ЛЕКЦІЯ
ДЕТЕКТУВАННЯ АНОМАЛІЙ
(Anomaly Detection)

ПЛАН

Вступ
1. Поняття аномальності та постановка задачі виявлення аномалій
2. Методи виявлення аномалій
3 Методи покращення алгоритмів
Висновки

ЛІТЕРАТУРА

https://habr.com/ru/post/491552/
https://proglib.io/p/moem-dataset-rukovodstvo-po-ochistke-dannyh-v-python-2020-03-27
https://waksoft.susu.ru/2020/04/12/midas-novoe-napravlenie-poiska-anomalij/
https://newtechaudit.ru/biblioteka-pyod-sravnivaem-algoritmy-poiska-vybrosov/

ВСТУП

Виділення в даних (датасетах) нетипових, аномальних представників є особливим завданням у машинному навчанні. Крім низки практичних застосувань (виявлення збоїв у показаннях датчиків, хакерських атак, незвичайних результатів діагностик), це завдання є етапом побудови будь-якого алгоритму машинного навчання, у якому дані перевіряються на консистентність та очищаються від викидів і шуму.

У цій лекції поговоримо про одну важливу і досить специфічну проблему навчання – завдання **пошуку аномалій (Anomaly Detection)**.

Існує три широкі категорії методів виявлення аномалій. **Методи контрольованого виявлення аномалій** вимагають набору даних, позначеного як «нормальний» і «ненормальний», і включають навчання класифікатора. Однак цей підхід рідко використовується для виявлення аномалій через загальну недоступність розмічених даних і притаманну класам незбалансованість. **Методи напівконтрольованого виявлення аномалій** припускають, що певна частина даних позначена. Це може бути будь-яка комбінація звичайних або аномальних даних, але найчастіше методи будують модель, що представляє нормальну поведінку, на основі заданого нормального набору даних навчання, а потім перевіряють ймовірність тестового екземпляра, який буде згенерований моделлю. **Методи неконтрольованого виявлення аномалій** припускають, що дані не позначені (нерозмічені), і вони, безумовно, найчастіше використовуються через їх ширше та частіше застосування.

1. Поняття аномальності та постановка задачі виявлення аномалій

1.1. Поняття аномальності

Строго кажучи, в аналізі даних є два напрямки, які займаються пошуком аномалій: **детектування викидів** (Outlier Detection) та **«новизни»** (Novelty Detection). Як і викид **"новий об'єкт"** - це об'єкт, який відрізняється за своїми властивостями від об'єктів (навчальної) вибірки. Але на відміну від викиду, його в самій вибірці поки немає (він з'явиться через деякий час, і завдання якраз і полягає в тому, щоб виявити його з появою). Наприклад, якщо ви аналізуєте виміри температури та відкидаєте аномально великі чи маленькі, то Ви боретеся з викидами. А якщо Ви створюєте алгоритм, який для кожного нового виміру стану мережі оцінює, наскільки він схожий на минулі, і виділяє аномальні — Ви шукаєте *новизну (по суті створюєте детектор вторгнень)*.

Викиди є наслідком:

• помилок у даних (неточності вимірювання, округлення, неправильного запису тощо)

• наявності шумових об'єктів (неправильно класифікованих об'єктів)

• присутності об'єктів «інших» вибірок (наприклад, показаннями датчика, що зламався).

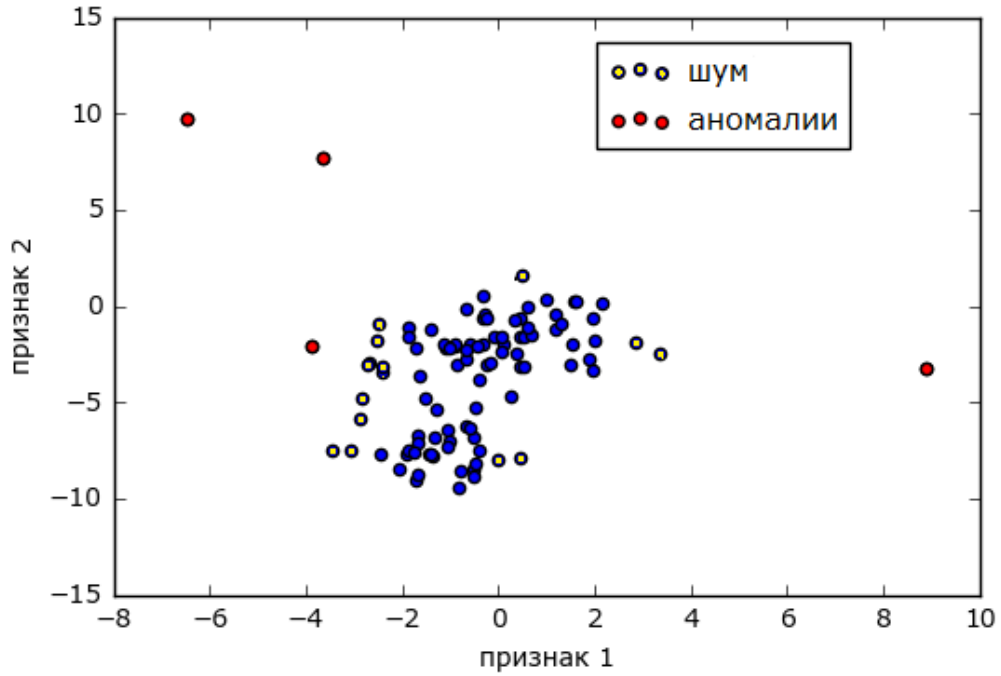

Рис. 1. Модельне завдання з двома ознаками

На рис. 1 видно, що шум (noise) - це викид "у слабкому сенсі" (він може трохи розмивати межі класу/кластера). Нас же цікавлять насамперед викиди «в сильному розумінні», які спотворюють ці межі.

Новизна, як правило, з'являється в результаті нової поведінки об'єкта. Скажімо, якщо наші об'єкти – опис роботи системи, то після проникнення в неї вірусу об'єкти стають *«новизною»*. Ще приклад – опис роботи двигуна після поломки. *Тут важливо розуміти, що «новизна» називається новизною з тієї причини, що* ***такі описи для нас абсолютно нові****, а нові вони тому, що ми не можемо в навчальній вибірці мати інформацію про всілякі зараження вірусами або всілякі поломки.* Формування такої навчальної вибірки трудомістке і часто не має сенсу. Проте можна набрати досить велику вибірку прикладів нормальної (штатної) роботи системи чи механізму.

Застосунків тут море:

• Виявлення підозрілих банківських операцій (Credit-card Fraud)
• Виявлення вторгнень (Intrusion Detection)
• Виявлення нестандартних гравців на біржі (інсайдерів)
• Виявлення неполадок у механізмах показання датчиків
• Медична діагностика (Medical Diagnosis)
• Сейсмологія

Варто зазначити, що *можливих постановок завдань тут також багато*. *Наприклад,* завдання *Positive-Unlabeled Classification (PU learning)* - це коли частина викидів позначена (клас 1), але в інших об'єктах навчання (клас 0) також можуть бути викиди. Наприклад, нам експерт сказав, що обладнання давало збій у такі моменти часу, але він міг помітити не всі збої.

Навіть коли завдання виявлення аномалій схожі на звичайні завдання класифікації, є особливості, скажімо, дисбаланс класів (наприклад, поломки обладнання відносно рідкісні).

Аномалії бувають у табличних даних, можуть бути у графах, часових рядах тощо.

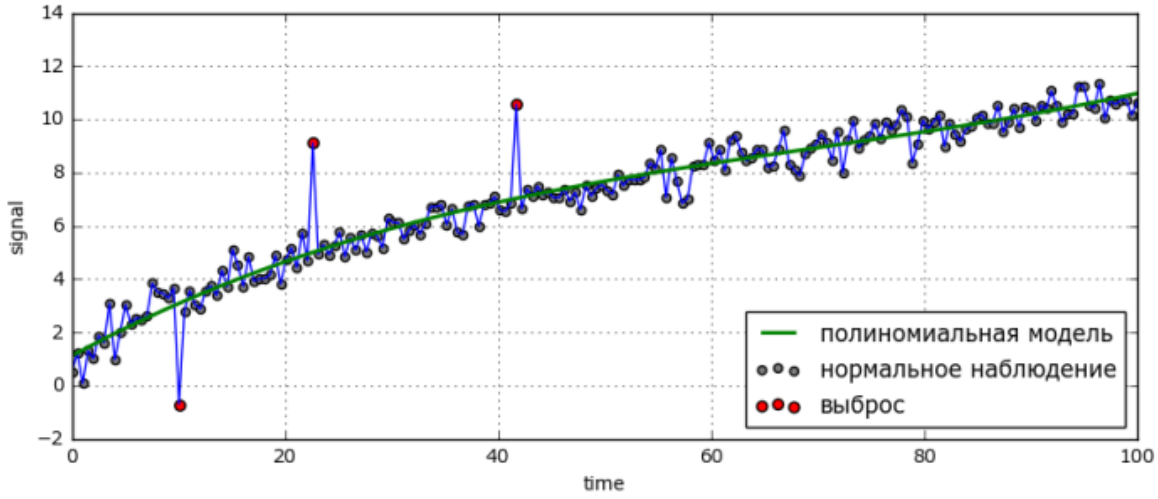

Рис. 2. Приклад викидів у часовому ряді.

