Министерство общего и профессионального образования
Российской Федерации
Московский Физико-технический институт
(Государственный университет)
Факультет управления и прикладной математики
Кафедра Acronis

Выпускная квалификационная работа "Поиск аномалий во временных рядах системы Acronis Storage"

Студента 4-го курса Угнивенко Виталия Алексеевича

Научный руководитель
Андрей Кулага

Москва, 2020

Аннотация

Данная работа посвящена проблеме детектирования аномалий во временных рядах, которая заключается в идентификации и обработке аномальных участков в потоках данных, получаемых из системы Acronis Storage. В результате работы была создана система поиска аномалий, главными преимуществами которой являются высокая точность и корректная работа на всём многообразии хранилищ данных компании Acronis.

Выявление аномального поведения в автоматическом режиме позволит повысить качество работы системы Acronis Storage, а также поможет предотвращать нештатные ситуации на этапах их зарождения. В этом и заключается актуальность проведения работы в данной области.

Содержание

1 Введение

Реалии современного мира вынуждают IT компании создавать всё более функциональные, но при этом и более технически сложные продукты. Очевидно, что с ростом глубины системы, уменьшается возможность прокрыть и обработать все возможные ситуации, потенциально ведущие к отклонению поведения данной системы от нормального.

В свою очередь любые сбои в работе сложных технологических систем могут привести к существенным потерям, как со стороны компании, так и со стороны пользователей. Вот почему задача детектирования сбоев работы и анализа состояния системы в реальном времени всегда останется актуальной.

Данная проблема касается и Acronis Storage, ведь это один из ключевых продуктов компании Acronis. Миллионы клиентов предпочли его для хранения, а также восстановления данных. Для предоставления своих услуг в этой области компания Acronis имеет десятки хранилищ данных.

Любые непредвиденные происшествия со стороны хранилищ данных могут привести к серьёзным проблемам, таким как выход из строя сервисов компании или потеря пользовательских данных, поэтому задача мониторинга работы систем и своевременного предупреждения сбоев в их работе становится крайне важной.

Учитывая количество информации, поступающей из хранилищ данных, остро встаёт вопрос об автоматизации этих процессов. Стандартными подходами к проблеме такого рода являются методы машинного обучения и анализ данных. Эти области современной науки позволяют нам построить автоматическую систему поиска аномалий в работе хранилищ данных, способную выявлять сложные шаблоны сбоев на ранней стадии их появления, тем самым заметно снижая сопутствующий ущерб.

Несмотря на то, что большое количество исследований на эту тему уже было проведено, финальный алгоритм должен учитывать интересующие нас систему и данные.

1.1 План работы

Данная работа организована следующим образом

- В **Главе 2** описывается, с данными какого рода мы имеем дело и что именно мы в этих данных хотим отыскать.
- В **Главе 3** рассматривается, какая часть всех данных была выбрана для построения модели, а также причины этого выбора.
- В **Главе 4** обсуждаются используемые методы предобработки данных, необходимые для создания качественной системы детектирования аномального поведения.
- В **Главе 5** приводится описание предиктивной модели.
- В **Главе 6** описывается алгоритм распознавания аномалий во временных рядах.
- В **Главе 7** рассматриваются результаты поиска.
- В **Главе 8** приводятся примеры найденных аномалий.
- В **Главе 9** доказывается корректность предложенного алгоритма.

2 Постановка задачи

2.1 Данные

Информацию о функционировании хранилищ данных компании Acronis мы получаем, анализируя временные ряды метрик системы Acronis Storage, полученные нами при помощи системы Promoteus [1].

Временной ряд $X = \{x(t)|1 \leqslant t \leqslant m\}$ - это последовательность векторов-наблюдений $x(t) = (x_1(t), x_2(t), \dots x_d(t)) \in \mathbb{R}^d$ упорядоченных во времени. Чаще всего эти наблюдения собираются через равные промежутки времени. Эти ряды называются одноменрными временными рядами, если $d = 1$, и, соответственно, многомерными, если $d \geqslant 2$.

