УДК:004.85

КЛАСТЕРИЗАЦИЯ ОДНОМЕРНЫХ ПОТОКОВЫХ ДАННЫХ НА ОСНОВЕ ПЛОТНОСТИ РАСПРЕДЕЛЕНИЯ ДАННЫХ В ПРОСТРАНСТВЕ ПРИЗНАКОВ

Митин Г.В., Панов А.В.

Федеральное государственное бюджетное образовательное учреждение высшего образования «МИРЭА – Российский технологический университет», 119454, Российская Федерация, г. Москва, пр-т Вернадского, 78, e-mail: grigory.mitin@mail.ru, insegnantenew@yandex.ru

В статье рассматривается оригинальный подход к кластеризации одномерных потоковых данных, опирающийся на принципы кластеризации на основании плотности распределения данных в пространстве признаков. Это позволяет работать в условиях информационного шума с целью отсечения выбросов и неинформативных аномальных данных. Для реализации данного подхода был разработан алгоритм, состоящий из нескольких функциональных блоков и предполагающий поиск одномерных границ кластеров, который эффективно использует информацию о появлении новых кластеров, сохраняя только значимые элементы данных, что положительно сказывается на требованиях к вычислительным ресурсам. Для дальнейшего повышения эффективности предложенного алгоритма применен подход адаптивного разбиения данных из входного потока на фреймы различного размера с последующей обработкой на основании эвристического подхода, учитывающего особенности одномерного пространства признаков и накопительный характер информации о наличии кластеров. Результирующий алгоритм демонстрирует высокую эффективность по скорости обработки данных и используемой памяти. Его вычислительная сложность стремится к линейной с течением времени, то есть к O(n). Также авторам удалось достичь высоких показателей качества кластеризации, оцениваемых по критериям компактности и разделимости кластеров, являющихся универсальными для любых алгоритмов кластеризации на основе плотности распределения данных в пространстве признаков. Перечисленные преимущества подтверждены при помощи эксперимента над 20 наборами тестовых данных, результаты которого также приведены в рамках данной работы. Представленный алгоритм является одним из немногих алгоритмов кластеризации потоковых данных способных работать в условиях информационного шума, и при этом, оптимизированных для работы с одномерными данными. По отдельности каждая из задач кластеризации потоковых данных и кластеризации одномерных данных рассматриваются научным сообществом довольно давно, однако, их совокупность остается без должного внимания, несмотря на очевидную пользу, например, для решения задач поиска устойчивых состояний или очистки от аномальных и шумовых значений при анализе одномерных сигналов, показаний датчиков и т.п.

Ключевые слова: машинное обучение, обучение без учителя, кластеризация, информационные технологии, плотность распределения данных, одномерная кластеризация, адаптивные фреймы данных, потоковые данные, отсечение информационного шума.

DENSITY BASED SINGLE DIMENSION STREAM DATA CLUSTERING

Mitin G.V., Panov A.V.

Federal State Budget Educational Institution of Higher Education «MIREA – Russian Technological University», 119454, Russian Federation, Moscow, Vernadsky pr., 78, e-mail:grigory.mitin@mail.ru, insegnantenew@yandex.ru

The article considers an original approach to clustering of single-dimensional streaming data, based on the principles of density based clustering. This allows to work in conditions of information noise in order to cut off outliers and uninformative anomalous data. To implement this approach, an algorithm was developed consisting of several functional blocks and involving the search for single-dimensional cluster boundaries using machine learning technologies, which effectively uses information about the appearance of new clusters, preserving only significant data elements, which has a positive effect on the requirements for computing resources.

To further improve the efficiency of the proposed algorithm, an approach of adaptive splitting of data from the input stream into frames of various sizes with subsequent processing based on a heuristic approach that takes into account the features of single-dimensional feature space and the cumulative nature of information about the presence of clusters is applied. The resulting algorithm demonstrates high efficiency in terms of data processing speed and memory usage. Its computational complexity tends to be linear over time. The authors also managed to achieve high clustering quality indicators, evaluated according to the criteria of compactness and separability of clusters, which are universal for any clustering algorithms based on the density of data distribution in the feature space. These advantages were confirmed by an experiment on 20 sets of test data, the results of which are also presented in the framework of this work. The presented algorithm occupies a rare niche of algorithms for clustering streaming data in conditions of information noise, optimized for working with one-dimensional data. Individually, each of the tasks of clustering streaming data and clustering one-dimensional data has been considered by the scientific community for quite a long time, however, their totality remains without due attention, despite the obvious benefits, for example, for solving problems of searching for stable states or clearing anomalous and noise values when analyzing one-dimensional signals, sensor readings, etc.

Keywords: machine learning, unsupervised learning, clustering, streaming data, information technologies, density based, single dimension clustering, adaptive data frame, dealing with noise.

Введение

На протяжении последних двух десятилетий наблюдается нерперывный рост объемов данных, с которыми работают информационные системы. В связи с этим растут и потребности в современных средствах интеллектуального анализа данных, способных обработать эти объемы и позволить человеку эффективно воспользоваться ими. Подобные аналитические системы выполняют обработку потоков данных с большим числом признаков, что требует исследования пространства признаков высокой размерности и сопряжено с рядом серьезных проблем [1].

Одной из таких проблем является так называемое «проклятие размерности», заключающееся в том, что при увеличении размерности данных происходит экспоненциальный рост «степеней свободы», которые необходимо отслеживать при обработке. Один из распространенных способов преодоления этой проблемы заключается в преобразовании многомерных входных данных в наборы одномерных и двумерных [2].

Преобразование к двумерным наборам применяют в тех случаях, когда искомый анализируемый параметр из многомерного набора данных не имеет смысла или, трудно извлекаем без явной связи с каким-либо другим параметром. Классическим примером необходимости двумерных наборов данных является анализ какой-либо величины, имеющей жесткую привязку ко времени, а не к порядковому номеру в выборке. Тогда время выступает не просто как один из параметров для сортировки данных, но и как значимый признак, выступающий в роли координатной оси x, как при визуализации, так и при других видах анализа данных.

Преобразование к одномерным наборам применяют в тех случаях, когда искомый анализируемый параметр из многомерного набора данных имеет собственный смысл в отрыве от остальных элементов данных этого набора и может быть обработан отдельно, либо с последующей синхронизацией результатов обработки по какому-либо косвенному признаку.

В описанном выше контексте, одним из востребованных способов анализа данных для современных прикладных задач является кластеризации одномерных потоковых данных. Несмотря на то, что многие алгоритмы кластеризации, предназначенные для обработки потоковых данных в пространстве признаков высокой размерности, могут быть использованы и для обработки одномерных потоковых данных, такие алгоритмы не в полной мере используют особенности одномерных данных, что снижает их эффективность.

Для решения задачи кластеризации одномерных данных существует несколько хорошо зарекомендовавших себя подходов. В частности, существует вариант алгоритма кластеризации k-means, оптимизированный для одномерных данных [3]. Авторы одномерной вариации k-means добились оптимальности разбиения одномерных данных на кластеры, сохраняя квадраты внутрикластерных отклонений в двумерный массив (n+1)*(k+1) элементов, где k – число кластеров, а n – число точек данных. Однако, высокая вычислительная сложность этого подхода делает его непригодным для работы с потоковыми данными. К тому же, использование алгоритма k-means не позволяет работать в условиях информационного шума.

