Выявление аномалий при обработке потоковых данных в реальном времени

Д.Е. Савицкий, М.Е. Дунаев, К.С. Зайцев

Аннотация **- Целью настоящей работы является исследование методов выявления аномалий при обработке потоковых данных в распределенных системах в режиме реального времени. Для этого авторами предлагается модификация алгоритма K-Means, названная Real-Time K-Means, и проведен сравнительный анализ эффективности разработанной модификации алгоритма с K-Means из библиотеки MLlib фреймворка Apache Spark. Сравнение подтвердило эффективность предложенной модификации. Для проведения экспериментов с алгоритмами был построен специальный массив данных (датасет), включивший около 1000 измерений метрик лога работы сервера Apache Kafka с одной темой, двумя поставщиками и одним потребителем. В этот датасет были добавлены аномальные фрагменты, с большим числом сообщений в секунду и/или размером сообщения. Значения датасета были предобработаны для выравнивания влияния метрик и исключения корреляций. Результаты применения разработанной авторами модификации алгоритма K-Means при решении задач поиска аномалий в реальном времени показали его эффективность.**

Ключевые слова **– поиск аномалий, режим реального времени, распределенная обработка, метрики журнала событий, машинное обучение, кластеризация, статистические методы, Apache Spark.**

I. ВВЕДЕНИЕ

В настоящее время каждая крупная IT-компания работает с десятками или даже сотнями сервисов, развернутых одновременно на своей платформе. Единственным надежным источником информации, характеризующей состояние приложений, являются журналы событий или логи (logs) сервисов – наборы метрик, представленных временными рядами (time series) значений. По значениям метрик логов с помощью методов машинного обучения можно отслеживать различные сбои в работе сервисов, попытки вмешательства в работу приложений, акты мошенничества, выявлять и классифицировать аномалии поведения сервисов. Получаемые сведения снижают риски отказов, помогают в управлении сервисами при принятии решений.

Для обнаружения аномалий обычно используют такие алгоритмы машинного обучения, как K-Means, KNN, OPTICS [1,2,3].

Эти методы машинного обучения, бесспорно, могут решать обозначенные выше задачи выявления аномалий поведения, и активно используются для различных направлений деятельности. Однако в исходном виде все они не пригодны для работы с непрерывными потоками больших данных высокой интенсивности в реальном масштабе времени. Особенно ярко это проявляется при работе с данными, имеющими значительную изменчивость. Стандартный подход для использования алгоритмов управляемого обучения (supervised learning) требует сначала обучить модель на заранее подготовленной обучающей выборке, затем протестировать корректность предсказаний на тестовой выборке и, наконец, интерполировать результаты на более широкое множество реальных данных. При работе с временными рядами, такой подход можно применять с серьезными ограничениями, так как, во-первых, данные временных рядов жестко упорядочены, и во-вторых эти данные непрерывно меняются во времени. Поэтому использовать модели, ранее обученные на старых данных – часто не разумно. Примером может служить невозможность использования модели, построенной на выборке нагрузки интернет-магазина, ориентированного на трафик в 2000 пользователей в день, после кратковременного увеличения трафика до 3000 клиентов в день из-за рекламной акции. Меняется трафик и, возможно, социальный статус покупателей с другими потребностями, поэтому использование ранее построенной модели даст в лучшем случае неточные результаты, а в худшем – совсем неверные.

Можно пытаться переучивать модель через определенные интервалы времени. Однако такое решение малоэффективно из-за того, что обрабатываемых данных, как правило, очень много и одновременно обычно развернуто сразу несколько экземпляров моделей для работы с разными временными рядами. Поэтому процесс

частого изменения модели связан с высокими дополнительными затратами и необходимостью контролировать процесс очередного обучения. Но, при высокой волатильности данных можно применять только такие алгоритмы, которые переобучаются быстрее, чем поступает очередная порция данных обучающей выборки из потока. Про такие алгоритмы принято говорить, что они обучаются “на лету”. Класс подобных моделей принято относить к области online learning. Для скоростной высокоинтенсивной работы удобно использовать фреймворки, ориентированные на распределенные вычисления при работе с большими данными. Одним из наиболее популярных фреймворков является Apache Spark, предназначенный для распределенной обработки структурированных и неструктурированных данных [4, 5].

