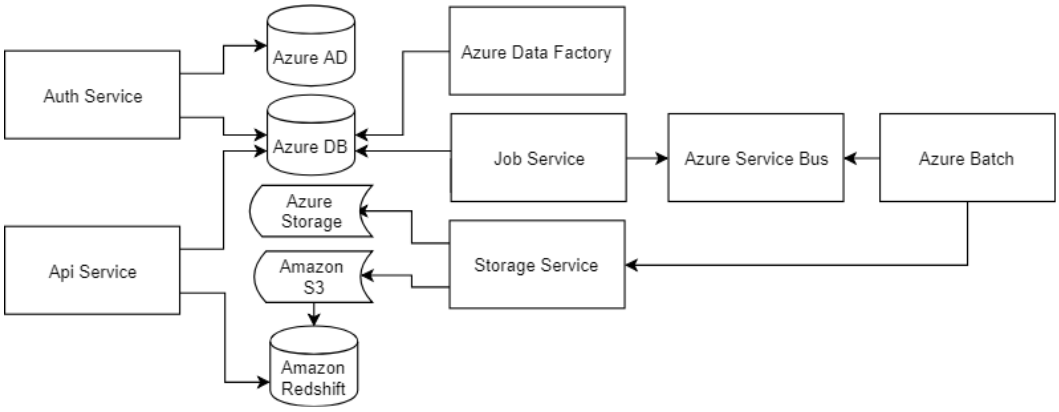

Рис. 1 – Архитектура системы для автоматизации контроля доступа к SAP

Решением проблемы стало использование программного инструмента Terraform от компании HashiCorp. Terraform предоставляет библиотеки для управления большинством сервисов всех крупных облачных поставщиков. Для достижения целей были использованы библиотеки для облаков Azure и AWS. Данное решение помогло описать используемые ресурсы в декларативном стиле, и, в последствии, использовать для создания новых окружений без особых сложностей. Описание каждого ресурса создавалось в отдельном файле на языке конфигураций HCL (Hashicorp Configuration Language). Так же была использована система контроля версий Git, что позволяет видеть изменения в инфраструктуре.

Список использованных источников:

1. Infrastructure as code [Электронный ресурс]. – Режим доступа: https://en.wikipedia.org/wiki/Infrastructure_as_code – Дата доступа: 24.03.2019.
2. Terraform Documentation [Электронный ресурс]. – Режим доступа: https://www.terraform.io/docs/index.html – Дата доступа: 24.03.2019.
3. Terraform on Azure [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/en-us/azure/terraform/ – Дата доступа: 24.03.2019.

ИСПОЛЬЗОВАНИЕ АЛГОРИТМА КЛАСТЕРИЗАЦИИ DBSCAN ДЛЯ ФИЛЬТРАЦИИ ВЫБРОСОВ В ДАННЫХ

Ковалёв С.П.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Калугина М.А. – к. ф.-м. н., доцент

При обработке больших объемов данных часто используется такой метод машинного обучения как кластеризация, который помогает найти группы похожих объектов в наборе данных. Но для успешной кластеризации набора данных его необходимо предварительно обработать – очистить данные от выбросов. Одним из таких способов является использование способности алгоритма DBSCAN определять выбросы в наборе кластеризуемых данных.

Кластеризация относится к методам обучения «без учителя». Эти методы не предполагают, что существует правильный ответ, для получения которого можно обучить алгоритм. Кластеризация призвана систематизировать данные и найти группы похожих объектов.

Методы кластерного анализа используются в психологии, археологии, биологии. В настоящее время кластеризация также активно применяется в социологии, экономике, статистике, в исторических исследованиях и многих других сферах. Страховые компании могут разделять автомобили по группам риска, а затем назначать каждой группе различную стоимость страховки. В вычислительной биологии этот метод активно применяется для обнаружения групп генов, демонстрирующих сходное поведение. Это может служить указанием на то, что они будут одинаково реагировать на лечение или что образовались на одном пути биологического развития [1]. Особенно расширилось использование этого метода машинного обучения в связи с появлением и развитием ЭВМ, поскольку это связано с трудоемкостью обработки больших объемов информации.