Существуют также варианты одномерных алгоритмов кластеризации на основании плотности распределения данных. Одним из примеров является алгоритм DBSCAN1D [4]. Этот алгоритм позволяет достичь линейной

вычислительной сложности на фиксированной выборке данных, сортированной по возрастанию. DBSCAN1D является двухпроходным алгоритмом: на первом проходе он определяет для каждого элемента выборки максимальный и минимальный элементы, достижимые по плотности из этой точки, на втором проходе алгоритм избавляется от шумовых элементов. Результирующий алгоритм имеет значительно более низкую сложность по сравнению с классическим n-мерным DBSCAN, однако требует в три раза больше памяти. А тот факт, что DBSCAN1D работает только с упорядоченной выборкой, делает алгоритм непригодным для работы с потоковыми данными даже при разбиении потока на фреймы.

Вариант EM-алгоритма для потоковых данных [5], который опирается на предположение, что данные подчиняются нормальному распределению, для каждого кластера вычисляет функцию плотности нормального распределения в пространстве признаков. Принятие решения о принадлежности новых точек тому или иному кластеру производится им на основании вероятности вхождения точки с заданными координатами в тот или иной кластер. Однако такой подход не позволяет отделять кластеры от информационного шума и аномальных данных.

Проводятся практические исследования динамики качества кластеризации реальных данных при помощи нейросетевых методов, например, на основе сетей Кохонена с использованием k-means и генетического алгоритма для предобработки данных и перенастройки нейросети [6]. Описанный подход показывает высокую точность на больших объемах данных, однако не оптимизирован для одномерных данных и требует значительных вычислительных ресурсов.

Актуальность проблемы

Задача кластеризации потоковых данных для одномерного сигнала в условиях информационного шума является весьма актуальной в наше время. Она возникает при решении многих прикладных задач, таких как анализ изменения частоты возникновения конкретных событий или поиск устойчивых состояний, когда по тем или иным причинам математические и другие классические методы оказываются бессильны. Например, когда часть входных данных оказывается неинформативной и должна быть отфильтрована по какому-либо сложному признаку, который нельзя задать заранее на этапе проектирования системы обработки данных. В этом случае на помощь приходят методы кластеризации с применением технологий машинного обучения [2].

Однако, большинство из существующих алгоритмов машинного обучения предназначено для обработки фиксированного набора данных и требуют, чтобы число точек данных было известно заранее. В то же время оценить число точек в потоковых данных невозможно, так как оно стремится к бесконечности с течением времени. В этих условиях, логическое разделение потоковых данных на фреймы с контролируемым числом элементов позволяет решить эту проблему, сводя задачу к обработке последовательности конечных выборок.

Многие методы работы с потоковыми данными не предусматривают сохранения уже обработанных точек, так как суммарный объем потоковых данных не ограничен [7]. Тем не менее, информация о характерных особенностях распределения данных может быть использована для оптимизации обработки последующих фреймов с целью снижения вычислительной сложность дальнейшего анализа [8].

Стоит заметить, что работа в условиях информационного шума налагает определенные требования на принципы, по которым осуществляется кластеризация. Например, алгоритмы, предполагающие разделение всего пространства признаков на фиксированное число областей-кластеров, являются непригодными для работы в условиях информационного шума, так как не позволяет отделять выбросы и неинформативные аномальные данные от данных, образующих информативные кластеры.

Наиболее интересным является принцип кластеризации на основании плотности распределения данных, так как алгоритмы, опирающиеся на этот метод, показывают хорошие результаты в задачах отделения информационного шума и аномальных данных.

Постановка задачи

Резюмируя все вышесказанное, можно сформулировать несколько важных этапов, которые нужно пройти для эффективного решения задачи кластеризации одномерных потоковых данных:

Организовать работу с потоковыми данными, разделяя поток данных на фреймы, размер которых должен подбираться адаптивно к соотношению полезного информационного сигнала и информационного шума в потоке в каждый момент времени.

Данные из уже обработанных фреймов необходимо не удалять, а сохранять, что должно увеличить эффективность по вычислительной сложности и расходу памяти при обработке последующих фреймов. Однако,

необходимо создать эффективный механизм, предотвращающий «затопление» памяти старыми данными теряющими актуальность с течением времени.

Решить задачу кластеризации при помощи кластеризации на основе плотности распределения данных, как хорошо зарекомендовавшим себя подходом, эффективно справляющимся с аномалиями, выбросами и информационным шумом.

Связать все функциональные блоки воедино, таким способом, который позволил бы им взаимодействовать в полностью автоматическом режиме, не требуя вмешательства оператора для настройки каких-либо параметров при значительных изменениях в составе поступающих данных.

Предлагаемое решение

Для решения поставленных задач был разработан алгоритм кластеризации потоковых данных на основании распределения плотности, оптимизированный для работы с одномерными данными. Здесь и далее он обозначен как DBSDSC (Density Based Single Dimension Stream Clustering).

В одномерном пространстве мало степеней свободы и любая фигура, образованная множеством точек, является отрезком некоторой длины, который может быть описан максимальной и минимальной точками. По аналогии с отрезком, для описания одномерного кластера необходимо сохранить всего два элемента кластера, вне зависимости от того, сколько точек данных находится между ними, что значительно снижает расход памяти. Такая граница сохраняется в структуру, обозначенную далее, как Список Элементов Границы 1D. Эта структура представляет собой два одномерных массива, один из которых хранит минимальные элементы кластеров, а второй – максимальные элементы кластеров.

Кластеризация на основании плотности распределения данных имеет ряд специфических особенностей. Базовыми понятиями являются понятие соседства между точками данных и понятие окрестности. Соседями некоторой выделенной точки данных являются точки данных, такие что модуль разности координат между каждой такой точкой и выделенной точкой не превышает некоторого заданного значения ***EPS***, называемое радиусом окрестности. Область пространства признаков, ограниченное расстоянием ***EPS*** от выделенной точки, называют окрестностью этой точки. В зависимости от числа и типа соседних точек, точки данных можно разделить на несколько типов, по аналогии с обозначениями классического алгоритма DBSCAN [9]:

1. Корневые точки кластера – точки данных, образующие достаточно плотные скопления, чтобы считать их элементами одного кластера. Точка считается корневой, если у нее не менее ***MinPts*** соседей;
2. Граничные точки кластера – точки данных, находящиеся близко от плотных скоплений точек данных, образующих кластер, но не входящие в такие скопления и выполняющие вспомогательную роль при обработке нового фрейма данных. Точка считается граничной, если среди ее соседей есть корневая точка, но число соседей меньше ***MinPts***;
3. Шумовые точки – точки данных, находящиеся далеко от плотных скоплений точек данных и не образующие плотных скоплений самостоятельно. Точка считается шумовой, если у нее менее MinPts соседей, и среди соседей нет ни одной корневой точки.