В настоящей статье предлагается для решения задачи выявления (детекции) аномалий в работе сервисов технологических платформ в реальном масштабе времени по метрикам журналов событий в среде фреймворка Apache Spark использовать модификацию алгоритма K-Means, т.н. Real-Time K-Means, которая показала свое превосходство в сравнении с алгоритмом с K-Means, реализованным в Apache Spark.

II. АЛГОРИТМЫ ВЫЯВЛЕНИЯ АНОМАЛИЙ

Рассмотрим два алгоритма, способных выделять аномалии, K-means и Streaming K-means.

a) K-means. Задачу детектирования аномалий можно рассматривать как задачу кластеризации. Пусть имеется n кластеров нормального поведения и один – аномального поведения. Позднее в работе будет приведен метод его определения. На первом этапе на исторических поведенческих данных строим модель, которую оценим, исходя из предположений экспертов. На втором этапе в режиме онлайн будем измерять расстояния для появляющихся в журнале событий новых данных, определяя их аномальность. При этом будем периодически обновлять модель.

K-means - это неиерархический и итерационный алгоритм машинного обучения, решающий задачу кластеризации. Он получил большую популярность, благодаря своей простоте, наглядности реализации и достаточно высокому качеству получаемых результатов.

Основная идея алгоритма заключается, как известно, в том, что на очередном i-ом шаге итерационной процедуры заново вычисляются центры масс каждого кластера, полученного на предыдущем i-1-ом шаге, затем векторы разбиваются на кластеры вновь в соответствии с тем, какой из новых центров оказался к ним ближе по выбранным метрикам.

Алгоритм завершается, если на очередной итерации не происходит изменения внутрикластерного расстояния. Число таких итераций - конечно, так как количество возможных разбиений конечного множества - конечно, и на каждом шаге суммарное квадратичное отклонение уменьшается. Поэтому зацикливания не произойдет.

Для определенности приведем псевдокод этого алгоритма:

1. Задать начальное положение центров кластеров μ_y, y∈Y.
2. Отнести каждый x_i к ближайшему центру:

$$i := argminp(x_i, \mu_y), i = 1, \dots, k, \quad (1)$$

где x_i, y_i - координата и кластер, к которому следует отнести объект (точку) i.

3. Вычислить новые положения центров, как среднее значение координат точек, отнесенных к кластеру i:

$$\mu_{yj} := \frac{\sum_{i=1}^{l}[y_i=y]f_j(x_i)}{\sum_{i=1}^{l}[y_i=y]}, y \in Y, j = 1, \dots, n \quad (2)$$

Повторять шаги 2 и 3, пока состав кластеров y_i не перестанет изменяться [6].

b) Streaming K-means. Этот алгоритм взаимодействует с потоковыми данными, обновляя мета-информацию о кластерах динамически по мере обновления данных с возможностью контроля забывания старых данных (параметр α в формуле 3) [7]. Алгоритм использует обобщенные правила обновления K-Means кластеров небольшими пакетами данных. Для каждого пакета данных каждой точке присваивается ближайший кластер, и затем вычисляются новые центры кластеров

$$c_{t+1} = \frac{c_t n_t \alpha + x_t m_t}{n_t \alpha + m_t}, \quad (3)$$

$$n_{t+1} = n_t + m_t, \quad (4)$$

где c_t предыдущий центр кластера, n_t - количество точек в кластере t на текущий момент времени,

x_t - новый центр кластера, составленный из точек, поступивших в пакете и отнесенных к кластеру t,

m_t - количество точек, добавленных к кластеру.