Есть две основные цели анализа временных рядов:

1. `Изучение природы данных`
2. `Прогнозирование - предсказание его будущих значений по прошлым или настоящим значениям, где время является независимой переменной`

Основное предположение в анализе временных рядов состоит в том, что некоторые шаблоны из прошлого будут появляться и в будущем. Как следствие, информация, полученная из анализа временных рядов может быть применена для прогнозирования, поиска аномалий и других приложений [2].

2.2 Понятие аномальности

Аномалия - это любое отклонение от стандартного поведения системы. На сегодняшний день существует большое количество работ по детектированию аномалий [3], в большей части которых используются методы идентификации отдельных объектов, которые отличаются от типичных объектов, но не учитывается аспект последовательности данных. Строго говоря аномалии подразделяют на 3 основных типа:

Точечная аномалия - единичный экземпляр данных рассматривается как аномальный по отношению к остальным. На рис.1 экземпляры $A1$ и $A2$ являются аномальными по отношению к экземплярам $C1$ и $C2$.

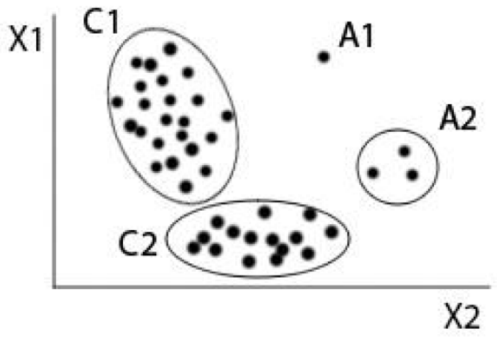

Рис. 1: Пример точечных аномалий

Контекстуальная аномалия - экземпляр данных рассматривается как аномальный в контексте. Для детектирования данных аномалий необходимы контекстуальные и поведенческие признаки. Во временных рядах контекстуальный признак - время. А поведенческие признаки - любые признаки, которые описывают экземпляр данных без оглядки на контекст. Таким образом отклонение от стандартного поведения системы определяется поведенческими признаками в контексте, определяемым контекстуальными признаками. Например на рис.2 точка A - аномалия, при этом точки $N_1, \ldots, N_5$ - таковыми не являются.

Коллективные аномалии - последовательность элементов является аномальной по отношению ко всему набору данных. То есть один элемент последовательности может являться аномалией, но одновременное появление таких элементов считается коллективной аномалией.

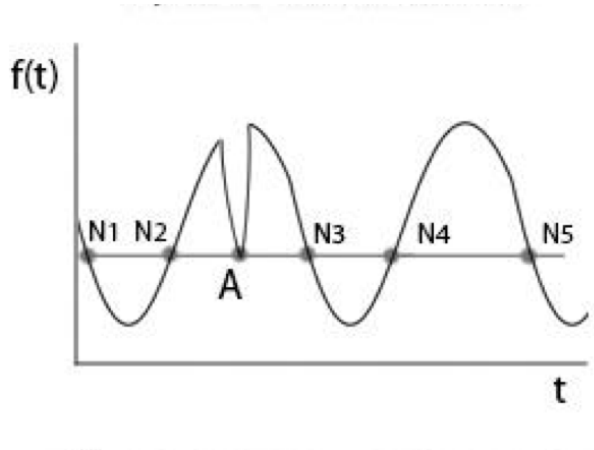

Рис. 2: Пример контекстуальной аномалии

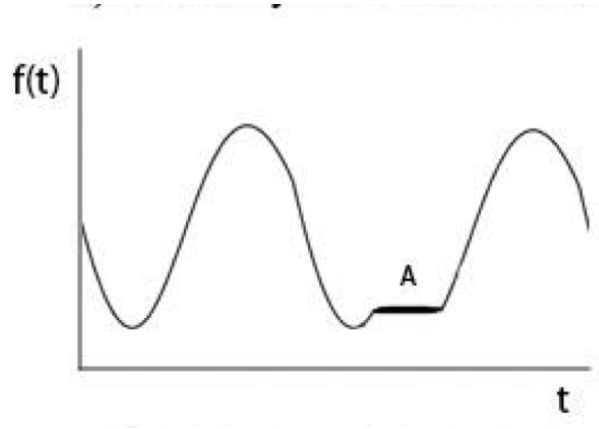

Рис. 3: Пример коллективных аномалий

3 Отбор данных

Чтобы покрыть максимальное количество шаблонов поведения системы с учётом высокой трудоёмкости вычислений, были использованы данные за полгода со следующих хранилищ:

1. `au2-acs1`
2. `us6-acs2`
3. `eu3-acs1`
4. `us3`

Детальный анализ временных рядов, описывающих работу системы, привёл к выводу о необходимости анализа временных рядов abgw, ввиду их интерперитуемости и широкого использования всеми хранилищами данных. В качестве целевой переменной была выбрана следующая метрика:

```
sum(abgw_req_latency_ms_sum{req ∈ {"OpenFile", "Append"}})
```

Выбор обусловлен тем, чем смысл данной метрики - суммарная задержка исполнения соответствующих операций. По мнению экспертов, аномальное поведение системы зачастую сопровождается скачком задержки исполнения данных операций. Обзор остальных метрик привел нас к выбору наиболее информативных:

1. `sum(abgw_req_latency_ms_count{req = "OpenFile"+ "Append"})`
2. `abgw_iop_latency_ms_count_open(pcs)`
3. `abgw_iop_latency_ms_sum_open(pcs)`
4. `abgw_conns`
5. `abgw_account_lookup_errs_total{err="OK"}`
6. `abgw_account_pull_errs_total{err="OK"}`
7. `abgw_accounts`

8. `abgw_append_throttle_delay_ms_total`
9. `abgw_fds`
10. `abgw_file_lookup_errs_total{err="OK"}`
11. `abgw_files`
12. `abgw_read_bufs`
13. `abgw_read_bufs_bytes`
14. `abgw_read_bytes_total{proxied="0"}`
15. `abgw_read_reqs_total`

4 Предобработка данных

Основные составляющие любого временного ряда

- Тренд - это плавное долгосрочное изменение уровня ряда в плане возрастания или убывания значений.
- Сезонность - циклические изменения уровня ряда с постоянным периодом.
- Цикл - изменение уровня ряда с переменным периодом.
- Ошибка - непрогнозируемая случайная компонента ряда. В идеале - нормальная центрированная случайная величина.

Стационарность – постоянство характеристик процесса со временем, таким образом стационарный временной ряд лишён сезонной и трендовой составляющей. По теореме Вольда любой стационарный ряд может быть описан моделью ARMA(p, q) с любой точностью. Именно поэтому остро стоит вопрос о поиске универсального метода предобработки временных рядов для всего многообразия хранилищ данных системы Acronis Storage.

4.1 Тренд

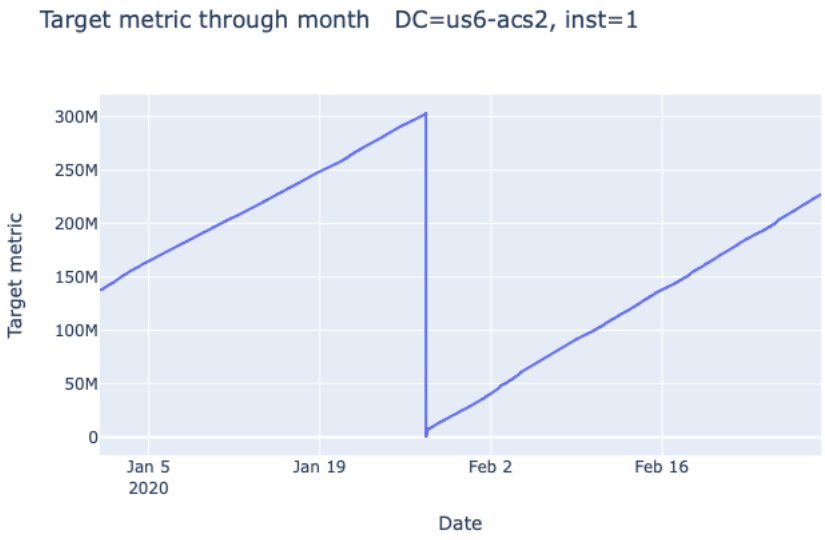

Рис. 4: Целевая метрика за исследуемый период

Если не брать во внимание резкий скачок целевой переменной вниз, обусловленный перезагрузкой системы, то легко заметить, что у нашей целевой переменной есть тренд к возрастанию, от которого можно избавиться путём перехода к приращениям первого порядка. Перейдём в пространство производных и уберём данные, соответствующие перезагрузке.