Пусть *Список Элементов Границы 1D* хранит только максимальные и минимальные точки каждого кластера, относящиеся к корневым точкам. Для того чтобы оценить плотность распределения данных в окрестностях этих точек, введем еще одну структуру, обозначенную далее как *Список Вспомогательных Элементов 1D*, в которой хранятся точки данных, оказавшиеся в радиусе окрестности максимального или минимального элемента того или иного кластера, как граничные, так и корневые, при этом каждой точке ставится в соответствие счетчик соседей – целое число, отражающее количество элементов *Списка Вспомогательных Элементов 1D*, находящихся в ее радиусе окрестности. Стоит уточнить, что *Список Вспомогательных Элементов 1D* имеет две части: в одной хранится список точек, находящихся в радиусе соседства «максимальной» границы кластера, а в другой части – список точек в радиусе соседства «минимальной» границы кластера.

Входящие потоковые данные разделяются на фреймы адаптивного размера, с которыми и работает предложенный алгоритм. Общий принцип работы алгоритма представлен на рис.1, где жирным контуром обведены процедуры, рассмотренные далее в тексте более подробно. В общих чертах его работа заключается в следующем:

1. При получении нового фрейма, проводится проверка на вхождение точки данных в один из ранее обнаруженных кластеров.

2. Точки данных, не являющиеся элементами одного из ранее обнаруженных кластеров, проходят промежуточную кластеризацию для выявления новых кластеров.

Рисунок 1. Общий вид алгоритма кластеризации

3. Границы кластеров, обнаруженных в ходе промежуточной кластеризации, обновляются. В *Список Элементов Границы 1D* добавляются максимальные и минимальные элементы каждого кластера.

Перейдем к детальному рассмотрению перечисленных этапов кластеризации. Процедура разметки точек фрейма на основании вхождения в существующие кластеры представлена на рис.2.

Как уже было сказано, одной из ключевых особенностей предложенного алгоритма является сохранение и обновление границ кластеров. Для многих алгоритмов, работающих в многомерном пространстве признаков, проверка вхождения точки данных в границу кластера представляла бы процедуру, включающую в себя ряд геометрических преобразований, сложность которых будет экспоненциально расти в зависимости от размерности пространства признаков. Однако при обработке одномерных данных оказывается достаточно проверки вхождения точки в диапазон значений, что является вычислительно простой операцией, не требующей отдельного разветвленного алгоритма, которую можно также совместить с обновлением границ уже существующих кластеров.

В общем виде обработка фрейма может быть сведена к следующему алгоритму:

1. Каждая входящая точка данных проверяется на попадание в радиус окрестности элементов *Списка Элементов Границы 1D* и, в случае попадания, сохраняется в *Список Вспомогательных Элементов 1D*, при этом определяется число соседей новой точки данных и обновляются счетчики соседей всех вспомогательных элементов, находящихся в радиусе соседства этой точки.

2. *Список Вспомогательных Элементов 1D* проверяется на появление точек, чей счетчик соседей равен или превышает ***MinPts***, и если такие элементы найдутся, то помечаются как корневые элементы.

3. Все корневые точки в *Списке Вспомогательных Элементов 1D,* проходят проверку на нахождение вне границ, описанных при помощи *Списка Элементов Границы 1D*. Корневые точки, чье значение окажется больше максимального элемента кластера или меньше минимального элемента, заменяют максимальный или минимальный элемент кластера соответственно.

4. При изменении границы кластера, удаляются все вспомогательные элементы, отстоящие дальше чем ***EPS*** от любой границы кластера.

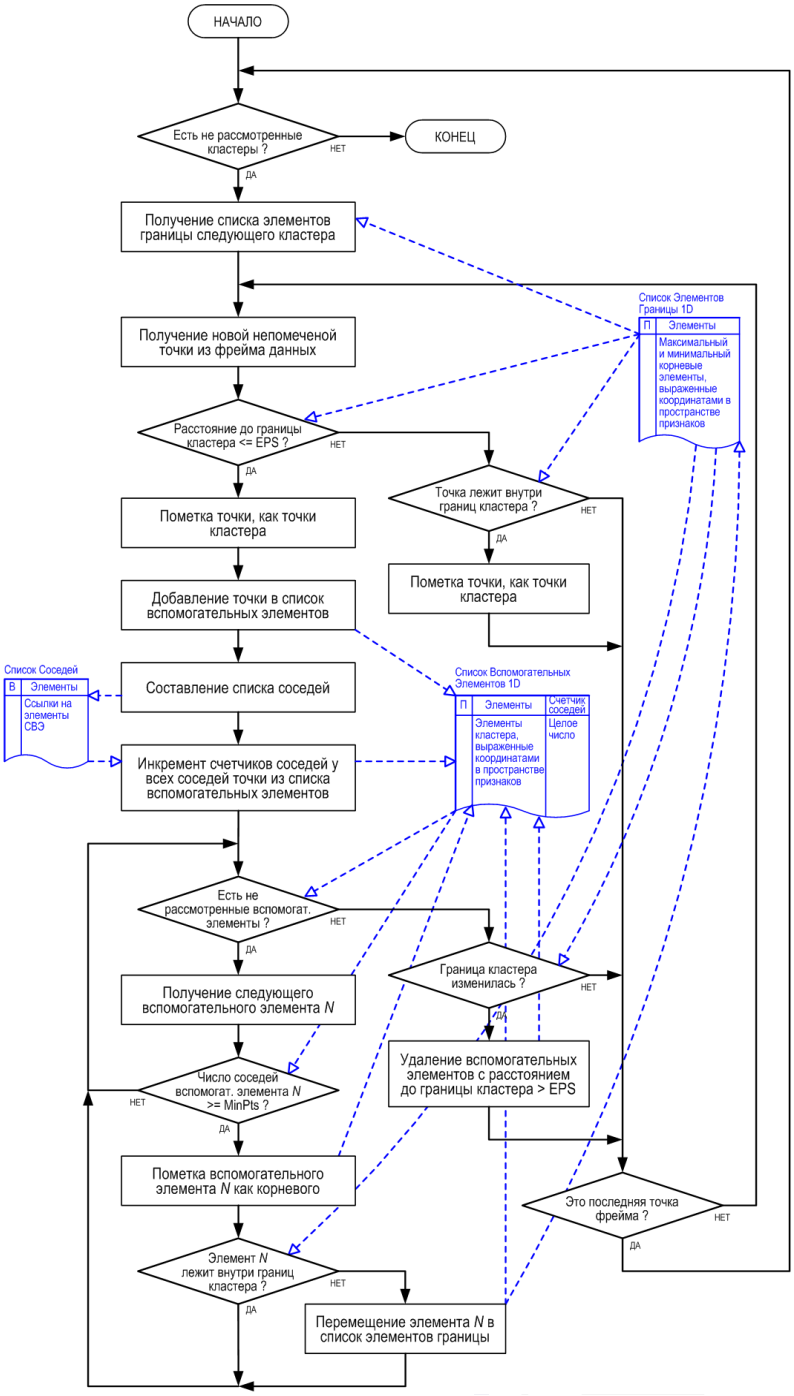

Рисунок 2. Разметка точек фрейма на основании вхождения в существующие кластеры

Таким образом, число точек, которые необходимо держать в памяти для каждого кластера будет ограничено окрестностями максимальной и минимальной точек. Для большего контроля расхода памяти, можно включить дополнительное условие – если число точек в *Списке Вспомогательных Элементов 1D* превышает ***MinPts*2***, следует удалить из *Списка Вспомогательных Элементов 1D* корневые точки, отстоящие дальше всех от границы кластера, чтобы общее число точек этого списка стало равно ***MinPts*2***. Это приведет к снижению как