Критерий затухания α, заданный в [0, 1], используется для регулирования степени “забывания” прошлых значений. Так, при α=1 все данные с начала наблюдения будут использованы, при α=0 все данные, кроме последних, будут отброшены. Этот параметр аналогичен

экспоненциально взвешенному скользящему среднему.

Затухание может также быть задано параметром halfLife (период полураспада), который определяет корректный коэффициент затухания α, так чтобы для данных, полученных в момент времени t, их вклад к моменту t + halfLife снизился до 0,5.

III. ПОДГОТОВКА ДАННЫХ

Для тестирования и сравнения результатов работы построенных моделей локально была развернута система с архитектурой, представленной на рис. 1.

Для подготовки экспериментальных данных используются метрики логов продукта с открытым кодом Apache Kafka. В качестве производителей (producers) Kafka выступали написанные на высокоуровневом языке программирования Python программы, посылающие сообщения с определенной периодичностью в заранее заданные темы (topics) Kafka.

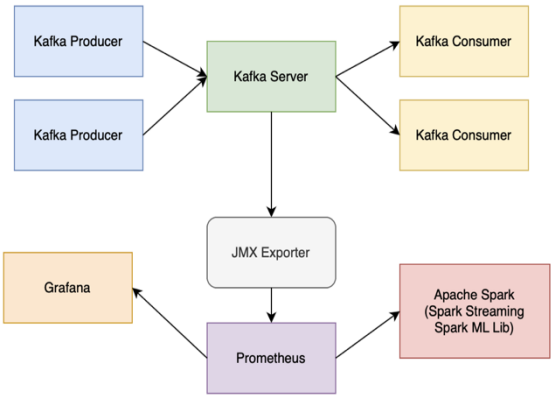

Рис. 1. Архитектура системы.

Эти же программы генерировали “аномалии”, представленные потоком сообщений с количеством сообщений в секунду, превышающим среднее значение в 2,5 тысячи раз по сравнению с аналогичной метрикой для любых других временных отрезков (рис. 2).

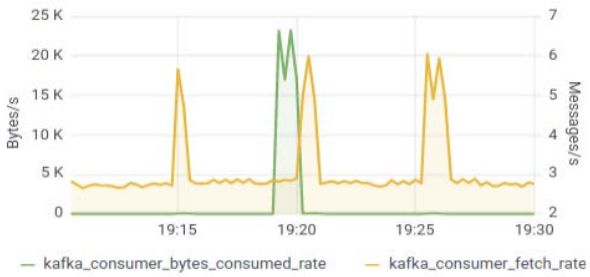

Рис. 2. Фрагменты метрик Apache Kafka с аномалиями в Grafana.

В подготовке данных использовались два производителя.

Первый производил сообщения, длиной 70 байт и отправлял:

- одно сообщение раз две секунды,
- два сообщение раз в десять секунд,
- пять сообщений каждую пятидесятую долю секунду,
- сто сообщений каждые пять минут и двадцать секунд.

Второй производил сообщения переменного размера, отправлял одно сообщение каждые две секунды размером 70 байт, и генерировал два аномальных сообщения размером 960 килобайт.

Получаемый датасет представляет собой набор из более чем двух тысяч временных рядов - метрик, собранных JMX-экспортером в количестве 1000 строк. Часть метрик, некоррелированных или слабо коррелированных с количеством сообщений в секунду, таких как jvm_threads_state, jmx_scrape_duration_seconds была исключена из датасета. Данные были нормализованы и смещены относительно среднего тренировочной выборки.

В настоящем исследовании не рассмотрены вопросы появления новых метрик (при заведении новой темы в Kafka) и иррелевантности старых метрик (при удалении тем или отдельных потребителей/производителей Kafka) потому, что они имеют исключительно технический характер.

В реальных условиях данные будут поступать из сервисов мониторинга работы приложений в реальном времени, однако для исследовательских целей можно использовать заранее подготовленный датасет, поступающий в потоковом режиме, например, с помощью Apache Streaming.

