenomenon is commonly described as concept drift. If drift is not identified, a model can remain numerically operational while its predictive error progressively increases [1,2].

Concept-drift research distinguishes sudden, gradual, incremental, and recurring changes. Recent reviews emphasize that detection and adaptation are separate problems: a detector first indicates that the data-generating process or model error has changed, after which the forecasting system may retrain, reset, reweight, or switch models [2,3]. Open-source ecosystems such as River provide incremental learning and drift-detection components for dynamic data streams [1], while recent regression research continues to use Page-Hinkley and adaptive-window mechanisms for online change detection [4].

The objective of this work is to build a small reproducible benchmark that can be executed with Python, NumPy, and SciPy and later connected to any regression model that produces a sequential error statistic. The benchmark intentionally separates detector behavior from a specific forecasting algorithm: the monitored stream can represent absolute error, squared error, normalized residual magnitude, or another operational score. Three detectors with different statistical logic are compared under identical controlled changes.

Materials and Methods

Each benchmark stream contained 2,600 observations. Before the change point, the monitored statistic followed a normal distribution with mean 0.65 and standard deviation 0.12. For sudden drift, the distribution changed at observation 1,300 to mean 1.00 and standard deviation 0.14. For gradual drift, the mean increased linearly from 0.65 to 1.00 during 350 observations and then remained at 1.00. A no-drift scenario preserved the original distribution for all 2,600 observations. One hundred independent runs were generated for each scenario.

Table 1 – Controlled data-stream scenarios

Scenario	Before change	Transition	After change
No drift	$N(0.65, 0.12^2)$	none	same distribution
Sudden	$N(0.65, 0.12^2)$	at t = 1300	$N(1.00, 0.14^2)$
Gradual	$N(0.65, 0.12^2)$	350-point linear shift	$N(1.00, 0.14^2)$

The Page-Hinkley detector tracks cumulative deviations from an online mean. For an observation x_t, the cumulative statistic was updated as

$$m_t = m_{t-1} + x_t - \bar{x}_t - \delta, \quad PH_t = m_t - \min(m_1, \ldots, m_t). \qquad (1)$$

A drift alarm was generated when PH_t exceeded λ. The benchmark used δ = 0.005 and λ = 7.5 after a 50-observation grace period. CUSUM used the first 300 observations to estimate the baseline mean μ_0 and accumulated only positive deviations,

$$S_t = \max[0, S_{t-1} + x_t - \mu^0 - k], \quad alarm\ if\ S_t > h. \qquad (2)$$

with k = 0.08 and h = 5.5. The third detector used a 160-point moving window, compared the first 80 and last 80 observations with the two-sample Kolmogorov-Smirnov test, and raised an alarm for $p < 10^{-4}$. Detection was checked every 10 observations. Figure 1 illustrates one sudden-drift realization; for this seed, Page-Hinkley, CUSUM, and KS-window signaled at observations 1306, 1318, and 1330, respectively.

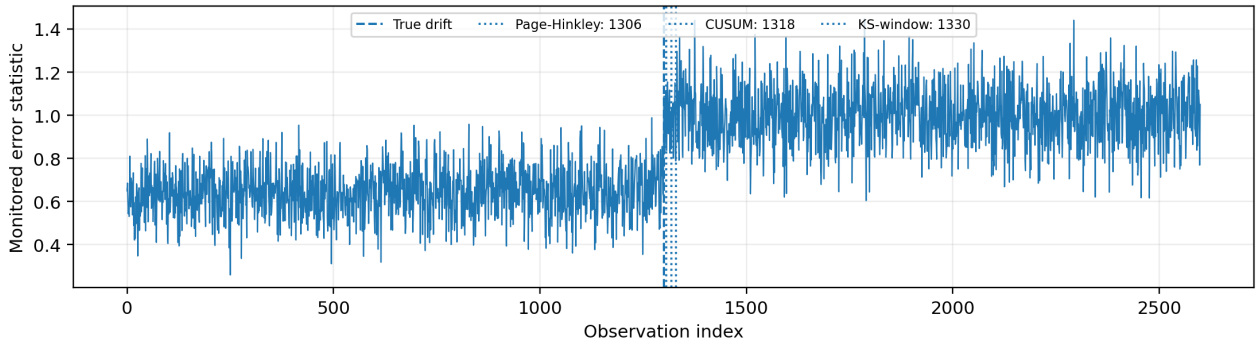

Figure 1 – Example sudden concept drift and detector alarm positions

Two evaluation criteria were used. Detection delay D was measured only for alarms after the true change point, $D = t_{alarm} - t_{change}$. False alarms were alarms before the true change point or any alarm in a no-drift stream. A detector was considered to have detected a scenario if at least one post-change alarm occurred. These measures are widely used because a practical detector should be fast after genuine change but conservative during stationary operation [2,5].