затрат по необходимому для работы алгоритма объему оперативной памяти, так и к снижению вычислительной сложности предобработки точек данных из входного потока. Хотя данный показатель был получен экспериментальным путем и является эвристическим, его эффективность проверена на нескольких десятках наборов тестовых данных, о чем будет сказано ниже в разделе «Экспериментальная оценка качества и скорости кластеризации».

Точки фрейма, не получившие метки кластера, проходят процедуру промежуточной кластеризации, чтобы выявить новые кластеры. Детальный алгоритм представлен на рис.3.

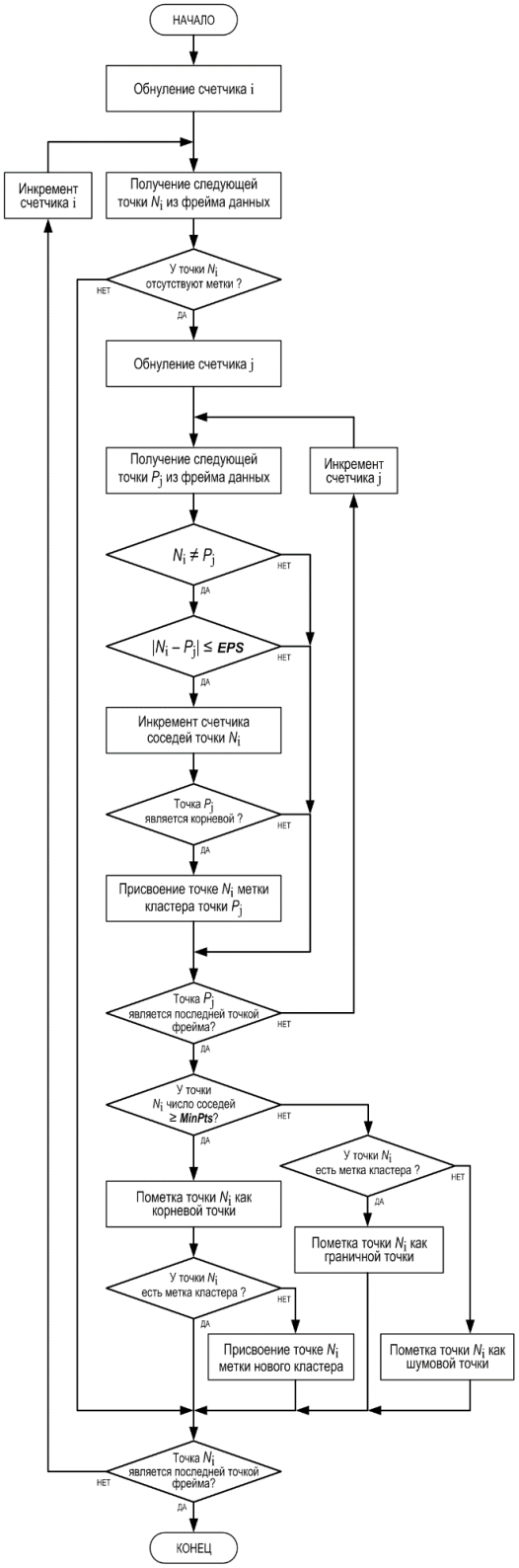

Рисунок 3. Промежуточная кластеризация

Процедура промежуточной кластеризации опирается на те же принципы, что и классический алгоритм кластеризации на основании плотности распределения данных DBSCAN. В рамках этой процедуры, для каждой точки данных, не имеющей метки кластера, проводится поиск соседей среди точек фрейма. В зависимости от числа и типа соседей, точка помечается как корневая, граничная или шумовая. Корневые точки участвуют в образовании новых кластеров, которые не были определены на предыдущих этапах обработки: при нахождении

корневой точки, не являющейся частью какого-либо кластера, ей присваивается метка нового кластера – например, порядковый номер, следующий за номером последнего обнаруженного кластера.

Процедура промежуточной кластеризации предполагает исследование всех пар непомеченных точек данных, то есть имеет квадратичную сложность. В то же время, по мере заполнения пространства признаков известными кластерами, число непомеченных точек данных будет снижаться. Таким образом, с течением времени вычислительная сложность промежуточной кластеризации будет стремиться к нулю, а совокупная вычислительная сложность всего алгоритма – к линейной сложности.

Данные о каждом новом кластере, выявленном в ходе промежуточной кластеризации, следует включить в *Список Элементов Границы 1D* и *Список Вспомогательных элементов 1D*, чтобы обеспечить их последующую обработку при помощи процедуры разметки точек фрейма. Эти действия производятся в рамках процедуры определения границы кластера, детальный алгоритм которой изображен на рис.4.