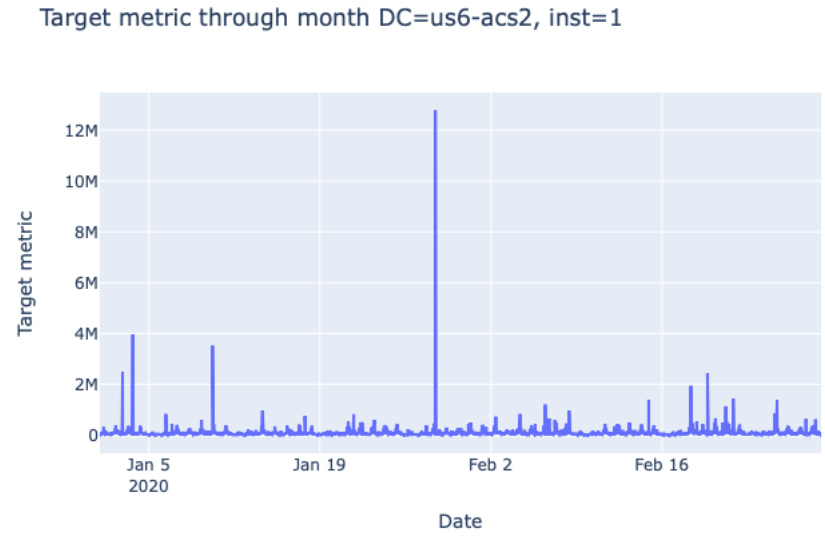

Рис. 5: Целевая метрика в пространстве производных за исследуемый период

У полученной переменной существует простая интерпретация - прира-

щение суммарной задержки, другими словами задержка, накопленная за одну минуту.

4.2 Сезонность

Если посмотреть на Рис. 6 (часть Рис. 5), то можно заметить, что нашему временному ряду присуща дневная сезонность. Но дискретность данных высока, поэтому избавиться от сезонности с помощью сезонного дифференцирования не удаётся.

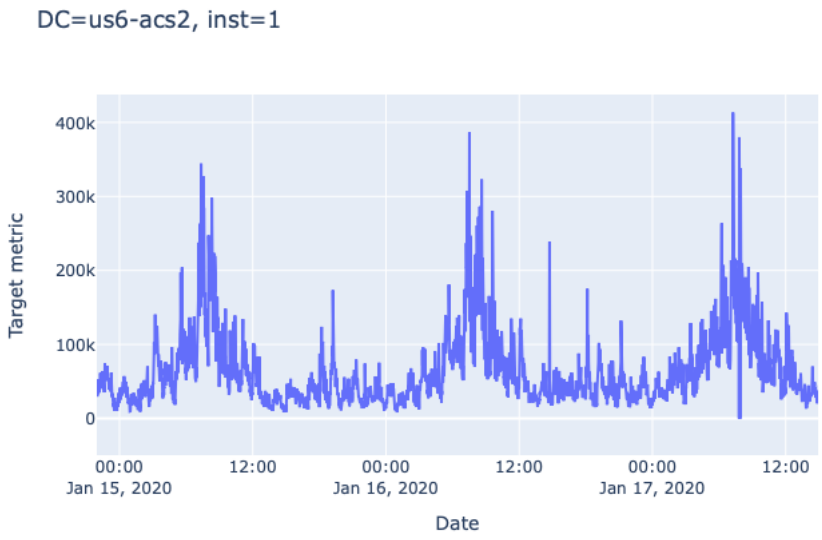

Рис. 6: Целевая метрика в пространстве производных

4.3 Масштабирование

Необходимость в масштабировании заключена в следующем: градиентный спуск прекрасно работает, если линии уровня функции походят на окружности, так как какая бы не была начальная точка, вектор антиградиента будет направлен точно в сторону минимума функции, а итеративный процесс будет сходиться быстро. А если линии уровня функции имеют эллипсоидный вид, другими словами признаки имеют разный масштаб, то направление вектора антиградиента будет слабо совпадать с направлением в сторону минимума функции. Как следствие, градиентный спуск будет делать много шагов и сходимость итеративного процесса будет медленная. Более того, существует риск расхождения процесса при неверно подобран-

ном размере шага.