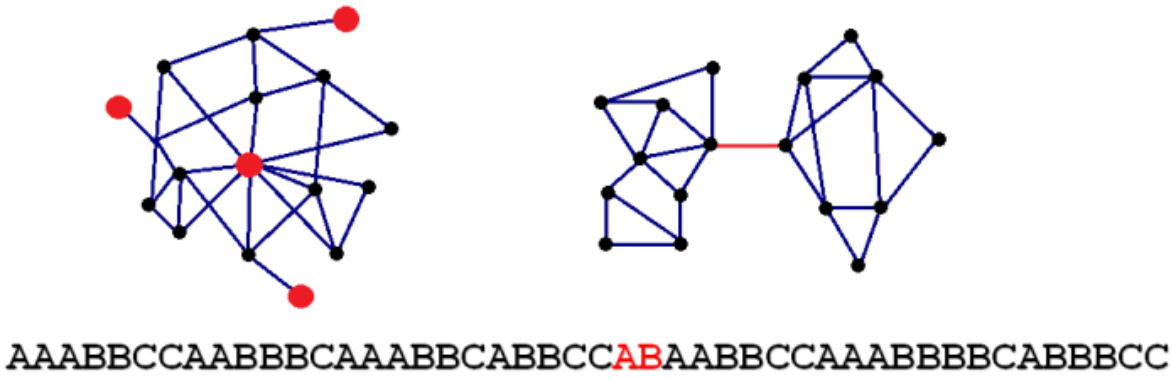

Рис. 3. Приклад викидів у графах та послідовностях.

Деякі визначення аномальності

У спільнотах статистики та комп'ютерних наук було зроблено багато спроб визначити аномалію. До найпоширеніших відносяться:

Викид (outlier) - це спостереження, яке настільки відрізняється від інших спостережень, що викликає підозру, що воно було зроблено іншим механізмом. [2]

Аномалії (Anomalies) - це екземпляри або набори даних, які дуже рідко зустрічаються в наборі даних та характеристики яких значно відрізняються від більшості даних.

Викид (outlier) - це спостереження (або підмножина спостережень), яке здається несумісним з рештою цього набору даних. [3]

Аномалія (anomaly) – це точка або набір точок, які відносно віддалені від інших точок у багатовимірному просторі ознак.

Аномалії - це закономірності в даних, які не відповідають чітко визначеному поняттю нормальної поведінки. [1]

Нехай T — спостереження з одномірного розподілу Гаусса, а O — точка з T. Тоді z-показник для O більший за попередньо вибраний поріг тоді і тільки тоді, коли O є викидом.

1.2 Постановка задачі виявлення аномалій

Завдання детектування аномалій немає єдиного формулювання і найчастіше інтерпретуються по-різному залежно від характеру даних і поставленої мети [1, 3, 5]. На інтуїтивному рівні аномаліями називають те, що не вписується у загальні правила та закони, справедливі для поданих даних. Таке визначення потребує формального уточнення, перш ніж вирішувати завдання математичними методами.

Будь-які методи детектування аномалій спираються на деяке суворе уявлення про те, що означає «відхилення від норми». У окремих випадках, вже це уявлення може відштовхуватися від контексту завдання та апріорної інформації. Наприклад, у найпростішому випадку, якщо дані є набором елементів деякої множини, будь-який елемент, що не належить цій множині, апріорі можна вважати аномалією. Проте така «характеристична» множина, за приналежністю до якої можна робити висновок про аномальність елемента, існує не завжди. Наприклад, якщо об'єктами є показання деякого датчика, а аномаліями – показання зламаного датчика, може статися так, що і зламаний, і правильно працюючий датчики видадуть той самий результат.

Істотно різниться і природа появи аномалій у даних. Це може бути як шум, тобто дані, що утворилися найчастіше випадковим чином, так і рідкісні явища, що становлять інтерес і потребують додаткового вивчення (наприклад, поява сторонніх об'єктів на знімках). В останньому випадку аномалії мають особливу природу і можуть навіть виділятися в відокремлені кластери (рис. 4).

Підходи до виявлення аномалій різних "видів" можуть суттєво відрізнятися.

Оскільки різні алгоритми розпізнавання аномалій походять від різних визначень аномальності, відрізняється не тільки їх результат, а й підхід.

Можливе й протилежне: пропонований підхід визначає інтерпретацію аномальності. Наприклад, якщо як дані взяти набір чисел, то можна вважати, що видалення з нього аномалій знизить дисперсію; потім можна знайти в наборі підмножину чисел (наприклад, заданого розміру), видалення яких максимально знизить дисперсію, і оголосити ці точки аномаліями.

Математична постановка задачі

Надалі будемо вважати, що дані мають ознакове уявлення, тобто кожен об'єкт *x* заданий у вигляді деякого вектора $\mathbf{R}^{\mathbf{d}}$. У класичній постановці завдання детектування аномалій формулюється так: у заданій множині **X** для кожного елемента видати **0** якщо цей об'єкт відноситься **до класу нормальних даних**, та **1**, якщо **цей об'єкт аномальний**. Таке завдання відноситься до класу завдань *навчання без вчителя*, оскільки правильних відповідей на частини вхідних даних не надається.

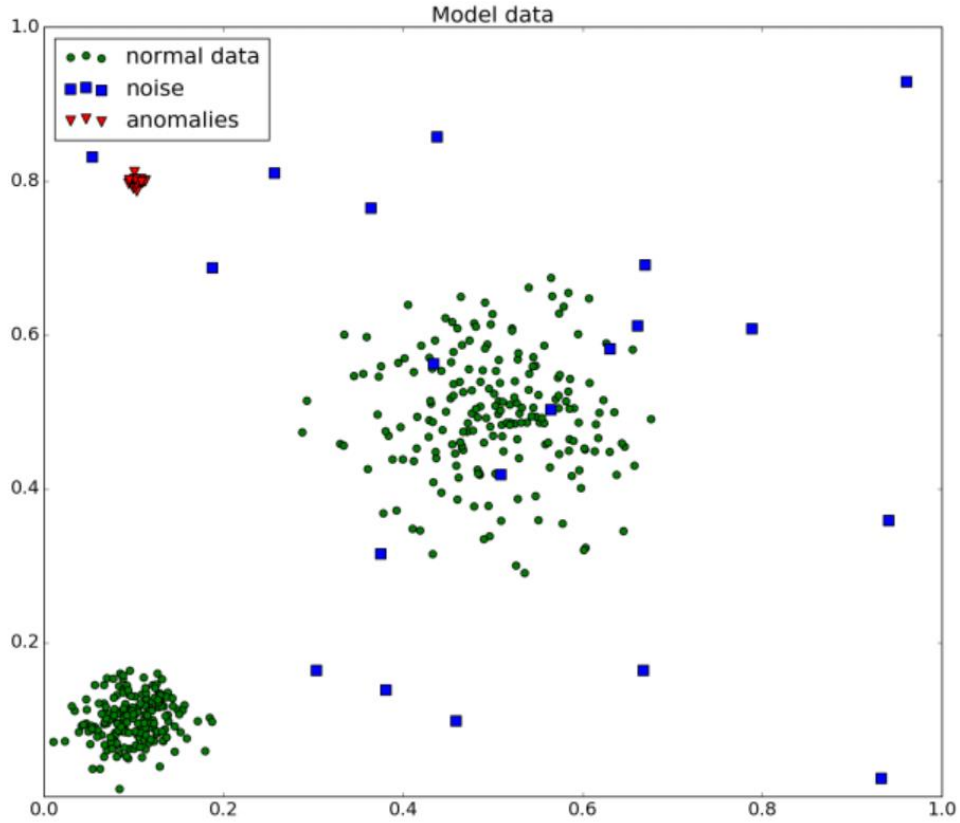

Рис. 4. Приклад модельної задачі. Нормальні дані є двома кластерами, семпльованими з нормального розподілу. Аномалії утворюють окремий кластер малого розміру. Крім того, є шум, семпльований з рівномірного розподілу

В аналогічному завданні *навчання з учителем* на деякій частині $\mathbf{X_{train}}$ вхідних даних відома вірна відповідь, тобто для кожного об'єкта $\boldsymbol{x} \in \mathbf{X_{train}}$ відомі мітки $\mathbf{y(x)} \in \mathbf{\{0, 1\}}$ – чи є об'єкт аномалією. Завдання видачі міток для нових даних, $\mathbf{X_{test}}$, формально є завданням бінарної класифікації, і, отже, може вирішуватися з допомогою будь-яких алгоритмів машинного навчання з учителем. Однак, можливий і «проміжний варіант», коли всі мітки $\mathbf{y(x)}$, $\boldsymbol{x} \in \mathbf{X_{train}}$ дорівнюють 0, тобто задані приклади лише нормальних («перевірених», «хороших») даних. У такому випадку алгоритми розв'язання задачі бінарної класифікації видаватимуть нерелевантний константний прогноз.

Варто зазначити, що ця проблема, нехай меншою мірою, присутня і у випадку, коли в навчальній вибірці є приклади аномалій: алгоритми навчання з учителем на даних, що не мають аналогів у навчанні, видають у випадку випадкову відповідь (причому часто, одиниця видається з ймовірністю $P(y(x) = 1)$, оціненої за навчальною вибіркою). З цієї причини є сенс розглядати завдання саме як завдання навчання без вчителя, а наявність навчальної вибірки (міток) – як можливість додатково налаштовувати параметри алгоритму.