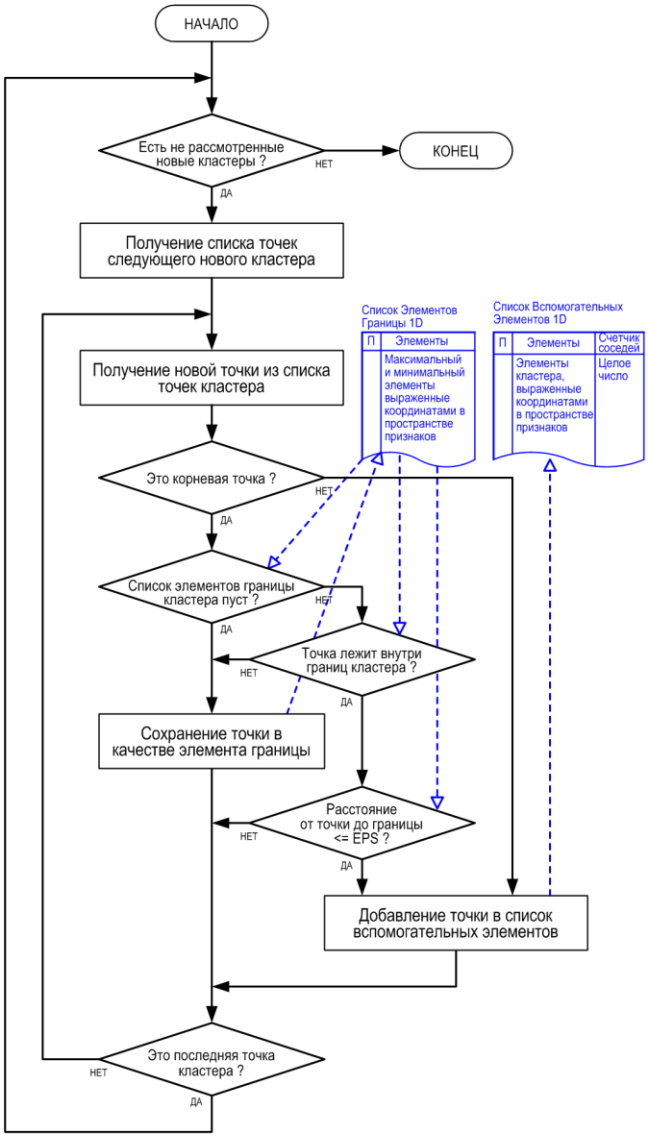

Рисунок 4. Определение границ кластеров

Общий принцип определения границы кластера прост. Вначале происходит перебор элементов каждого кластера, обнаруженного на этапе промежуточной кластеризации, в поисках максимального и минимального корневых элементов, с последующим с последующим сохранением в *Список Элементов Границы 1D*, и соседних с ними элементов того же кластера для сохранения в *Список Вспомогательных Элементов 1D*. Точки

кластера, не вошедшие в *Список Элементов Границы 1D* и *Список Вспомогательных Элементов 1D* не сохраняются.

После того, как все точки данных во фрейме прошли обработку, следует произвести коррекцию размера следующего фрейма данных для поддержания динамического баланса между качеством и скоростью обработки потоковых данных. Логика такого преобразования зависит от прикладной задачи. По умолчанию предлагается использовать алгоритм, изображенный на рис.5:

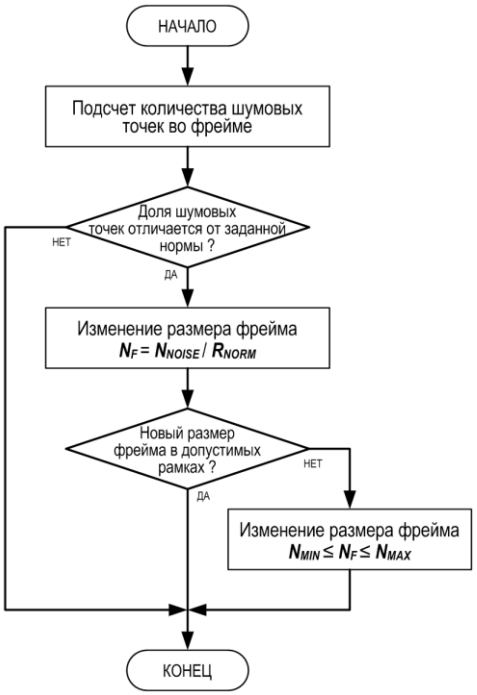

Рисунок 5. Корректировка размера фрейма

Первым шагом является подсчет параметра N_{NOISE} – количества шумовых точек в последнем обработанном фрейме данных, чтобы измерить общую степень информативности данных и вычислить отношение количества шумовых точек к общему числу точек фрейма N_F

$$R_F = N_{NOISE} / N_F \quad (1)$$

Полученное значение сравнивается с R_{NORM} – заранее заданным параметром, описывающим допустимую долю шумовых точек во фрейме.

Если доля шумовых точек соответствует заданной норме, размер фрейма не изменяется, в противном случае, размер фрейма изменяется, в соответствии с

$$N_F = N_{NOISE} / R_{NORM} \quad (2)$$

Заключительным этапом корректировки размера фрейма является проверка, находится ли новый размер фрейма в рамках допустимых значений, определенных заранее заданными параметрами минимального и максимального размера фрейма (N_{MIN}, N_{MAX}).

Если новый размер фрейма оказался меньше или больше допустимого, он заменяется на соответсвующее допустимое значение.

Логика этого алгоритма опирается на предположение, что увеличение размера фрейма повышает вероятность получения информативных данных. Таким образом, если доля информационного шума во фрейме оказывается слишком большой, необходимо увеличить размер фрейма, чтобы получить больше информативных данных, а если доля информационного шума оказывается низкой, следует уменьшить размер фрейма, тем самым ускорив обработку.

Стоит отметить, что по мере заполнения пространства признаков обнаруженными кластерами, вычислительная сложность предложенного алгоритма кластеризации одномерных потоковых данных должна

стремиться к линейной сложности, таким образом выигрыш по быстродействию, вызванный изменением размера фрейма, будет уменьшаться с течением времени.

Расчет вычислительной сложности

Далее приведены ***O***-оценки временной сложности каждой процедуры представленного алгоритма.

Вычислительная сложность *Определения границы кластера* составляет

$$O(cpn), \quad (3)$$

где ***cpn*** – число элементов нового кластера.

Вычислительная сложность *Разметки точек одного фрейма на основании вхождения в существующие кластеры* составляет

$$O(fp*se) = O(fp), \quad (4)$$

где ***fp*** – число точек данных во фрейме, ***se*** – число элементов *Списка Вспомогательных Элементов 1D*.

Вычислительная сложность *Промежуточной кластеризации* составляет

$$O((fp-fpo)2), \quad (5)$$

С учетом *Промежуточной кластеризации* точек данных, не определенных как элементы одного из существующих кластеров, и с учетом (1) и (2), получаем итоговую вычислительную сложность:

$$O(fp + (fp-fpo)^2 + cpn) = O((fp-fpo)^2), \quad (6)$$

где ***fp*** – число точек данных во фрейме, ***fpo*** – число элементов фрейма, принадлежащих к кластерам, определенным в предыдущих фреймах, ***cpn*** – число элементов фрейма, образующих новые кластеры, появившиеся после поступления фрейма на обработку. По мере увеличения числа элементов, вошедших в один из ранее существовавших кластеров (число ***fpo***), вычислительная сложность обработки каждого нового фрейма будет снижаться, стремясь к линейной. Таким образом, в случае полного заполнения пространства признаков кластерами (***fp = fpo, cpn = 0)*** , вычислительная сложность составит:

$$O(fp) \quad (7)$$

Экспериментальная оценка качества и скорости кластеризации

Чтобы оценить качество и быстродействие представленного алгоритма, был проведен эксперимент, в ходе которого программная реализация предложенного алгоритма получала на вход данные из 20-и различных тестовых наборов данных, после чего производились замеры времени обработки этих наборов и показателей качества кластеризации. Эксперимент проводился в 64-битной среде Windows 10 на домашнем ПК (CPU: Intel Core i7-7700 3.6GHz, RAM: 32GB, HDD: SSD512GB).

Для того чтобы лучше оценить предложенный алгоритм, названный DBSDSC, было проведено сравнение с аналогами. Одним из них явился более примитивный вариант алгоритма кластеризации одномерных потоковых данных, являющийся частью гибридной модификации DBSCAN [10]. В нем качество кластеризации было снижено в угоду быстродействию и экономии памяти – например, отсутствует *Список Вспомогательных Элементов 1D*. Одномерный алгоритм из гибридной модификации DBSCAN [10] был введен, чтобы показать разницу между алгоритмом, нацеленным на скорость обработки фрейма, и представленным новым алгоритмом DBSDSC, нацеленным на качество кластеризации. На графиках одномерный вариант гибридной модификации DBSCAN обозначается «*1d*».