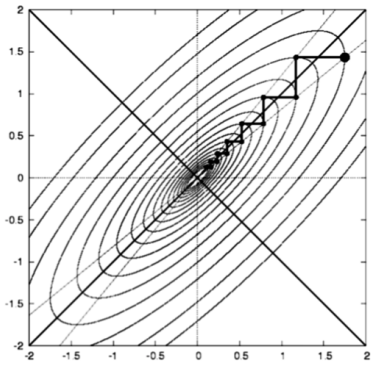

Рис. 7: Градиентный спуск без масштабирования

Используем масштабирование на отрезок $[0, 1]$. Для этого необходимо вычислить максимальное и минимальное значение каждого признака на обучающей выборке:

$$m_j = min(x_1^j, \ldots, x_l^j)$$

$$M_j = max(x_1^j, \ldots, x_l^j)$$

Тогда значение каждого признака на конкретном объекте преобразовывается следующим образом:

$$x_i^j := \frac{x_i^j - m_j}{M_j - m_j}$$

4.4 Удаление участков с перезагрузками

Во время перезагрузок системы целевая переменная в пространстве производных принимает большие по модулю отрицательные значения. Перезагрузка - это не стабильное состояние, обучаться на таких интервалах нет смысла, поэтому мы убираем их из обучающей выборки.

5 Модель прогнозирования временного ряда

5.1 Введение в линейную регрессию

Линейный алгоритм в задаче регрессии выглядит следующим образом:

$$a(x) = w_0 + \sum_{j=1}^{d} w_j \cdot x^j,$$

где w_0 — свободный коэффициент, x^j - признаки, описывающие объект x, а w_j — соответствующие веса. Если добавить $(d+1)-$ый признак, принимающий на каждом объекте значение 1, линейную алгоритм может быть переписан в следующем виде:

$$a(x) = \sum_{j=1}^{d+1} w_j \cdot x^j = (w, x),$$

где (w, x) - скалярное произведение векторов. В качестве меры ошибки можно выбрать отклонения от предсказания: $|a(x) - y|$. Но данная функция не является гладкой, следовательно мы не сможем легко использовать градиентные методы для оптимизации данного функционала. Лучше в качестве меры ошибки выбрать квадрат отклонения от прогноза: $(a(x) - y)^2$

А функционал ошибки, он же среднеквадратичная ошибка алгоритма, он же MSE, задаётся следующим образом:

$$Q(a(x), X) = \frac{1}{l} \cdot \sum_{i=1}^{l} (a(x_i) - y_i)^2$$

А так как функционал зависит только от вектора весов w, мы можем переписать в следующем виде:

$$Q(w, X) = \frac{1}{l} \cdot \sum_{i=1}^{l} ((w, x_i) - y_i)^2$$

Обучение модели линейной регрессии состоит в том, чтобы минимизиро-

вать функционал ошибки:

$$Q(w, X) = \frac{1}{l} \cdot \sum_{i=1}^{l}((w, x_i) - y_i)^2 \longrightarrow \min_w$$

Перепишем в матричной форме. Пусть $y = (y_1, \ldots, y_l)^T$- вектор ответов. А матрица X состоит из описаний объектов из обучающей выборки:

$$X = \begin{pmatrix} x_{11} \ldots x_{1d} \\ \ldots\ldots\ldots \\ x_{l1} \ldots x_{ld} \end{pmatrix}$$

Тогда задачу обучения можно записать следующим образом:

$$Q(w, X) = \frac{1}{l}||Xw - y||_2^2 \longrightarrow \min_w$$

Заметим, что среднеквадратическая ошибка - выпуклая и гладкая функция. Выпуклость гарантирует единственность минимума, а гладкость - существование вектора градиента в каждой точке. Поэтому для поиска решения используется метод градиентного спуска:

$$w^0 \in \mathcal{N}(0, 1)$$

$$w^t = w^{t-1} - \eta_t \nabla Q(w^{t-1}, X)$$

$$||w^t - w^{t-1}|| < \varepsilon$$

5.2 Использование модели линейной регрессии для прогнозирования временных рядов

Предложенная предиктивная модель является логическим усовершенствованием модели ARIMA(p, d, q).