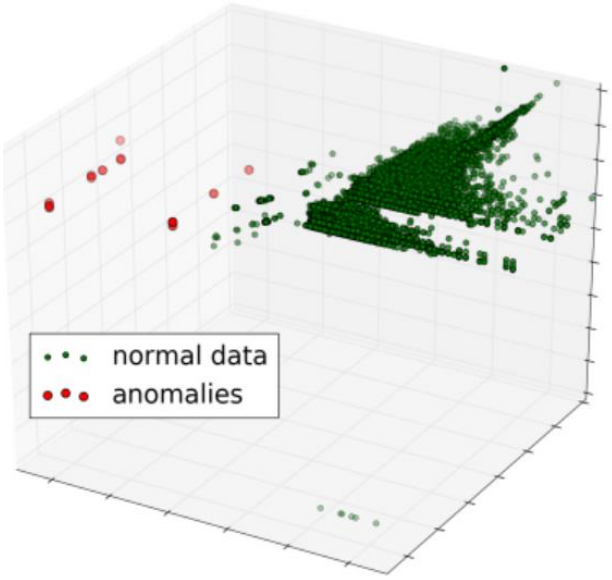

Рис. 5. Дані датасета Smtp із репозиторія [12]

Практично всі алгоритми детектування аномалій зводяться до побудови деякої функції *anomaly_score(x)*, яка по кожному об'єкту видає певний «рейтинг» аномальності. Після цього поділ на клас аномалій і клас нормальних даних проводиться бінаризацією по деякому порогу, вибір якого є особливим етапом розв'язання задачі. В умовах відсутності міток або апріорної інформації, єдиною наявною інформацією є *одномірний розподіл* значень *anomaly_score* на наявних даних, чого для обґрунтованого вибору недостатньо. Найчастіше відома приблизна частка аномалій даних, у таких випадках як поріг вибирається відповідний квантиль одномірного розподілу.

Функціонали якості в задачах детектування аномалій використовують приблизно такі самі, як і в задачах класифікації: PR AUC, AUROC, і визначаються контекстом завдання (замовником).

2. Методи виявлення викидів

2.1. Статистичні випробування (експерименти)

Як правило, застосовують для окремих ознак та відловлюють екстремальні значення (Extreme-Value Analysis). Для цього використовують, наприклад, Z-Value або Kurtosis measure.

$$\text{Z-value } Z_i = \frac{|x_i - \mu|}{\sigma} \qquad \text{Kurtosis } \frac{1}{n}\sum_{i=1}^{n} Z_i^4$$

Будь-який практик має якийсь свій перевірений спосіб знаходження екстремальних значень для певних типів даних. Багато методів візуалізації, наприклад ящик з вусами, мають вбудовані засоби для детектування та показу таких екстремальних значень.

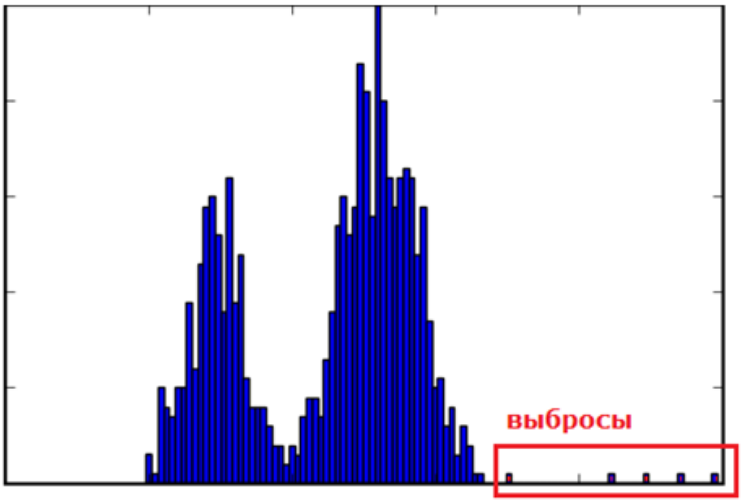

Рис. 6. Приклад викидів.

Важливо розуміти, що екстремальне значення та аномалія – це різні поняття. *Наприклад*, у невеликій вибірці

[1, 39, 2, 1, 101, 2, 1, 100, 1, 3, 101, 1, 3, 100, 101, 100, 100]

значення 39 можна вважати аномалією, хоча воно не є максимальним чи мінімальним. Також варто зазначити, що аномалія характеризується, як правило, не лише екстремальними значеннями окремих ознак (див. рис. 7).

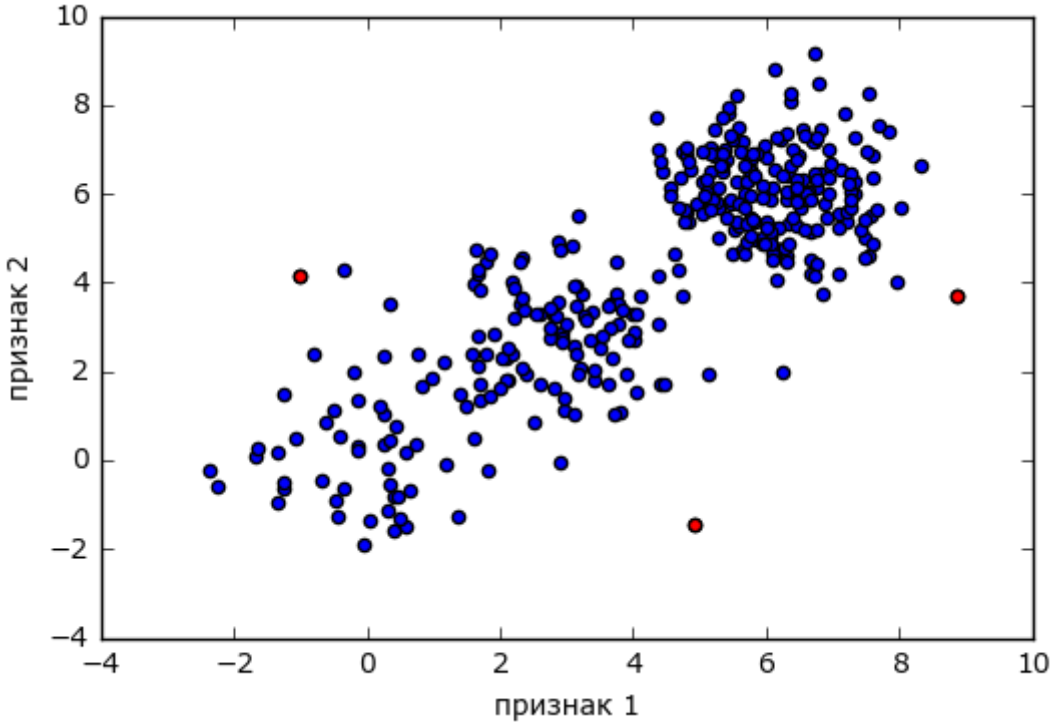

Рис. 7. Приклад викидів у задачі із двома ознаками.

2.2. Модельні тести

Ідея дуже проста – ми будуємо модель, яка описує дані. Точки, які сильно відхиляються від моделі (на яких модель сильно помиляється) і є аномалії (див. рис. 2). При виборі моделі ми можемо зважити на природу завдання, функціонал якості тощо.

Такі методи хороші для визначення новизни, але гірше працюють у пошуку викидів. Дійсно, при налаштуванні моделі ми використовуємо дані, в яких є викиди (і вона під них «заточується»).

На рис. 8 показано застосування модельного підходу. Ми маємо матрицю і потрібно знайти в ній викиди. Ми використовуємо неповне сингулярне розкладання (SVD), щоб знайти матрицю невеликого рангу максимально схожу на нашу (для наочності всі числа округлені). Елементи, які сильно відрізняються від відповідних елементів матриці невеликого рангу, вважатимемо викидами.

Рис. 8. Застосування SVD для знаходження викидів у матриці

2.3. Ітераційні методи

Методи, які складаються з ітерацій, на кожній із яких видаляється група «особливо підозрілих об'єктів». Наприклад, у n-вимірному ознаковому просторі можна виділяти опуклу оболонку наших точок-об'єктів, вважаючи її представників викидами. Як правило, методи цієї групи досить трудомісткі.

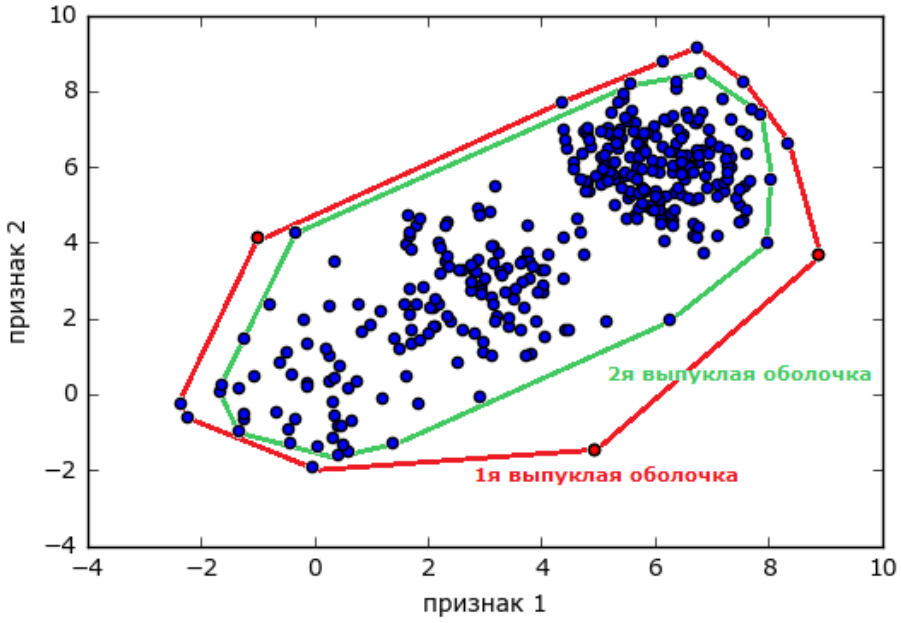

Рис. 9. Випуклі оболонки множини точок.