IV. МОДИФИКЦИЯ МОДЕЛИ K-MEANS

В библиотеке Spark MLlib присутствует потоковая модель Streaming K-Means, которую

можно использовать для стриминговой кластеризации данных. Однако в нее нельзя загрузить заранее подготовленную модель, поэтому работу она начинает с холодного старта. Изменить это можно созданием специального адаптера (рис. 2) между обычной моделью K-Means, которая используется для предобучения модели на некотором объеме исторических данных, и написанной заново моделью K-Means Streaming, работающей по формулам (3 и 4).

Модификация K-Means реализована следующим образом: после предобучения на модели K-Means, модель K-Means Streaming подгружает метаданные модели K-Means, такие как количество кластеров и координаты их центров. Затем, используя датасет, на котором происходило обучение модели, K-Means Streaming определяет количество точек в каждом кластере и сохраняет их.

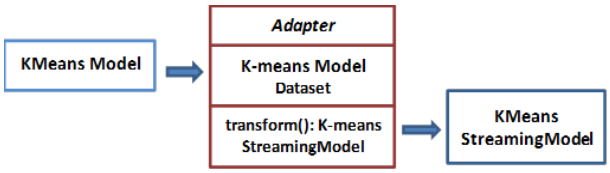

Рис.3 Схема классов реализованного адаптера между KMeans и Streaming Kmeans.

V. РЕЗУЛЬТАТЫ СРАВНЕНИЯ МОДЕЛЕЙ

Было проведено сравнение эффективности выявления аномалий моделями K-Means Streaming библиотеки MLlib фреймворка Apache Spark и предложенной авторами ее модификации, названной Real-Time K-Means. Эксперимент проводился для трех кластеров, определенных по методу локтя.

Встроенная в Spark MLlib Streaming модель K-Means обучается на тренировочном датасете в потоковом режиме, и затем может использоваться для реальных данных. Реализованная авторами ее модификация действует иначе: модель принимает предобученную классическую (не real-time) модель K-Means, точность которой выше за счет дополнительной настройки модели и оптимизации параметров, невозможных в потоковом режиме. Далее, используя предложенный адаптер (рис. 3), можно получить real-time модель Streaming K-Means, работающую с реальными данными. Таким образом, достигается более гибкая настройка параметров модели.

В результате из-за разных схем обучения, центры кластеров этих двух моделей значительно отличаются координатами, что объясняется большим количеством итераций в стандартной версии K-Means, по сравнению со Spark MLlib Streaming K-Means, в которой акцент сделан на скорости вычислений.

Рассмотрим результаты определения координат центров кластеров для классической K-Means и Spark MLlib Streaming K-Means, построенных на двух метриках журнала событий, напрямую определяющих аномальность события:

- 'message rate' - количество сообщений в секунду,
- 'kafka_consumer_bytes_consumed_rate' - число байт в секунду, получаемых потребителем.

Для моделей, полученных стандартным алгоритмом K-Means, центры кластеров при одних и тех же исходных данных и числе итераций больше 10 практически совпадают. Для модели Spark MLlib Streaming K-Means центры моделей, исходно получившие одинаковые данные, могут значительно отличаться координатами.

Рассмотрим результаты работы модели Spark MLlib Streaming K-Means по определению координат центров трех кластеров для двух случаев обучения модели.

Первый случай:

№ Кл	Первая координата	Вторая координата
1	7.074178082686644	-0.1179546559097857
2	0.699246487204938	5.296479067828162
3	-0.083346635154719	-0.18317698019505607

Второй случай:

№ Кл	Первая координата	Вторая координата
1	8.840045919479378	0.5943960022854669
2	-0.08328441646883958	-0.005599962336461511
3	-0.08328441646885959	-0.005599962336481511

Легко видеть, что координаты центров кластеров отличаются друг от друга, иногда правда в 13 знаке после точки, т.е. точность модели различается при разных тестах.