Results

All three detectors identified the injected sudden and gradual shifts in the 100-run benchmark. Table 2 shows the mean detection delay and the no-drift false-alarm count. Page-Hinkley was the most responsive detector, with an 18.0-observation mean delay after sudden drift and 114.8 observations after gradual drift. Its higher sensitivity was accompanied by five false alarms in the no-drift runs. CUSUM required 20.1 observations for sudden drift and 176.7 observations for gradual drift but produced no no-drift false alarms. KS-window was slower than Page-Hinkley for both scenarios and produced four no-drift false alarms.

Table 2 – Detector performance over 100 runs per scenario

Detector	Sudden delay, mean	Gradual delay, mean	Detection rate	False alarms /100 no-drift
Page-Hinkley	18.0	114.8	100%	5
CUSUM	20.1	176.7	100%	0
KS-window	34.4	167.6	100%	4

Discussion

The experiment demonstrates a common drift-monitoring trade-off. Page-Hinkley accumulated evidence rapidly and was therefore effective for both sudden and gradual upward changes, but the same responsiveness increased false-alarm risk. CUSUM was more conservative under the selected parameters: this reduced false alarms but also delayed detection when the change developed gradually. The KS-window detector was distribution-oriented rather than focused specifically on a mean increase, which makes it conceptually more general but less efficient in this benchmark, where the injected drift was dominated by a mean shift.

The results should not be interpreted as a universal ranking. Drift-detector performance depends on change magnitude, noise level, autocorrelation, parameter settings, and the monitored

statistic. Current surveys similarly emphasize that no single detector dominates every stream and that evaluation should reflect the expected operating conditions [2,3,5]. The advantage of the proposed benchmark is reproducibility: stream generators, detector parameters, and metrics are explicit and can be replayed before deployment on real residual streams.

For regression monitoring, an effective workflow is to apply a drift detector to an online error statistic rather than directly to the raw target. This links the alarm to loss of predictive performance. After an alarm, the system can retrain on a recent window, update an incremental model, or activate a backup model. River provides a suitable open-source architecture for this type of streaming workflow [1], while Page-Hinkley-based adaptation has also been evaluated recently in multi-target regression [4].

The main limitation is the controlled Gaussian benchmark. Real errors may be heavy-tailed, autocorrelated, seasonal, and affected by missing data. Future experiments should therefore use residual streams from gas-consumption and network-telemetry models, include recurring and variance-only drift, and compare adaptation policies in addition to alarm quality. Confidence intervals for delay and false-alarm rate should also be reported when the benchmark is expanded.

Conclusions

A reproducible FOSS benchmark of three concept-drift detectors was developed for streaming regression monitoring. Page-Hinkley achieved the shortest mean delay for sudden and gradual changes, CUSUM provided the strongest false-alarm control, and KS-window offered a nonparametric distributional alternative. The benchmark provides explicit stream definitions, detector parameters, and evaluation metrics and can be integrated with online forecasting systems implemented in Python and River.

References

1. Montiel J., Halford M., Mastelini S. M. et al. River: machine learning for streaming data in Python. Journal of Machine Learning Research, 2021, 22(110), 1-8.
2. Arora S., Rani R., Saxena N. A systematic review on detection and adaptation of concept drift in streaming data using machine learning techniques. WIREs Data Mining and Knowledge Discovery, 2024, 14(4), e1536. https://doi.org/10.1002/widm.1536.
3. Hinder F., Vaquet V., Hammer B. One or two things we know about concept drift – a survey on monitoring in evolving environments. Part A: detecting concept drift. Frontiers in Artificial Intelligence, 2024, 7, 1330257. https://doi.org/10.3389/frai.2024.1330257.
4. Stevanoski B., Kostovska A., Panov P., Dzeroski S. Change detection and adaptation in multi-target regression on data streams. Machine Learning, 2024, 113, 8585-8622. https://doi.org/10.1007/s10994-024-06621-z.
5. Lukats D., Zielinski O., Hahn A., Stahl F. T. A benchmark and survey of fully unsupervised concept drift detectors on real-world data streams. International Journal of Data Science and Analytics, 2025, 19, 1-31. https://doi.org/10.1007/s41060-024-00620-y.
6. Hinder F., Vaquet V., Hammer B. One or two things we know about concept drift – a survey on monitoring in evolving environments. Part B: locating and explaining concept drift. Frontiers in Artificial Intelligence, 2024, 7, 1330258. https://doi.org/10.3389/frai.2024.1330258.
7. Virtanen P., Gommers R., Oliphant T. E. et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods, 2020, 17, 261-272. https://doi.org/10.1038/s41592-019-0686-2.
8. Page E. S. Continuous Inspection Schemes. Biometrika, 1954, 41(1/2), 100-115. https://doi.org/10.1093/biomet/41.1-2.100.