2.4. Метричні методи

У них постулюється існування певної метрики у просторі об'єктів, що й допомагає знайти аномалії. Інтуїтивно зрозуміло, що у викида мало сусідів, а у типової точки багато. Тому хорошою мірою аномальності може бути, наприклад «відстань до k-го сусіда» (метод Local Outlier Factor).

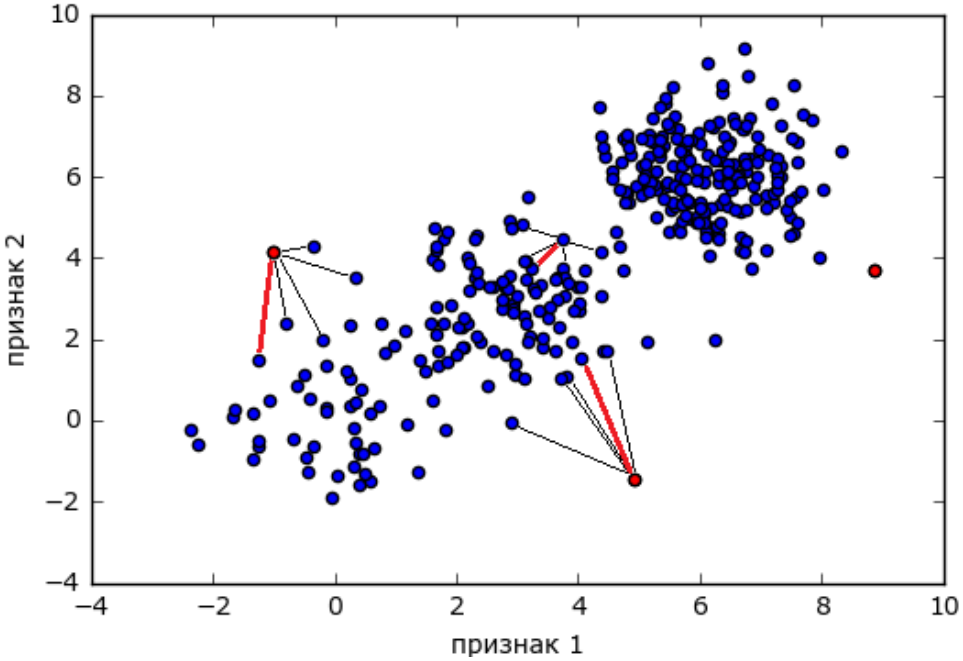

Рис. 10. Сусіди кількох елементів вибірки, зв'язок з 5-м показаний червоним

Метричні методи намагаються знайти у наборі даних такі точки, що у певному сенсі ізольовані від інших [4, 5]. Якщо у просторі задана деяка метрика *ρ(x1, x2)*, можна вводити такі подання аномальності:

• Аномалії – точки, які не потрапляють у жодний кластер. До даних застосовується один із алгоритмів кластеризації; Розмір кластера, в якому виявилася точка, оголошується її *anomaly_score*.

• Локальна щільність в аномальних точках низька. Для цієї точки *anomaly_score* визначається локальна густина, яка оцінюється деяким непараметричним способом (наприклад, ядерною оцінкою щільності Розенблата - Парзена).

• Відстань від цієї точки до найближчих сусідів велика. Як *anomaly_score* може виступати:

1. відстань до k-го найближчого сусіда;
2. середня відстань до найближчих сусідів;
3. медіана відстаней до найближчих сусідів;
4. гармонійне середнє до k найближчих сусідів;
5. частка з k найближчих сусідів, котрим дана точка не більше ніж k-им сусідом.

Використання подібних алгоритмів з параметрами (наприклад, k) накладає вимогу наявності апріорної інформації про потенційні розміри кластерів аномалій (рис. 11).

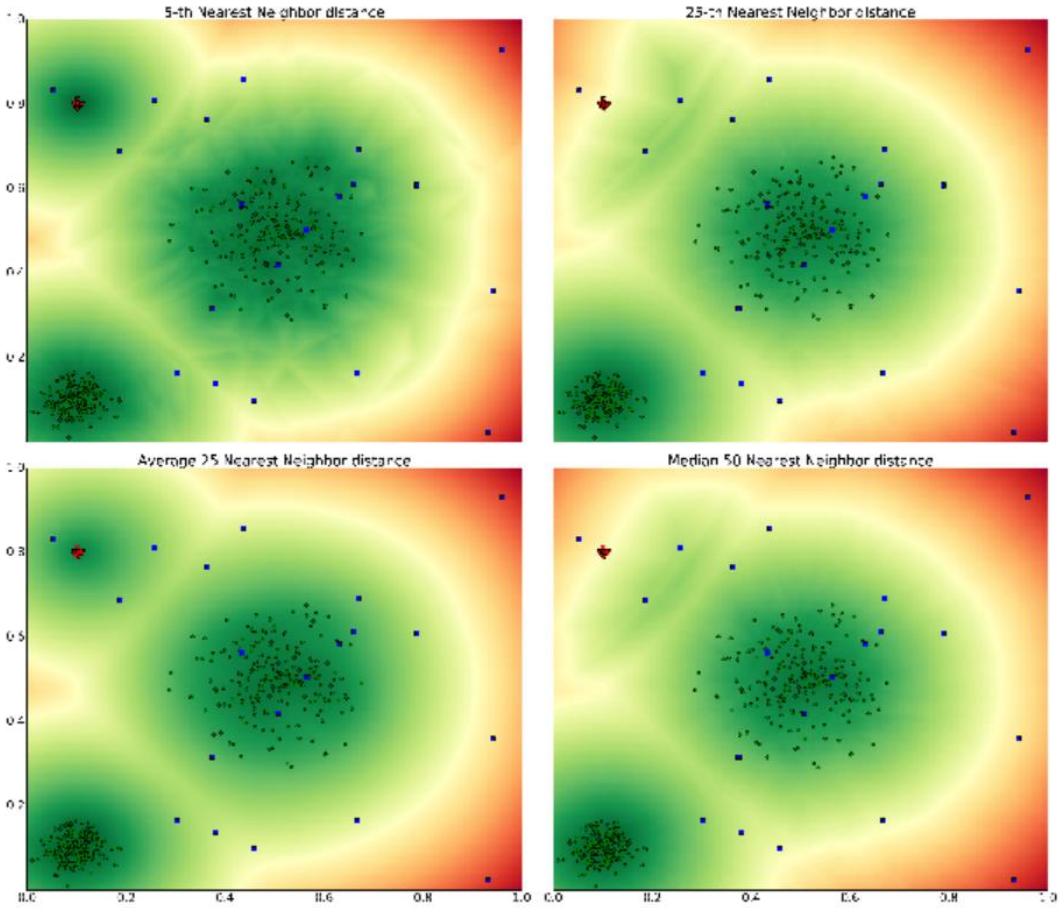

Рис 11: Приклади роботи метричних алгоритмів

Такі кластери можуть залишитися не виявленими, якщо алгоритм розглядає невелику кількість найближчих сусідів. При великому числі сусідів мінімальне значення побудованої функції може бути далеко від підпростору нормальних даних (рис. 12).

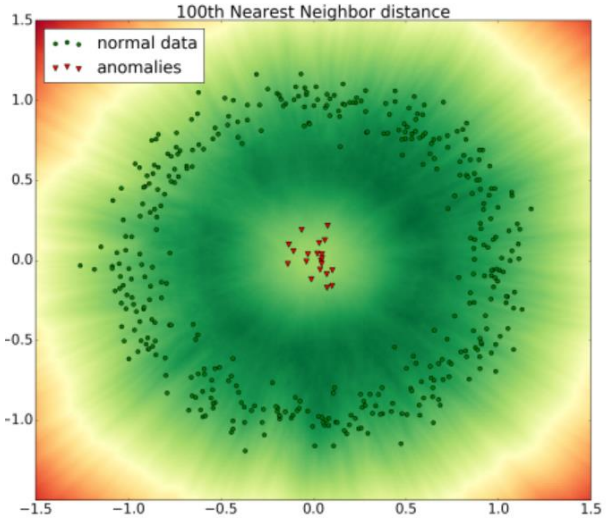

Рис. 12. Приклад завдання з нетривіальним підпростором нормальних даних, в якому аномалії розташовані в центрі мас

Головна перевага метричних методів – інтерпретованість. Можна сказати, що саме метричними методами завдання знаходження аномалій вирішує людина.