Другим алгоритмом для сравнения стал классический DBSCAN, настроенный на обработку фреймов с сохранением в памяти всех точек данных, попавших в кластеры. Классический DBSCAN был введен, чтобы показать разницу между одним из распространенных алгоритмом, и представленным новым алгоритмом DBSDSC. На графиках классический DBSCAN обозначается «DBSCAN».

Для тестирования показателей перечисленных алгоритмов были взяты наборы данных из открытых источников:

Набор данных «Electrical Fault detection and classification» [11]. Набор содержит замеры силы тока трехфазной сети в нормальном режиме работы и короткого замыкания, которые можно считать информационным шумом (короткое замыкание) и кластерами (нормальный режим работы). Данный многомерный набор данных был разделен на одномерные составляющие, что позволило получить несколько не связанных одномерных наборов.

Набор данных «Individual Household Power Consumption» [12]. В нем содержится статистика расхода электроэнергии одного частного домохозяйства в течении нескольких лет. Потребление энергии в тои или ином помещении зависит от времени, что создает несколько пересекающихся кластеров. Данный многомерный набор

данных был также разделен на одномерные составляющие, что позволило получить несколько не связанных одномерных наборов.

Набор данных «Online shoppers intention» [13]. В нем содержится статистика посещения пользователями некоторого сайта страниц с разным содержанием. И этот многомерный набор данных был разделен на одномерные составляющие, что позволило получить несколько не связанных одномерных наборов.

Набор данных «Postures» [14]. В нем содержатся данные о положении в пространстве ключевых точек на специальных перчатках для отслеживания жестов руками. Как и предыдущие, данный многомерный набор данных был разделен на одномерные составляющие, что позволило получить несколько не связанных одномерных наборов.

В результате основные показатели предложенного алгоритма были измерены на эквиваленте 20 различных наборов данных. Из каждого набора было взято по 10 000 записей, которые подавались на вход алгоритмам кластеризации в формате фреймов.

Существуют разные методики оценки качества кластеризации, например оценка на основании компактности [15], в которой базовыми метриками качества могут служить среднее расстояние между парами элементов одного кластера (F_0) и среднее расстояние между парами элементов разных кластеров (F_1):

$$F_0 = \frac{\sum_{c=1}^{Nc} \sum_{i \neq j}^{Npt_{(c)}} \rho_{(x_{c,i}, x_{c,j})}}{\sum_{c=1}^{Nc} Npr_{(c)}} \rightarrow \min \tag{8}$$

$$F_1 = \frac{\sum_{ca \neq cb}^{Nc} \sum_{i,j=1}^{Npt_{(ca)}, Npt_{(cb)}} \rho_{(x_{ca,i}, x_{cb,j})}}{\sum_{ca,cb=1}^{Nc} Npr_{(ca,cb)}} \rightarrow \max \tag{9}$$

где c, ca, cb – номера кластеров, Nc – общее число кластеров, ρ – расстояние между точками, $x_{(c,i)}$ – точка с индексом i, принадлежащая кластеру c, Npr_c – число пар элементов внутри кластера c, $Npr_{(ca,cb)}$ – число пар элементов между кластерами ca и cb, $Npt_{(c)}$ – число элементов кластера c.

Однако, согласно общей методике оценки качества кластеризации, подходящей для любых алгоритмов, генерирующих выпуклые кластеры с выраженным центром, [16] в качестве базовых метрик качества можно использовать

$$\Phi_0 = \frac{\sum_{c=1}^{Nc} \sum_{i=1}^{Npt_{(c)}} \rho_{(x_{c,i}, \mu_c)}}{\sum_{c=1}^{Nc} Npt_{(c)}} \rightarrow \min \tag{10}$$

$$\Phi_1 = \sum_{ca \neq cb}^{Nc} \rho^2_{(\mu_{ca}, \mu_{cb})} \rightarrow \max \tag{11}$$

где c, ca, cb – номера кластеров, μ_c – геометрический центр кластера c, Nc – общее число кластеров, ρ – расстояние между точками, $x_{c,i}$ – точка с индексом i, принадлежащая кластеру c, $Npt_{(c)}$ – число элементов кластера c.

В этом случае, при работе с одномерными данными допустимо перейти от квадратов расстояний к прямым измерениям расстояний, выраженных в рамках данной работы как модули разностей координат точек данных. Слабым местом данного параметра является необходимость в масштабировании при сравнении результатов кластеризации для разных наборов данных. В данной работе проблема была решена следующим образом: была проведена обработка наборов данных при помощи классического DBSCAN, настроенного на сохранение всех точек данных, попавших в тот или иной кластер, затем было измерено среднее внутрикластерное расстояние для

полученных кластеров, а после вычислялось произведение аналогичных показателей других алгоритмов на показатели DBSCAN. Таким образом, приходим к формуле:

$$\Phi_0^{'} = \frac{\Phi_0}{\Phi_0^{DBSCAN}} \quad (12)$$

$$\Phi_1^{'} = \frac{\Phi_1}{\Phi_1^{DBSCAN}} \quad (13)$$

Результирующие оценки не зависят от конкретных расстояний внутри наборов данных.

На рис.6. отражены показатели Φ'_0, полученные в ходе эксперимента. Оценки Φ'_0 указаны по оси y. Как можно видеть, предлагаемый в данной работе алгоритм кластеризации (обозначенный как *DBSDSC*), показывает качество кластеризации более высокое либо равное, чем у DBSCAN. Это говорит о достаточно высокой компактности кластеров, сгенерированных в ходе работы. В то же время, результаты, показанные более быстрым алгоритмом (обозначенном как *1d*) значительно хуже, что подтверждает ранее озвученный тезис о низком качестве этого алгоритма.

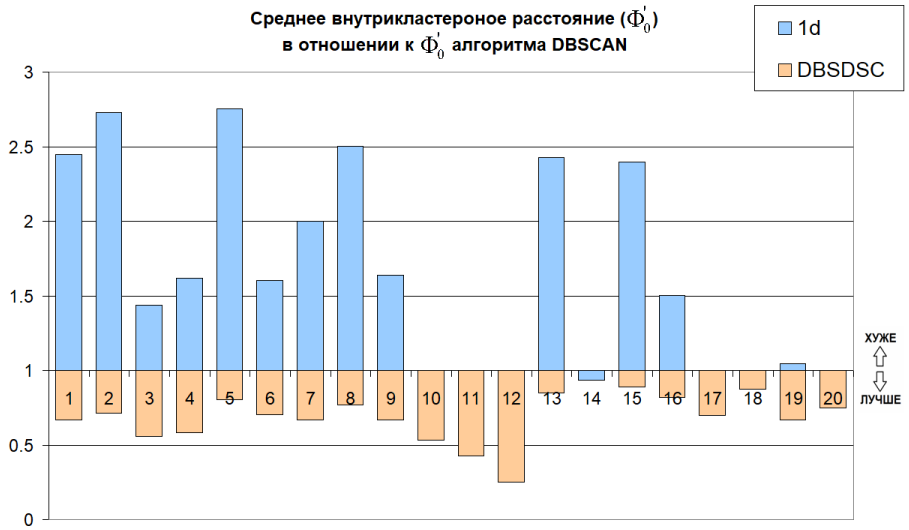

Рисунок 6. Сравнение среднего внутрикластерного расстояния

На рис.7. изображены показатели Φ'_1, полученные в ходе эксперимента. Условные обозначения аналогичны рис.6. Согласно полученным данным, DBSDSC в большинстве случаев показывает качество кластеризации более высокое либо равное, по сравнению с DBSCAN, за исключением нескольких выделенных случаев.