Характеристики качества сравниваемых моделей на тестовом датасете дали следующие результаты:

Модель	Характеристики качества
Модификация Streaming K-Means, предобученная на K-Means	TPR = TNR = PPV = = NPV = 1
Spark MLlib Streaming K-Means	TPR в пределах 0.90 – 0.95, TNR = 1 = PPV = 1, NPV = 0.99

Здесь TPR – чувствительность, TNR – специфичность, PPV – точность, PNV – доля ложных пропусков [8].

Падение показателей качества для второй модели вызвано недетерминированностью центров кластеров после обучения Spark MLlib Streaming K-Means на тестовой выборке.

VI. ДИСКУССИЯ ПО ТЕМЕ ИССЛЕДОВАНИЙ

Сегодня стандартом в индустрии являются модели, предобученные на выборках из данных, поступающих из производственной среды. Для повышения достоверности ответов модели, используемые для задач обработки временных рядов, обучаются заново через некоторые промежутки времени [9]. Такой подход вызван не развитостью моделей реального времени, является временным и не решает задачу системно.

Большая часть работ по решению задачи обнаружения аномалий основывается на моделях, предобученных на конечных выборках. И априори в предсказаниях подобных моделей заложена погрешность, так как в процессе работы зачастую возникает необходимость переобучать модели. Авторы ряда статей, например [10], указывают на чувствительность Kmeans к выбросам значений, которые ограничивают его использование для поиска аномалий, советуя при этом использовать OTPICS. Очевидно, однако, что выбор моделей в сильной степени зависит от анализируемых данных. Мы попытались протестировать улучшенный OPTICS (DBScan) на своих датасетах, но результаты были близкими плачевными, будем исследовать RealTime-OPTICS (real time) [11].

Применение нейронных сетей же демонстрирует хороший результат, однако такие модели занимают много времени для переобучения, поэтому менее пригодны для работы в реальном времени при выявлении аномалий. Также результаты работы нейронных сетей трудно интерпретировать человеку [12]. Значительный интерес сегодня представляют исследования возможностей создания модификаций для работы в реальном времени таких классических алгоритмов выявления аномалий, как KNN, Isolation-forest, DBScan [13].

VII. ЗАКЛЮЧЕНИЕ

В работе исследовались подходы к решению задачи выявления (детекции) аномалий при обработке больших объемов потоковых данных в распределенных системах в режиме реального времени. При этом предполагалось, что в реальных процессах возможны резкие изменения значений отдельных показателей временных рядов, вызванные разными причинами.

Для решения задачи выявления аномалий авторами предложена модификация алгоритма K-Means, названная Real-Time K-Means, позволяющая работать с потоковыми данными, минуя «холодный старт».

В работе проведен сравнительный анализ эффективности выявления аномалий разработанной
авторами модификации алгоритма Streaming K-Means и реализация Streaming K-Means из библиотеки MLlib фреймворка Apache Spark, предобученной на тренировочном датасете. Сравнение подтвердило эффективность предложенной модификации.

Для проведения испытаний выбранных алгоритмов был подготовлен датасет, состоящий из метрик JVM и Kafka при передаче сообщений двумя производителями. При создании датасета изменялись количество сообщений в секунду и размер передаваемых сообщений. В этот датасет были добавлены аномальные фрагменты, с большим числом сообщений в секунду и/или размером сообщения. Значения датасета были предобработаны для выравнивания влияния метрик и исключения корреляций.

При учете всех некоррелирующих метрик центры кластеров стандартной модели K-Means и модели Streaming K-Means из библиотеки MLlib фреймворка Apache Spark оказываются разнесены, однако при учете только метрик, напрямую связанных с аномальностью, центры с высокой точностью совпадают. Однако, из-за случайного выбора начальных точек центров кластеров, точность Streaming K-Means может оказаться ниже, так как число итераций Streaming K-Means ограничено необходимостью обрабатывать потоковые данные с большой скоростью.

Подводя итог, можно сказать, что разработанная авторами модификация модели K-Means для работы в реальном времени показала хорошие результаты и может являться основой для модификации других классических алгоритмов выявления аномалий для работы в реальном времени с потоковыми данными.