Головним же недоліком є необхідність підбору метрики. На практиці в даних інтерпретована метрика відсутня; вона апроксимується стандартними метриками – наприклад, манхетенівською, евклідовою чи чебишевською. Вибір метрики без міток може виявитися сліпим вибором, а апроксимація навіть при найкращому підборі - грубою.

ПРИКЛАД: Алгоритм LOF (Local Outlier Factor)

Більш тонкою проблемою метричних методів є той факт, що всі припущення, що лежать в їх основі, справедливі лише в доповненні один з одним: так, локальна щільність точки, що лежить в центрі невеликого кластера аномалій, може виявитися вищою, ніж для будь-якої точки з великого кластера нормальних даних. Можливо і зворотне: ізольована точка-аномалія може розташовуватися, наприклад, у центрі мас кластера нормалій, і тоді середня відстань від неї до сусідів буде меншою, ніж для нормальних точок. Ця «властивість» метричних алгоритмів намагається врахувати алгоритм LOF (Local Outlier Factor) [6].

Нехай $D_k(y)$ – відстань від точки y до k найближчого сусіда. Тоді досяжністю (reachability distance) точки x щодо точки y будемо називати величину

$$R_k(x, y) = \max(\rho(x, y), D_k(y)) \quad (5)$$

Нехай $AR_k(x)$ – середня досяжність точки x щодо своїх найближчих сусідів, $N_k(x)$ – множина k найближчих сусідів x. Тоді:

$$LOF_k(x) = \underset{y \in N_k(x)}{mean} \frac{AR_k(x)}{AR_k(y)} \quad (6)$$

Інтуїція формули (6) полягає в тому, щоб порівняти середню досяжність точки та її найближчих сусідів. Для представників нормальних даних вірно не тільки те, що оцінка (5) локальної щільності мала, а й що вона незначно відрізняється від такої ж оцінки для найближчих сусідів. Приклад роботи алгоритму наведено на рис. 10.

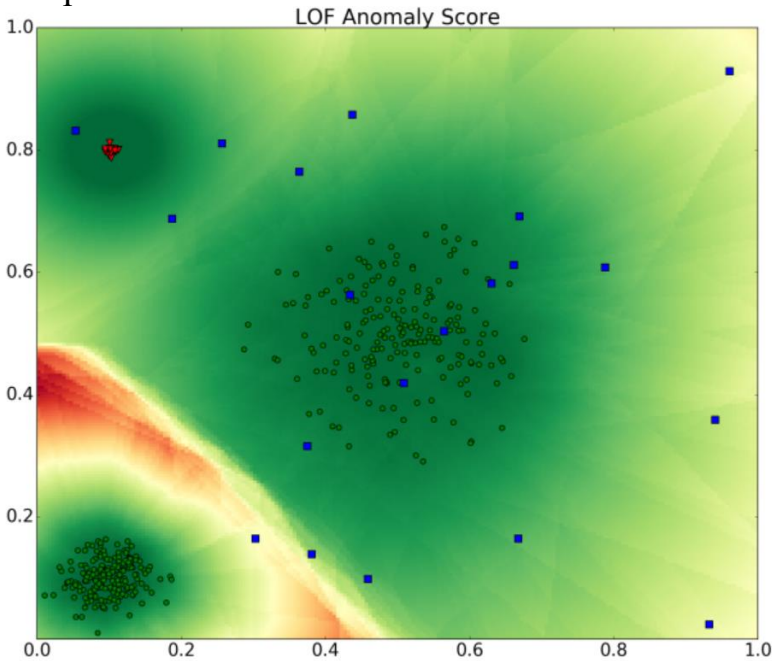

Рис. 13: Приклад роботи Local Outlier Factor

2.5. Методи підміни завдання

Коли виникає нове завдання, є велика спокуса вирішити її старими методами (орієнтованими на вже відомі завдання). Наприклад, можна зробити ***кластеризацію***, тоді маленькі кластери, швидше за все, складаються з аномалій. Якщо ми маємо часткову інформацію про аномалії (як у задачі PUC), можна вирішити її як завдання класифікації з класами 1 (розмічені аномалії) і 0 (всі інші об'єкти). Якби клас 0 складався лише з нормальних об'єктів, то таке рішення було б зовсім законним, інакше залишається сподіватися, що недетектованих аномалій у ньому небагато.

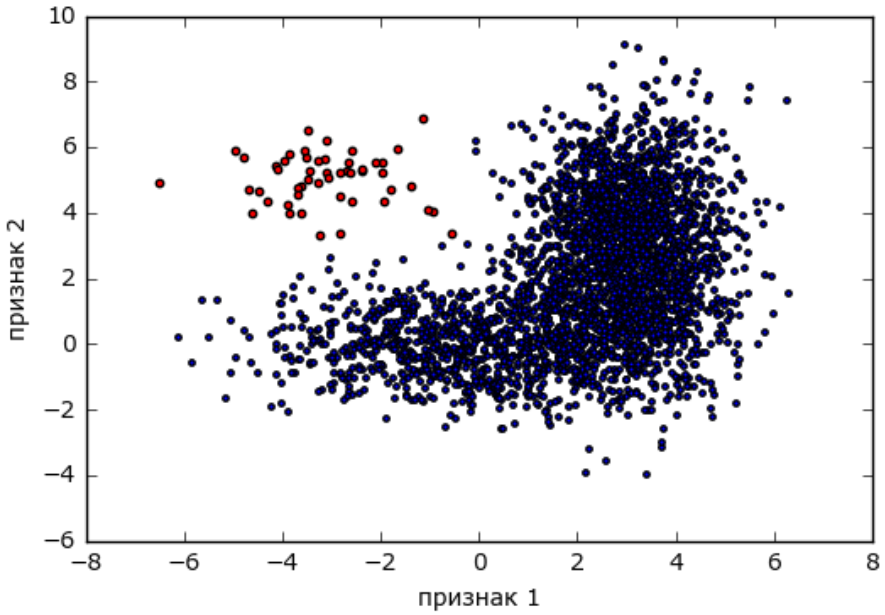

Рис. 14. Приклад кластеризації на малий (червоний) та великий (синій) кластери

2.6. Специфічні методи машинного навчання

А що якщо сприйняти завдання знаходження аномалій як нове завдання машинного навчання (відмінне від класифікації та кластеризації)?!

Найпопулярніші алгоритми (є реалізація навіть у scikit-learn) тут:

- Метод опорних векторів для одного класу (OneClassSVM)
- Ізолюючий ліс (IsolationForest)
- Еліпсоїдна апроксимація даних (EllipticEnvelope)

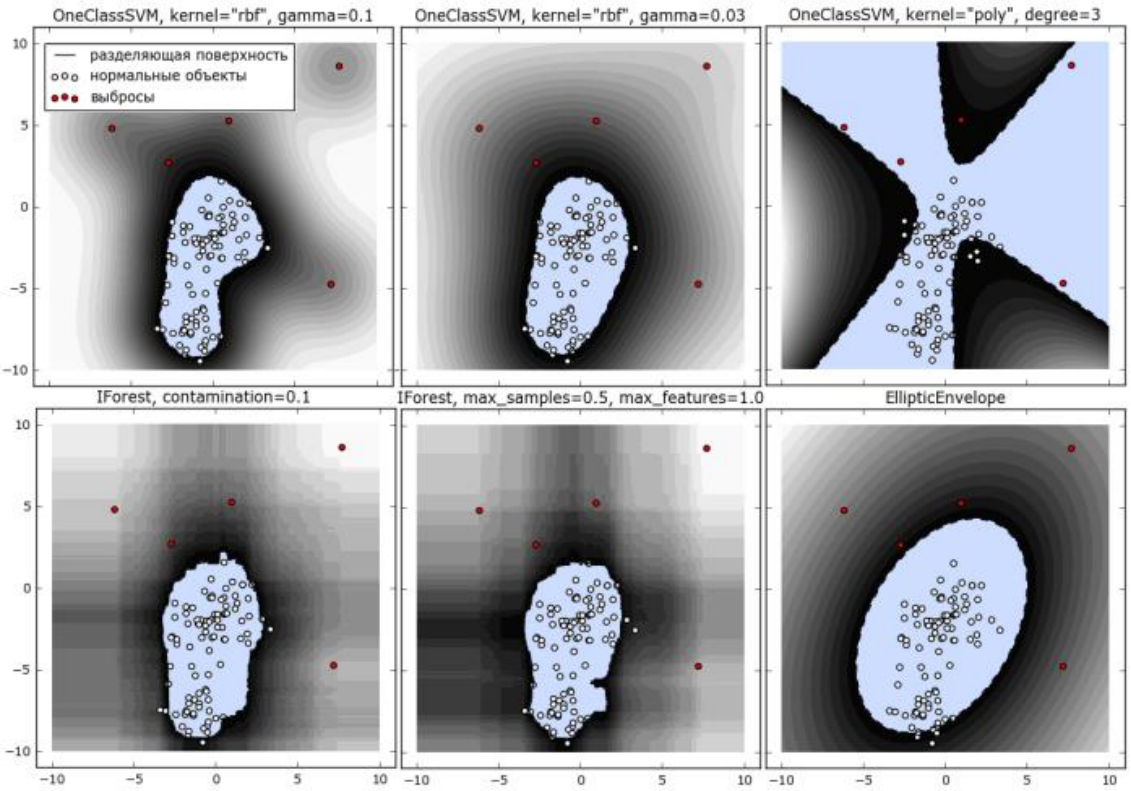

Рис. 15. Візуалізація роботи різних алгоритмів пошуку аномалій.