Далее приведены сравнительные характеристики времени, затраченного на обработку наборов данных при помощи тех же алгоритмов кластеризации потоковых одномерных данных. Результаты сравнения представлены на рис.8. Время указано по оси y в миллисекундах. Из-за большой разницы по времени обработки, была использована логарифмическая шкала.

Несложно заметить, что скорость обработки данных у DBSDSC более чем в 100 раз превышает таковую у неспециализированного DBSCAN. Как показал эксперимент, среднее время обработки фрейма из 100 точек данных при помощи алгоритма DBSDSC не превышала 300 мс, в среднем составляя около 160 мс. В то же время обработка фрейма при помощи алгоритма *1d* составила в среднем около 20 мс, что делает алгоритм до 10 раз быстрее по сравнению с DBSDSC. Разброс по времени обработки, продемонстрированный на рис.8. обусловлен особенностями распределения данных по пространству признаков.

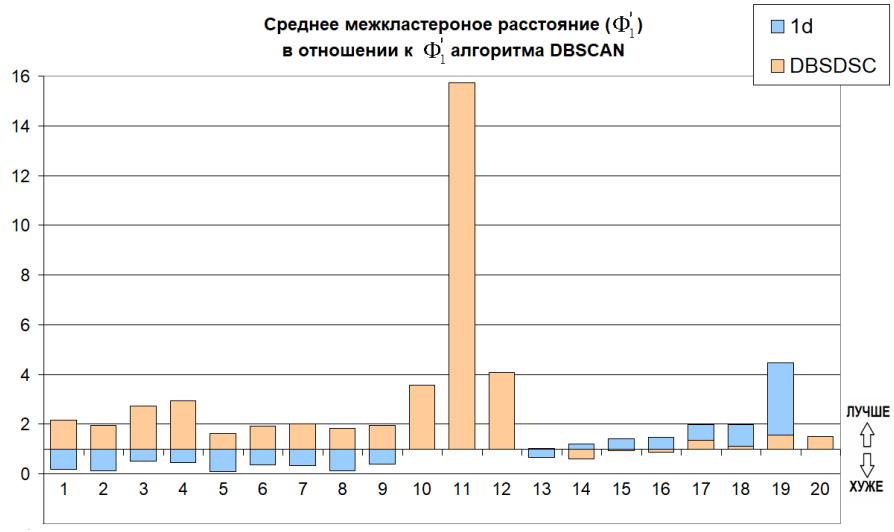

Рисунок 7. Сравнение среднего межкластерного расстояния

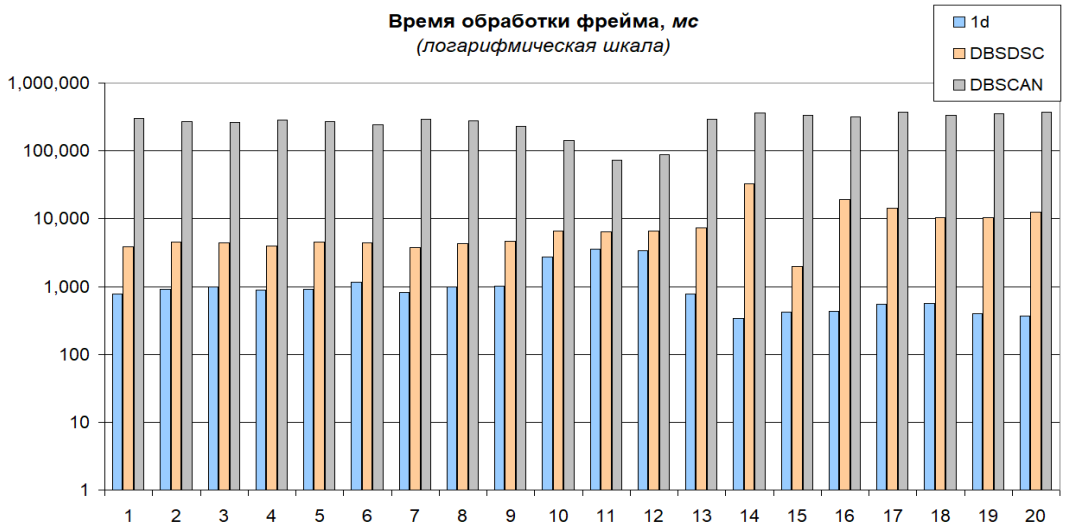

Рисунок 8. Сравнение времени обработки тестовой выборки

Заключение

Представленный в данной работе алгоритм кластеризации DBSDSC организует разделение потоковых данных на фреймы адаптивного размера, что позволяет регулировать скорость и качество кластеризации в зависимости от организации входящих данных. Также, алгоритм использует эвристический подход, опирающийся как на особенности одномерного пространства признаков, так и на накопительный характер информации о кластерах, которые можно обнаружить в исследуемом пространстве признаков. Как показали расчеты, подтвержденные на опыте, вычислительная сложность предложенного алгоритма стремится к линейной сложности по мере заполнения пространства признаков, что позволяет работать с неограниченным объемом данных, и, в частности, с потоковыми данными.

Алгоритм реализует сохранение результатов обработки предыдущих фреймов и использует их для ускорения дальнейшей кластеризации без потери качества. Итоговый расход памяти зависит от количества кластеров в пространстве признаков, а не от количества точек данных, содержащихся в этих кластерах. А конкретно, требуется не более чем ***2 * (MinPts)*** точек на каждый кластер, что является весьма экономным показателем в сравнении с большинством существующих алгоритмов. Несмотря на то, что представленный алгоритм не подразумевает хранения всех входящих точек данных, основные характеристики кластеров остаются такими же, как если бы все элементы кластеров сохранялись в качестве единой обучающей выборки. Таким образом, представленный в данной работе алгоритм не является алгоритмом с потерей данных в отличии от большинства алгоритмов обработки данных пригодных для работы с потоками.

Предложенный алгоритм опирается на принципы кластеризации на основании плотности распределения данных в пространстве признаков. Это позволяет осуществлять кластеризацию в условиях информационного шума, отбрасывая неинформативные аномальные данные.

Представленный алгоритм состоит из нескольких функциональных блоков, действующих как единая система в полностью автоматическом режиме. После запуска в работу и первоначальной настройки, вмешательство человека-оператора не требуется.