Алгоритмы кластеризации можно рассматривать как иерархические и неиерархические. Результатом работы иерархических алгоритмов является разбиение данных по уровням. В этом случае исследователь может выбрать любую ступень построенной иерархии при интерпретации

результатов. Неиерархическими считаются все алгоритмы, результатом которых не является такая процедура (или выбор интерпретации происходит не по уровню иерархии).

Различные методы кластеризации имеют свои ограничения и могут выделять кластеры определенных типов и форм. Например, одни из самых известных алгоритмов, использующие методы K-средних и агломеративной иерархической кластеризации, хорошо и быстро находят кластеры сферической формы, но неспособны выделить кластеры необычной несферической формы (вытянутые, закругленные). Различные библиотеки для машинного обучения предоставляют примеры нахождения кластеров [2]. Эти методы распределяют абсолютно все данные по кластерам, учитывая и точки, которые однозначно можно назвать выбросами. Эти точки, не принадлежащие ни к какому из кластеров, являются аномальными данными, которые не следует учитывать. Из-за них положение центра кластера при использовании вышеуказанных методов искривляется и при анализе результатов можно получить неверные выводы.

В свою очередь, алгоритм кластеризации DBSCAN решает проблему выбросов в наборе данных. DBSCAN (Density-based spatial clustering of applications with noise) – молодой алгоритм кластеризации, созданный только в 1996 году [3]. В отличие от множества других алгоритмов он не требует указания количества кластеров и может находить произвольное их количество. Также, в отличие от названных выше, этот алгоритм может находить кластеры различных типа и формы – главное, чтобы точки находились рядом.

На вход необходимо передать два параметра – радиус ε окрестности и количество m соседей. DBSCAN не может хорошо кластеризовать наборы данных с большой разницей в плотности, поскольку не удается выбрать приемлемую для всех кластеров комбинацию этих входных параметров.

Этот алгоритм способен хорошо находить шумы и выбросы в данных. Происходит это в тех случаях, когда точку невозможно отнести ни к какому из существующих кластеров, то есть в окрестности данной точки находится мало точек-соседей.

Именно способность данного алгоритма находить выбросы можно использовать для фильтрации этих самых выбросов. После того, как в наборе данных не останется выбросов, каждый кластер будет точно описывать группу, на положение центра кластера не будут влиять выбросы и шумы и можно гораздо лучше проводить анализ данных.

Пример использования алгоритма кластеризации DBSCAN приведен на рисунке 1. На левой половине изображены все точки исходного набора данных. На правой - уже очищенные с помощью DBSCAN.

В данном примере использован набор данных, симулирующий поведение пользователей в системе по управлению проектами небольших групп по примеру сайта trello.com. Пользователи могут регистрироваться, заходить в аккаунт, открывать главную страницу, открывать доску с карточками, создавать карточки и т.д.

При симуляции поведения пользователей были заданы 2 типа пользователей: корпоративные и персональные. Поэтому на рисунке легко обнаружить 2 кластера. Корпоративные пользователи, которые были симулированы для этого набора данных, пользуются сервисом активно в будние дни, почти не используют в выходные, создают много событий. Группа персональных пользователей использует приложение для личных нужд каждый день недели, но в заметно меньшем объеме. Набор данных включает события 546 пользователей в течение нескольких недель.

Данные для каждого пользователя представляют собой набор характеристик его деятельности – общее количество событий различного типа, количество событий в среднем по дням недели и т.д. В итоге получается около 30 характеристик, поэтому не представляется возможным отобразить пользователей на двумерной плоскости. В этом случае можно уменьшить размерность данных с помощью метода главных компонент (англ. principal component analysis, PCA). Это один из основных способов уменьшить размерность данных, потеряв наименьшее количество информации.

На рисунке с исходными данными легко определить, что здесь представлены две группы точек. Группа с большим разбросом точек слева – корпоративные пользователи. И небольшая группа с плотно лежащими точками справа, которая в основном состоит из персональных пользователей. Но группы выделены не совсем четко: много точек между кластерами и на их границах. Эти выбросы плохо влияют на положение центра кластера. Хорошим решение является избавление от этих точек.

Исходный набор данных состоит из 546 точек. С помощью алгоритма DBSCAN удалось обнаружить 104 выброса в наборе данных. Недостатком этого способа является исключение этих точек из дальнейшего анализа. Но большим плюсом является возможность более четко и качественно выделить кластеры, найти близкое к истинному положение центров каждого кластера.