Перший метод - це звичайний SVM, який відокремлює вибірку від початку координат. Ідея трохи сумнівна, але виявилася досить працездатною (див. рис. 15). Тут правда не так багато різноманітності у виборі параметрів, як при вирішенні завдань класифікації, оскільки як ядро підійде лише rbf (радіальні базисні функції), решта всіх ядер показують феноменально огидний результат. Цікаво, що багато років завдання детектування поломок складних механізмів вирішувалися саме за допомогою OneClassSVM, чомусь без розгляду альтернатив. Корисно пам'ятати, що OneClassSVM це швидше алгоритм пошуку новизни, а не викидів, т.я. «заточується» під навчальну вибірку.

Isolation Forest

Ідея ізолюючого лісу (Isolation Forest) [2] заснована на принципі Монте-Карло: проводиться випадкове розбиття простору ознак, таке, що в середньому ізольовані точки відсікаються від нормальних, кластеризованих даних. Остаточний результат усереднюється з кількох запусків стохастичного алгоритму.

Алгоритм ізолюючого дерева (Isolation Tree) полягає у побудові випадкового бінарного вирішального дерева. Коренем дерева є весь простір ознак; у черговому вузлі вибирається випадкова ознака і випадковий поріг розбиття, семпльований з рівномірного розподілу на відрізку від мінімального до максимального значення обраної ознаки. Критерієм зупинки є тотожний збіг всіх об'єктів у вузлі, тобто вирішальне дерево будується повністю. Відповіддю в листку, яка також відповідає *anomaly_score* для алгоритма, оголошується глибина листка в побудованому дереві (рис.16)

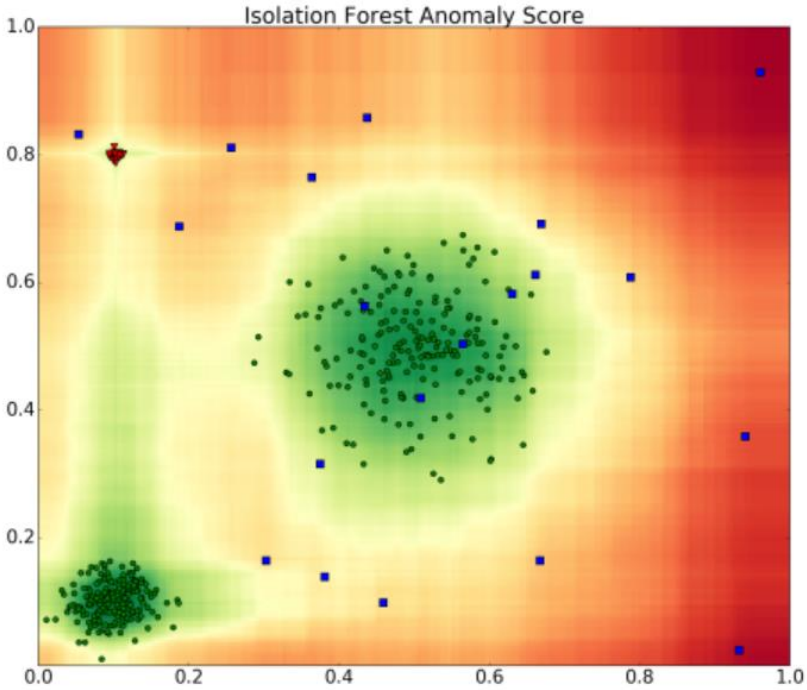

Рис. 16. Приклад роботи Isolation Fores

Стверджується, що аномальним точкам властиво опинятися у листі з низькою глибиною, тобто у листі, близьким до кореня, коли ж для розбиття гіперплощинами кластера нормальних даних дереву потрібно побудувати ще кілька рівнів. У цьому кількість таких рівнів пропорційно розміру кластера; отже, пропорційно і *anomaly_score* для точок, що лежать у ньому. Це означає, що об'єкти з кластерів малих розмірів, які потенційно є аномаліями, будуть мати *anomaly_score* нижче, ніж із кластерів нормальних даних.

Одна із найпростіших версій алгоритму працює так:

• Кожне дерево будується до вичерпання вибірки.

• Для побудови розгалуження у дереві: вибирається випадкова ознака та випадкове розщеплення.

• Для кожного об'єкта обчислюється міра його нормальності – середнє арифметичне глибин листя, в яке він потрапив (ізолювався).

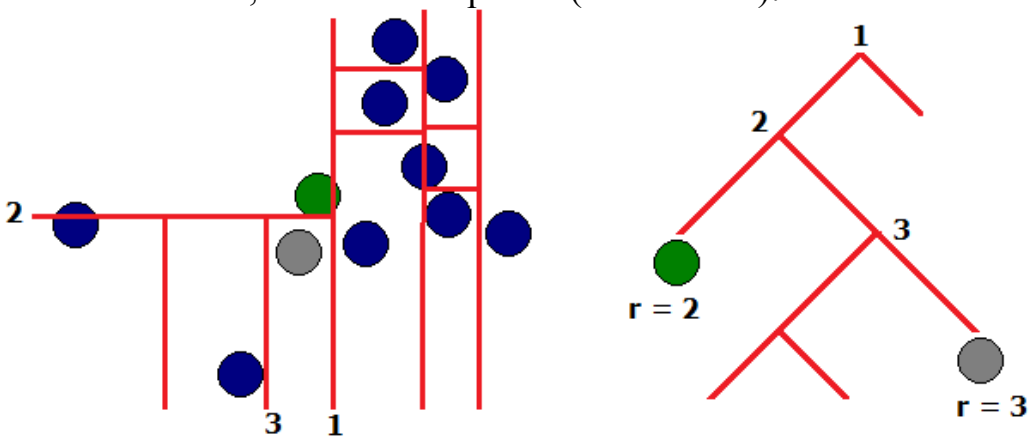

Рис. 17. Обчислення оцінки аномальності в ізолюючому лісі

Логіка алгоритму проста: при описаному «випадковому» способі побудови дерев викиди потраплятимуть у листя на ранніх етапах (на невеликій глибині дерева). Тобто, викиди простіше «ізолювати» (нагадаємо, що дерево будується доти, доки кожен об'єкт не опиниться в окремому листку). Алгоритм добре відловлює саме викиди (рис. 18).

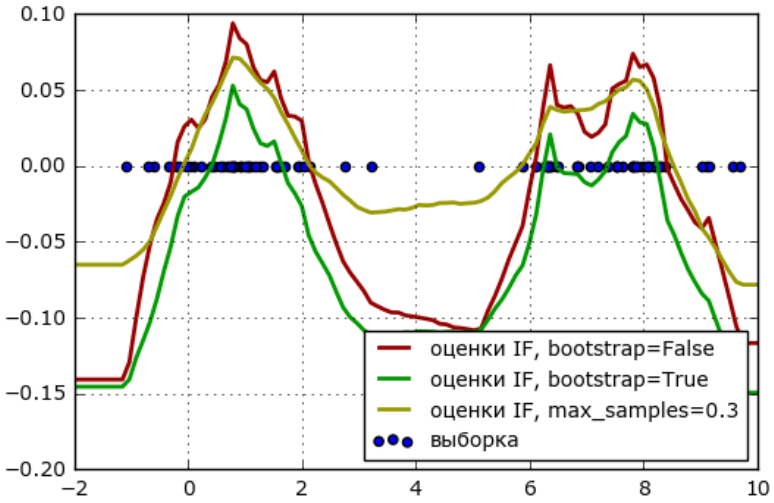

Рис. 18. Оцінки ізолюючого лісу для одновимірної вибірки

Даний алгоритм має ряд істотних переваг:

• Алгоритм розпізнає аномалії різних видів: як ізольовані точки з низькою локальною щільністю,так і кластери аномалій малих розмірів.

• Складність ізолюючого дерева – $O(n \log n)$, що ефективніше за більшість інших алгоритмів.

• Не вимагає суттєвих витрат по пам'яті, на відміну від, наприклад, метричних методів, які часто потребують побудови матриці попарних відстаней.

• Відсутні параметри, які потребують вибору.

• Інваріантний до масштабування ознак; не вимагає завдання метрики або іншої апріорної інформації про будову даних.

• Стійкий до прокляття розмірності.

3 Методи покращення алгоритмів

3.1 Семплювання

Більшість алгоритмів розпізнавання аномалій успішно працюють на вибірках малих розмірів. Ідея семплювання полягає в тому, щоб запустити алгоритм на кількох випадкових підвибірках та усереднити результат.

Розмір випадкових підвибірок може бути як фіксований, так і випадковий з деякого діапазону, але найчастіше він на порядки менше розміру вихідної вибірки. Інтуїція такого вибору полягає в тому, що шумові об'єкти потраплять у вибірку з низькою ймовірністю; кластери нормальних даних будуть представлені кількома представниками, а кластери аномалій виродяться в ізольовані точки (рис. 19). Зазвичай на таких підвибірках алгоритми будують функцію *anomaly_score(x)*, що не сильно поступається результату, отриманому за всіма

вихідними даними, навіть незважаючи на те, що у підвибірках може загубитися глобальна структура простору ознак. Усереднення по невеликій кількості запусків, зазвичай близько 100, на практиці вистачає для вирішення цієї проблеми.