БЛАГОДАРНОСТИ

Авторы выражают благодарность Высшей инжиниринговой школе НИЯУ МИФИ за помощь в возможности опубликовать результаты выполненной работы.

БИБЛИОГРАФИЯ

[1] Sarvani A., Venugopal B., Devarakonda N. (2019) Anomaly Detection Using K-means Approach and Outliers Detection Technique. In: Ray K., Sharma T., Rawat S., Saini R., Bandyopadhyay A. (eds) Soft Computing: Theories and Applications. Advances in Intelligent Systems and Computing, vol 742. Springer, Singapore.

[2] Lemaire, V., Alaoui Ismaili, O., Cornu´ejols, A., Gay, D.: Predictive k-means with localmodels. In: Workshop LDRC-2020 (Workshop on Learning Data Representation for Clustering) in PAKDD-2020 (The 24th Pacific-Asia Conf. On Knowledge Discovery and DataMining), Singapore, 11-16 May 2020.

[3] Tsigkritis, T. , Groumas, G. and Schneider, M. (2018) On the Use of k-NN in Anomaly Detection. Journal of Information Security, 9, 70-84.

[4] Unified engine for large-scale data analytics [электронный ресурс] https://spark.apache.org/ Дата обращения 01.10.2021

[5] Apache Hadoop https://hadoop.apache.org/ [электронный ресурс] Дата обращения 01.10.2021

[6] Wang, Z.; Zhou, Y.H.; Li, G.M. Anomaly Detection by Using Streaming K-Means and Batch K-Means. 2020 5th Ieee International Conference on Big Data Analytics (IEEE ICBDA 2020), Xiamen, China, 8–11 May 2020; pp. 11–17

[7] Clustering - RDD-based API https://spark.apache.org/docs/latest/mllib-clustering.html [электронный ресурс] Дата обращения 01.10.2021

[8] Fawcett T. An introduction to ROC analysis. Pattern Recogn Lett. 2006; 27(8): 861–74.

[9] Hyndman, R.J., & Athanasopoulos, G. (2021) Forecasting: principles and practice, 3rd edition, OTexts: Melbourne, Australia..

[10] Vannel Zeufacka, Donghyun Kimb, Daehee Seoc, Ahyoung Leea An unsupervised anomaly detection frame-work for detecting anomalies in real time through network system's log files analysis, High-Confidence Computing Volume 1, Issue 2, December 2021, 100030.

[11] Authors: D. Benmahdi, L. Rasolofondraibe, X. Chiementin, S. Murer, A. Felkaoui, RT-OPTICS: real-time classification based on OPTICS method to monitor bearings faults, Journal of Intelligent Manufacturing, Volume 30, Issue 5, June 2019, pp. 2157–2170.

[12] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. 2020. Deep Learning for Anomaly Detection: A Review. ACM Comput. Surv. 1, 1, Article 1 (January 2020), 36 pages.

[13] Md Tahmid Rahman Laskar, Jimmy Xiangji Huang, Vladan Smetana, Chris Stewart, Kees Pouw, Aijun An, Stephen Chan, and Lei Liu. 2021. Extending Isolation Forest for Anomaly Detection in Big Data via K-Means. ACM Trans. Cyber-Phys. Syst. 5, 4, Article 41 (September 2021), 26 pages.

Статья получена 23 марта 2022.

Д.Е. Савицкий, Национальный Исследовательский Ядерный Университет МИФИ, магистрант, (e-mail: Denis_savickii96@mail.ru)

М.Е. Дунаев, Национальный Исследовательский Ядерный Университет МИФИ, аспирант, (e-mail: Max.dunaev@mail.ru)

К.С. Зайцев, Национальный Исследовательский Ядерный Университет МИФИ, профессор, (e-mail: KSZajtsev@mephi.ru)