Подводя итог можно сказать, что в данной работе представлен оригинальный алгоритм кластеризации одномерных потоковых данных на основании плотности распределения, нацеленный на работу в условиях информационного шума. Научная новизна данной работы заключается в том, что, несмотря на существование одномерных модификаций классических алгоритмов кластеризации и алгоритмов кластеризации потоковых данных, на настоящий момент это один из немногих алгоритмов кластеризации на основании плотности распределения данных, предназначенных для работы с потоковыми данными, приспособленных к наличию информационного шума и одновременно оптимизированных для обработки одномерных данных.

Список литературы

1. К.А. Федутинов, Интеллектуализация процессов принятия решений в организационных системах в условиях оперативного анализа мониторинговых данных // автореф. дис.канд. техн. наук - Воронеж – 2023
2. А.В. Панов, Г.В. Митин, Сравнительный анализ современных методов машинного обучения в контексте специфики типов их выходных значений // Актуальные проблемы прикладной математики, информатики и механики: сборник трудов Международной научной конференции, 12-14 декабря 2022 г. Воронеж, 2023. - С. 1426-1435
3. Haizhou Wang, Mingzhou Song, Ckmeans.1d.dp: Optimal k-means Clustering in One Dimension by Dynamic Programming // The R Journal Vol. 3/2, December 2011 p 29-33
4. Bartosz Meglicki, Linear time DBSCAN for sorted 1D data and laser range scan segmentation // ResearchGate, URL: https://www.researchgate.net/publication/350512143_Linear_time_DBSCAN_for_sorted_1D_data_and_laser_range_scan_segmentation (access time: 08.02.2024)
5. О.В. Ниссенбаум, Алгоритм кластеризации потока данных с изменяющимися параметрами распределения // Вестник Тюменского государственного университета. – 2013. – №7
6. В.А. Хархинов, Нейросетевые технологии решения задач кластеризации и классификации данных в технических системах // автореф. дис. канд. техн. наук - Иркутск – 2023
7. Mu C, Hou Y, Zhao J, Wei S, Wu Y, Stream-DBSCAN: A Streaming Distributed Clustering Model for Water Quality Monitoring // Applied Sciences. 2023 Apr 26;13(9):5408.
8. Г.В. Митин, А.В. Панов, Адаптивная технология классификации с обратной связью на основе методов машинного обучения // Современные наукоемкие технологии. – 2024. – № 1 – С. 55-61
9. Martin Ester, Hans-Peter Kriegel, Jorg Sander, Xiaowei Xu, A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise // From: KDD-96 Proceedings. Copyright © 1996, -p 226-231
10. Г.В. Митин, А.В. Панов, Модификация алгоритма DBSCAN с использованием гибридных подходов к определению границ кластеров для обработки потоковых данных // ИТ-Стандарт – 2023. – № 4 – С. 32-53
11. Electrical Fault detection and classification / Kaggle, Available at https://www.kaggle.com/datasets/esathyaprakash/electrical-fault-detection-and-classification (accessed: 12/19/2023)
12. Individual household electric power consumption / UC Irvine Machine Learning Repository, Available at https://archive.ics.uci.edu/dataset/235/individual+household+electric+power+consumption (accessed: 12/19/2023)
13. Online shoppers intention / UC Irvine Machine Learning Repository, Available at https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset (date of application: 12/19/2023)
14. Motion capture hand postures / UC Irvine Machine Learning Repository, Available at https://archive.ics.uci.edu/dataset/405/motion+capture+hand+postures (accessed: 12/19/2023)
15. А.А. Кузьмин, Иерархическая классификация коллекций документов // дис.канд. техн. наук - Москва - 2017

16. К.В. Воронцов, Лекции по алгоритмам кластеризации и многомерного шакалирования // URL: http://www.ccas.ru/voron/download/Clustering.pdf (дата обращения: 01/03/2024)

References

1. K.A. Fedutinov, Intellectualization of decision-making processes in organizational systems in the context of operational analysis of monitoring data // abstract of the dissertation of the Candidate of Technical Sciences – Voronezh – 2023

2. A.V. Panov, G.V. Mitin, Comparative analysis of modern machine learning methods in the context of the specifics of the types of their output values // Actual problems of applied mathematics, computer science and mechanics: proceedings of the International Scientific Conference, December 12-14, 2022 Voronezh, 2023. - pp. 1426-1435

3. Haizhou Wang, Mingzhou Song, Ckmeans.1d.dp: Optimal k-means Clustering in One Dimension by Dynamic Programming // The R Journal Vol. 3/2, December 2011 p 29-33

4. Bartosz Meglicki, Linear time DBSCAN for sorted 1D data and laser range scan segmentation // ResearchGate, URL: https://www.researchgate.net/ publication/350512143_Linear_time_DBSCAN_for_sorted_1D_data_and_laser_ range_scan_segmentation (access time: 02/08/2024)

5. O.V. Nissenbaum, Data flow clustering algorithm with changing distribution parameters // Bulletin of the Tyumen State University. – 2013. – №7

6. V.A. Kharkhinov, Neural network technologies for solving problems of clustering and classification of data in technical systems // abstract of the dissertation of the Candidate of Technical Sciences - Irkutsk – 2023

7. Mu C, Hou Y, Zhao J, Wei S, Wu Y, Stream-DBSCAN: A Streaming Distributed Clustering Model for Water Quality Monitoring // Applied Sciences. 2023 Apr 26;13(9):5408.

8. G.V. Mitin, A.V. Panov, Adaptive feedback classification technology based on machine learning methods // Modern high-tech technologies. – 2024. – No. 1 – pp. 55-61

9. Martin Ester, Hans-Peter Kriegel, Jorg Sander, Xiaowei Xu, A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise // From: KDD-96 Proceedings. Copyright © 1996, -p 226-231

10. G.V. Mitin, A.V. Panov Modification of the DBSCAN algorithm using hybrid approaches to defining cluster boundaries for processing streaming data // IT-Standard – 2023. – No. 4 – pp. 32-53

11. Electrical Fault detection and classification / Kaggle, Available at https://www.kaggle.com/datasets/ esathyaprakash/ electrical-fault-detection-and-classification (accessed: 12/19/2023)

12. Individual household electric power consumption / UC Irvine Machine Learning Repository, Available at https://archive.ics.uci.edu/dataset/235/individual+household+electric+power+consumption (accessed: 12/19/2023)

13. Online shoppers intention / UC Irvine Machine Learning Repository, Available at https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset (date of application: 12/19/2023)

14. Motion capture hand postures / UC Irvine Machine Learning Repository, Available at https://archive.ics.uci.edu/dataset/405/motion+capture+hand+postures (accessed: 12/19/2023)

15. A.A. Kuzmin, Hierarchical classification of document collections // Dissertation of the Candidate of Technical Sciences - Moscow - 2017

16. K.V. Vorontsov, Lectures on clustering and multidimensional jackal algorithms // URL: http://www.ccas.ru/voron/download/Clustering.pdf (accessed: 01/03/2024)