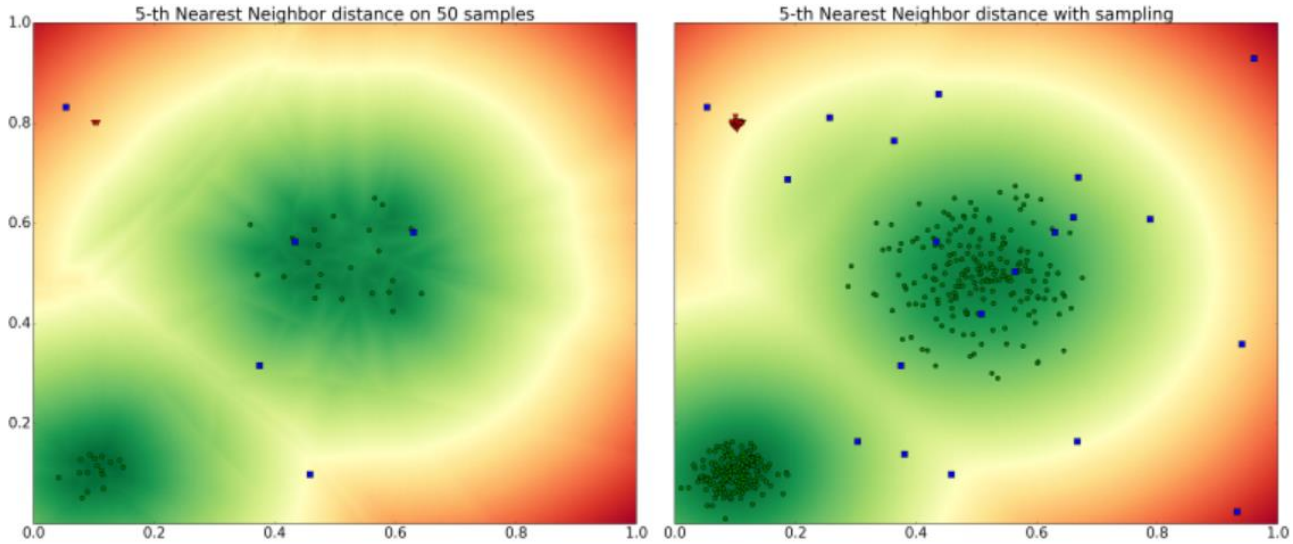

Рис. 19. Відстань до 5-го сусіда на випадковій підвиборці з 50 елементів (ліворуч) та результат усереднення по 100 випадковим підвибіркам (праворуч)

Крім значної оптимізації обчислювальної складності, семплювання зменшує ймовірність «підгону» результату алгоритму під конкретні наявні дані. З огляду на особливості завдання, необхідна умова відсутності параметризації алгоритмів часто означає їх детермінованість (без стохастичності *anomaly_score* однозначно визначається за заданою вибіркою). Це важливо у тому випадку, якщо передбачається використовувати отриманий *anomaly_score* на нових даних. У випадку, з появою таких нових даних, будь-який алгоритм навчання без вчителя можна запустити наново на об'єднанні всієї наявної інформації; у разі, якщо використовується семплювання, можна не перенавчати алгоритм повністю, а додати запуски в ансамбль (т. зв. warm start).

Для ізолюючого лісу семплювання безпосередньо «вбудовано» в сам алгоритм: оскільки сам алгоритм є ансамблем з кількох ізолюючих дерев, кожне дерево будується за підвибором з *max_samples* елементів. На практиці *max_samples* вважають рівним 250, оскільки стохастична природа алгоритму робить налаштування цього параметра безглуздим. Таким чином, для великих вибірок алгоритм не використовує всі наявні дані, що збільшує його узагальнюючу здатність. Хороша працездатність алгоритму на малих вибірках обумовлена тим, що у кожному вузлі вирішального дерева використовується лише інформація про мінімальне і максимальне значення ознаки; а ось використання великої кількості точок для побудови одного дерева може призвести до занадто об'ємного дерева, що несе надмірну і, часто, марну інформацію.

3.2 Випадкове масштабування

Алгоритми детектування аномалій часто не інваріантні щодо лінійних перетворень простору. Це призводить до залежності результату від ознакового уявлення об'єктів.

Так, однією з важливих переваг ізолюючого лісу є інваріантність до масштабування ознак. Наприклад, для метричних алгоритмів суттєвим є нормалізація ознак (приведення значень до діапазону [−1, 1] або центрування та поділ на дисперсію). Однак таке стандартне нормування призводить до втрати потенційної інформації про значущість ознак. Часто погані результати роботи метричного алгоритму пов'язані саме з вибором стандартної метрики на нормованому просторі, що не відповідає фізичному змісту ознак (яка, у свою чергу, зазвичай прихована) [1].

За наявності міток цю проблему можна вирішувати параметризацією метрики та подальшим підбором параметрів; проте, за їх відсутності це неможливо. Потенційним рішенням виступає таке міркування: якщо використання стандартно нормалізованих ознак необґрунтовано, можна взяти кілька різних необґрунтованих нормалізацій і усереднити їх. Іншими словами, для j-го алгоритму в ансамблі використовується перетворена вибірка:

$$w(j) \sim Uniform[0,1]^d$$
$$\forall j : \widetilde{X}(j)_i = w(j) X_i, \quad (7)$$

де X_i - *i*-а компонента вихідної, попередньо нормалізованої стандартним чином, вибірки.

Це досить загальний спосіб боротьби з проблемою невідомої метрики, проте гарантій того, що результат виявиться кращим, не надає. Більше того, немає способу порівняти результат, отриманий із випадковими масштабуваннями, і без.

3.3 Rotated Bagging

На відміну від масштабування, ізолюючий ліс не інваріантний щодо поворотів простору ознак. Так, при побудові ліній рівня *anomaly_score* ізолюючого лісу на модельних даних (d = 2) можна виявити (див. рис. 16), що відходячи від кластерів нормальних даних уздовж осей координат, значення *anomaly_score* змінюється незначно. Потенційні аномалії, що потрапляють в цю область, можуть бути сплутані з граничними об'єктами кластера, незважаючи на те, що вони виражено ізольовані. Причина полягає в тому, що всі поділи простору ізолюючий ліс проводить гіперплощинами, паралельними осям координат.

Загальна ідея Rotated Bagging полягає у виборі випадкових підпросторів, на які проектується вибірка чергового алгоритму в ансамблі. Таким чином, для j-го алгоритму вибирається $Q(j) \in R^{d' \times d}$ – довільна матриця з нормованих попарно-ортогональних векторів, після чого перетворена вибірка виходить за формулою

$$\forall j : \widetilde{X}(j) = XQ(j)^T \quad (8)$$

Зокрема, вважаючи $d' = d$, отримуємо довільне ортогональне перетворення простору ознак. Для ізолюючого лісу це рівносильно використанню гіперплощин, паралельних осям випадково обраного ортогонального базису; усереднення дозволяє отримати згладжені лінії рівня (рис. 20).

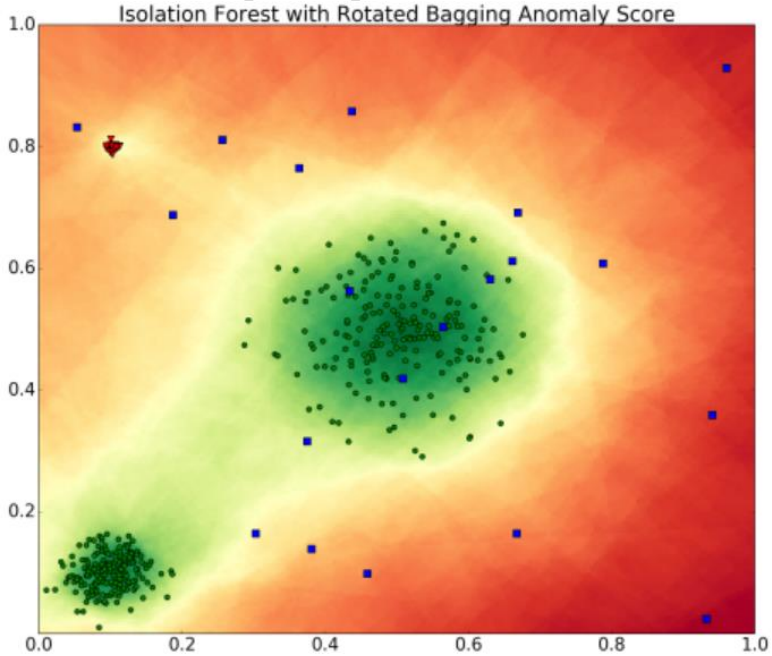

Рис. 20. Результат роботи ізолюючого лісу з використанням Rotated Bagging

Недоліком такої модифікації є втрата інваріантності масштабування ознак. Оскільки в модифікованому просторі кожна ознака є лінійною комбінацією вихідних, домінування значень одного з них за модулем призведе до підвищеної значущості цієї ознаки.

Тому використання цього методу для ізолюючого лісу вимагає попереднього нормування і потенційно може призвести до тих же проблем, що й використання метричних алгоритмів.

Теоретично Rotated Bagging, у тому числі і у випадку $d' = d$, може бути застосований і для інших алгоритмів, не інваріантних щодо поворотів простору, у тому числі й для метричних методів (при використанні неевклідової метрики).