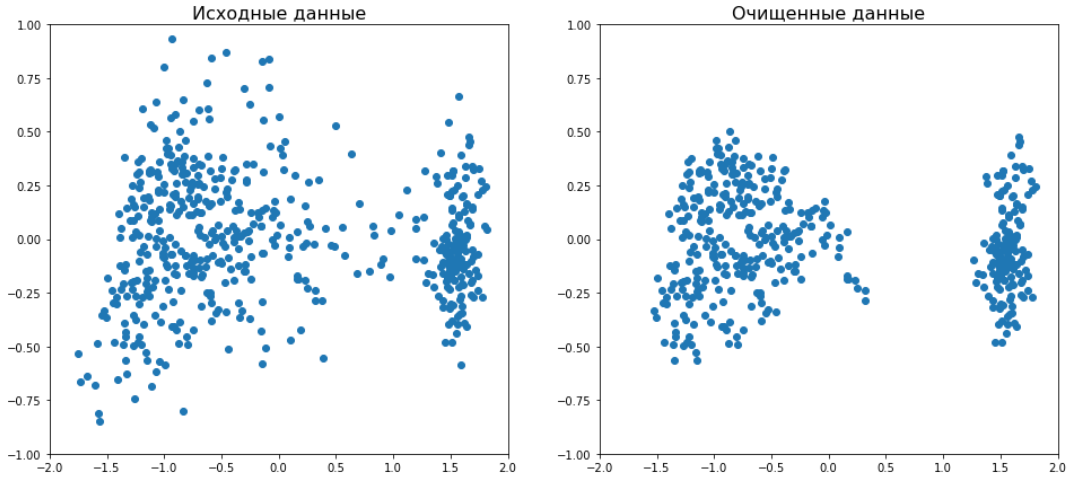

Рисунок 1 – Исходные и очищенные данные

Теперь посмотрим на результаты фильтрации выбросов алгоритмом DBSCAN. В связи с тем, что были убраны выбросы в данных, кластеры приобрели более четкие формы.

При применении метода K-средних на исходном наборе данных группа точек справа на 20% состояла из корпоративных пользователей и на 80% из персональных. То есть в этот кластер попало много объектов, которые изначально должны были попасть в другой кластер. Из-за некоторых аномалий в поведении эти пользователи находятся между двумя кластерами и при кластеризации отнесены к неправильной группе. Таким образом, центр кластера смещен относительно его правильного положения.

При использовании набора очищенных данных группа точек справа теперь только на 7% состоит из корпоративных пользователей, все остальные – персональные. Это значит, что теперь этот кластер более точно описывает группу персональных пользователей, центр кластера находится ближе к его истинному значению.

Таким образом, использование алгоритма кластеризации DBSCAN на практике позволяет очистить и подготовить набор данных для дальнейшего анализа, полученные при этом кластеры выделяются в более отчетливые формы, что несомненно приносит пользу, когда исследователю необходимо наиболее точно охарактеризовать свойства кластеров.

Список использованных источников:

1. Сегаран Т. Программируем коллективный разум. / Сегаран. Т. – Пер. с англ. – СПб: Символ-Плюс, 2008. — 50-51 с.

2. Comparing different clustering algorithms on toy datasets [Электронный ресурс]. — Режим доступа: https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html/. — Дата доступа: 05.03.2019.

3. Ester, M., Kriegel, H. P., Sander, J., & Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. / Ester, M., Kriegel, H. P., Sander, J., & Xu, X. // KDD. — 1996. — Vol. 96, №34 — P. 226–231.

БАЙЕСОВСКАЯ ОПТИМИЗАЦИЯ

Козак А. В., Сухов Н. Ю.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Теслюк В. Н. – доцент, кандидат физико-математических наук

В данной работе рассмотрен алгоритм глобальной оптимизации функций без использования производных. Приведен класс функций, для которых может быть использована байесовская оптимизация. Рассмотрены потенциальные проблемы использования байесовской оптимизации. Приведены потенциальные области для улучшения и дальнейшего исследования данного класса оптимизационных алгоритмов.

Байесовская оптимизация – это подход к оптимизации целевых функций, оценка и подсчет которых требует большого числа ресурсов, например, времени (минуты или часы). Она лучше всего подходит для оптимизации непрерывных функций, определенных на многомерном пространстве размерности не более 20 и допускающих стохастический шум в значениях самой функций.