Anomaly detection in real-time streaming data processing

D.E. Savitsky, M.E. Dunaev, K.S. Zaytsev

Annotation **— The purpose of this work is to study methods for detecting anomalies in the processing of data streams in distributed streams in real time. To do this, the authors carried out a modification of the K-Means algorithm, called K-Means in real time, and carried out a comparative analysis of the effectiveness of the developed algorithm with K-Means from the MLlib library of the Apache Spark framework. The comparison confirmed the effectiveness of the proposed modification. To conduct experiments with the algorithm, a special data array (dataset) was built, which included about 1000 measurements of the Apache Kafka server log metrics with one topic, two providers and a consumer. Anomalous fragments have been added to this set of dates, with a large number of messages in the blink of an eye and/or size. The dataset values have been pre-processed to align the index of metrics and exclude correlations. Results developed by the authors of the K-Means algorithm for solving anomaly search problems, taking into account the detection time of its effectiveness.**

Keywords **— anomaly detection, real-time mode, distributed processing, event log metrics, machine learning, clustering, statistical methods, Apache Spark**

REFERENCES

[1] Sarvani A., Venugopal B., Devarakonda N. (2019) Anomaly Detection Using K-means Approach and Outliers Detection Technique. In: Ray K., Sharma T., Rawat S., Saini R., Bandyopadhyay A. (eds) Soft Computing: Theories and Applications. Advances in Intelligent Systems and Computing, vol 742. Springer, Singapore.

[2] Lemaire, V., Alaoui Ismaili, O., Cornu´ejols, A., Gay, D.: Predictive k-means with localmodels. In: Workshop LDRC-2020 (Workshop on Learning Data Representation for Clustering) in PAKDD-2020 (The 24th Pacific-Asia Conf. On Knowledge Discovery and DataMining), Singapore, 11-16 May 2020.

[3] Tsigkritis, T., Groumas, G. and Schneider, M. (2018) On the Use of k-NN in Anomaly Detection. Journal of Information Security, 9, 70-84. doi: 10.4236/jis.2018.91006.

[4] Unified engine for large-scale data analytics https://spark.apache.org/ Reviewed 01.10.2021

[5] Apache Hadoop https://hadoop.apache.org/ Reviewed 01.10.2021

[6] Wang, Z.; Zhou, Y.H.; Li, G.M. Anomaly Detection by Using Streaming K-Means and Batch K-Means. 2020 5th Ieee International Conference on Big Data Analytics (IEEE ICBDA 2020), Xiamen, China, 8–11 May 2020; pp. 11–17

[7] Clustering - RDD-based API https://spark.apache.org/docs/latest/mllib-clustering.html Reviewed 01.10.2021

[8] Fawcett T. An introduction to ROC analysis. Pattern Recogn Lett. 2006; 27(8): 861–74

[9] Hyndman, R.J., & Athanasopoulos, G. (2021) Forecasting: principles and practice, 3rd edition, OTexts: Melbourne, Australia.

[10] Vannel Zeufacka, Donghyun Kimb, Daehee Seoc, Ahyoung Leea An unsupervised anomaly detection frame-work for detecting anomalies in real time through network system's log files analysis, High-Confidence Computing Volume 1, Issue 2, December 2021, 100030

[11] Authors: D. Benmahdi, L. Rasolofondraibe, X. Chiementin, S. Murer, A. Felkaoui, RT-OPTICS: real-time classification based on OPTICS method to monitor bearings faults, Journal of Intelligent Manufacturing, Volume 30, Issue 5, June 2019, pp. 2157–2170

[12] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. 2020. Deep Learning for Anomaly Detection: A Review. ACM Comput. Surv. 1, 1, Article 1 (January 2020), 36 pages. https://doi.org/10.1145/3439950

[13] Md Tahmid Rahman Laskar, Jimmy Xiangji Huang, Vladan Smetana, Chris Stewart, Kees Pouw, Aijun An, Stephen Chan, and Lei Liu. 2021. Extending Isolation Forest for Anomaly Detection in Big Data via K-Means. ACM Trans. Cyber-Phys. Syst. 5, 4, Article 41 (September 2021), 26 pages, DOI: https://doi.org/10.1145/3460976.