3.4 Ансамблювання

У більш загальному вигляді, ансамблювання в задачі аномалій може бути задано як використання кількох різних алгоритмів з подальшим усередненням їх *anomaly_score*. Однак, для різних алгоритмів видається ними *anomaly_score* має різні шкали та масштаби. Тому традиційне приведення значень функцій різних алгоритмів одного діапазону (наприклад, до [0, 1]) з наступним усередненням позбавлено своїх звичайних переваг.

На практиці більшого успіху можна досягти ансамблюванням одного і того ж алгоритму на модифікованих вибірках, наприклад, за допомогою семплювання, традиційного бегінга, випадкового масштабування або Rotated Bagging. Поєднання ж результатів різнорідних алгоритмів, заснованих на несхожих

принципах і підходах, краще проводити не усереднення їх *anomaly_score*, а голосуванням за кінцевою відповіддю, чи є точка аномальною чи ні. Для цього для кожного з видів алгоритмів слід спочатку індивідуально визначити поріг бінаризації.

3.5 Ітеративний відбір

Ітеративний відбір в задачі детектування аномалій є особливим методом ансамблювання кількох алгоритмів. [1]

Припустимо, побудовано деяку модель, яка описує нормальні дані. Ця модель побудована на всіх наявних даних, що містять аномалії, і тому її точність може бути невеликою. Однак її наближення достатньо, щоб виявити явні аномалії. Відсортувавши всі точки по *anomaly_score*, можна вибрати найаномальніші об'єкти в даних і виключити їх з даних. Потім модель будується заново, причому точність її наближення буде вищою. Цей процес можна повторити кілька разів або ввести будь-які критерії зупинки (Рис. 21).

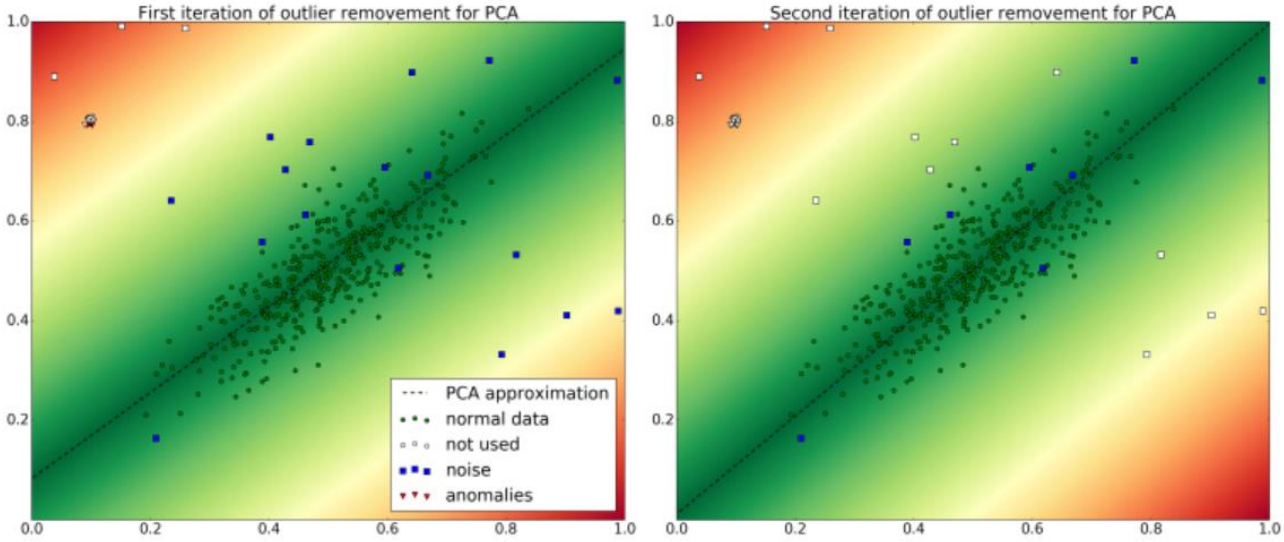

Рис. 21. Ітеративна побудова апроксимуючого одновимірного підпростору. На кожному кроці видаляється 15 аномальних об'єктів, після чого модель перебудовується

Ідея ітеративного відбору може бути узагальнена у різний спосіб. Результат роботи одного алгоритму може бути використаний для відсіювання явних аномалій і налаштування нового алгоритму, що не обов'язково збігається з попереднім, на даних, що залишилися. Можлива і протилежна механіка: за результатами роботи одного алгоритму відбираються явні, гарантовані представники нормальних даних, і виключно на них будується модель, що їх описує (рис. 15).

Важливо, що одні алгоритми працюють тим краще, ніж менше аномалій у даних (побудова моделей), коли інші припускають наявність у даних аномалій (наприклад, ізолюючий ліс). Якщо останнім подати на вхід лише нормальні дані, побудована ними функція *anomaly_score* зазвичай не зможе розрізняти граничні нормальні дані з аномаліями, у тому числі явними.

Тому ітеративний відбір слід організовувати так, щоб перші алгоритми відсіювали явні аномалії та/або відбирали представників нормальних (Isolation Forest, LOF), а фінальні алгоритми були модельними, тобто будують функцію, виходячи з припущення про нормальність вхідної інформації (наприклад, лінійні моделі, EM -Алгоритм, поліноміальна функція).

Більш специфічним інструментом є можливість вбудовування результатів роботи одного алгоритму "всередину" інших. Наприклад, можна проводити побудову моделі за всіма наявними даними з вагами, обернено пропорційними їх рейтингу аномальності, виданим деяким іншим алгоритмом.

ІСНУЄ І БАГАТО ІНШИХ ПОПУЛЯРНИХ АЛГОРИТМІВ ЯКІ ЗАСТОСОВУЮТЬСЯ ДЛЯ РІШЕННЯ ЗАДАЧІ ПОШУКУ АНОМАЛІЙ!

Реплікаторні нейронні мережі, [24] автоенкодери, варіаційні автоенкодери, [25] нейронні мережі з довготривалою короткочасною пам'яттю [26]

Байєсівські мережі [24]

Приховані марківські моделі (HMM) [24]

Детермінант мінімальної коваріації [27] [28]

Кластеризація: виявлення викидів на основі кластерного аналізу [29] [30]

Відхилення від правил асоціації та часті набори елементів

Виявлення викидів на основі нечіткої логіки

Методи ансамблю, що використовують пакетування ознак, [31] [32], нормалізацію оцінок [33] [34] та різні джерела різноманітності [35] [36]

Існують і набори інструментів для виявлення аномалій

ELKI — це набір інструментів для інтелектуального аналізу даних Java із відкритим кодом, який містить декілька алгоритмів виявлення аномалій, а також прискорення індексування для них.

PyOD — це бібліотека Python з відкритим кодом, розроблена спеціально для виявлення аномалій.[39]

scikit-learn — це бібліотека Python з відкритим вихідним кодом, яка має вбудовані функції для забезпечення неконтрольованого виявлення аномалій.

Wolfram Mathematica забезпечує функціональність для неконтрольованого виявлення аномалій у різних типах даних

Висновки

Вибір методу для детектування аномалій залежить насамперед від поставленого завдання, даних та наявної апріорної інформації. Розглянуті підходи є лише математичними моделями поняття аномальності і відштовхуються від інтерпретації завдання.

За відсутності апріорної інформації при необхідності побудувати модель, що описує нормальні дані, варто використовувати поліноміальну або гармонійну функцію. Використання семплювання дозволяє узагальнити ці алгоритми на випадок, якщо в даних є невелика частка аномалій, причому найчастіше така модифікація працює краще за ітеративні методи.

Найбільш інтерпретованими алгоритмами є метричні, для яких необхідний правильний підбір метрики та нормування ознак. Якщо остання процедура потенційно може призвести до втрати інформації, може статися випадкове масштабування. Семплювання для цього виду алгоритмів призводить як до зростання якості, так і знижує обчислювальну складність.

Найбільш стабільним і загальним алгоритмом вважається Isolation Forest, який не вимагає ніякої апріорної інформації. При цьому, будь-які модифікації, що «метрично виправляють» простір ознак, для нього не потрібні і частіше всього призводять до втрати якості.

Список литературы

[1] Aggarwal, Charu C. Outlier Analysis // - 2017. -C. 1-247)

[2] Liu F. T., Ting K. M., Zhou Z. H. Isolation Forest // Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on. – IEEE, 2008. – С. 413-422.

[3] He Z. et al. Fp-outlier: Frequent pattern based outlier detection // Computer Science and Information Systems. – 2005. – Т. 2. –№. 1. – С. 103-118.

[4] Hodge V., Austin J. A survey of outlier detection methodologies // Artificial intelligence review. – 2004. – Т. 22. –№. 2. – С. 85-126.

[11] Asuncion A., Newman D. UCI machine learning repository. – 2007.

URL: http://archive.ics.uci.edu/ml (Апрель, 2017)

[12] Shebuti Rayana. ODDS Library. Stony Brook, - 2016. NY: Stony Brook University, Department of Computer Science. URL: http://odds.cs.stonybrook.edu (Апрель, 2017)

[13] Pedregosa F. et al. Scikit-learn: Machine learning in Python // Journal of Machine Learning Research. – 2011. – Т. 12. –№. Oct. – С. 2825-2830.

URL: http://scikit-learn.org/stable/index.html (Апрель, 2017)

[14] Репозиторий с материалами и экспериментами:

URL: https://github.com/FortsAndMills/AnomalyDetectionMethods