ARIMA(p, d, q) - интегрированная модель авторегресси - скользящего среднего, которая представляет из себя расширение модели ARMA(p, q) для нестационарных временных рядов. Иначе говоря, разности нестационарного временного ряда порядка d описываются моделью ARMA(p, q).

В свою очередь, модель ARMA(p, q) состоит из двух моделей:

AR(p) - значение временного ряда в настоящий момент времени линейным образом зависит от предыдущих p значений данного ряда.

MA(q) - значение временного ряда в настоящий момент времени линейным образом зависит от предыдущих q значений ошибки модели.

Как уже было сказано, от чтобы избавиться от нестационарности во временных рядах, был произведён переход к приращениям первого порядка, таким образом мы используем модель ARIMA(p, d, q) с d = 1. Таким образом в новом признаковом пространстве модель ARIMA(p, 1, q) выглядит следующим образом:

$$y_t = \varepsilon_t + \sum_{i=1}^{p} \alpha_i \cdot y_{t-i} + \sum_{i=1}^{q} \beta_i \cdot \varepsilon_{t-i}$$

$\{\varepsilon_t\}$ - независимые и одинаково распределённые случайные величины, с нулевым средним - белый шум, а $\alpha_1, \ldots, \alpha_p$ и $\beta_1, \ldots, \beta_q$ - авторегрессионные коэффициенты и коэффициенты скользящего среднего. Поскольку ошибку наблюдать невозможно, значения $\{\varepsilon_t\}$ из прошлого заменяются на остатки, а из будущего заменяются на нули , так как ошибка в среднем ноль. Поэтому предсказание можно записать следующим образом:

$$\hat{y_t} = \sum_{i=1}^{p} \alpha_i \cdot y_{t-i} + \sum_{i=1}^{q} \beta_i \cdot \hat{\varepsilon_{t-i}}$$

Так вот остатки хорошей предиктивной модели должны обладать следующими свойствами:

1. Несмещённость. Если остатки модели в среднем не равны нулю, то прогноз настолько плох, что если его сдвинуть на константу, то он станет лучше.

2. Стационарность. Если её нет, то в остатках есть структура и мы не смогли выбрать всю информацию из данных.

3. Неавтокореллированность. Если остатки коррелированны, то они неслучайны и несут информацию, которую мы не смогли выбрать из данных.

Остатки подобранной классическими методами модели ARIMA(8,1,5) не удовлетворяли некоторым из этих свойств. Это значит, что класс моделей ARIMA(p, d, q) недостаточно богат, чтобы содержать в себе модель, хорошо описывающую целевую переменную. Поэтому для предсказания используем логическое усовершенствование модели ARIMA(p, d, q) - в качестве признаков будем использовать не только целевую переменную и остатки модели, но и другие метрики, описывающие работу системы. Таким образом предиктивная модель имеет следующий вид:

$$\hat{y}_t = \sum_{i=1}^{p} \alpha_i \cdot y_{t-i} + \sum_{i=1}^{q} \beta_i \cdot \hat{\varepsilon}_{t-i} + \sum_{k=1}^{r} \sum_{i=1}^{p} \gamma_i x_{t-i}^k$$

Где x_{t-i}^k - значение k-го признака в момент времени $(t-i)$. Таким образом, нам необходимо подобрать параметры $\{\alpha_i\}$, $\{\beta_i\}$ и $\{\gamma_i\}$ на обучении, а параметры p, q и r на валидации.

5.2.1 Отбор признаков

Построим самую простую линейную модель на всех рассматриваемых метриках. Так как мы используем MinMaxScaler, то веса при признаках-метриках равносильны их значимости. Тогда для каждой метрики посчитаем её суммарную по всем хранилищам данных значимость для модели.

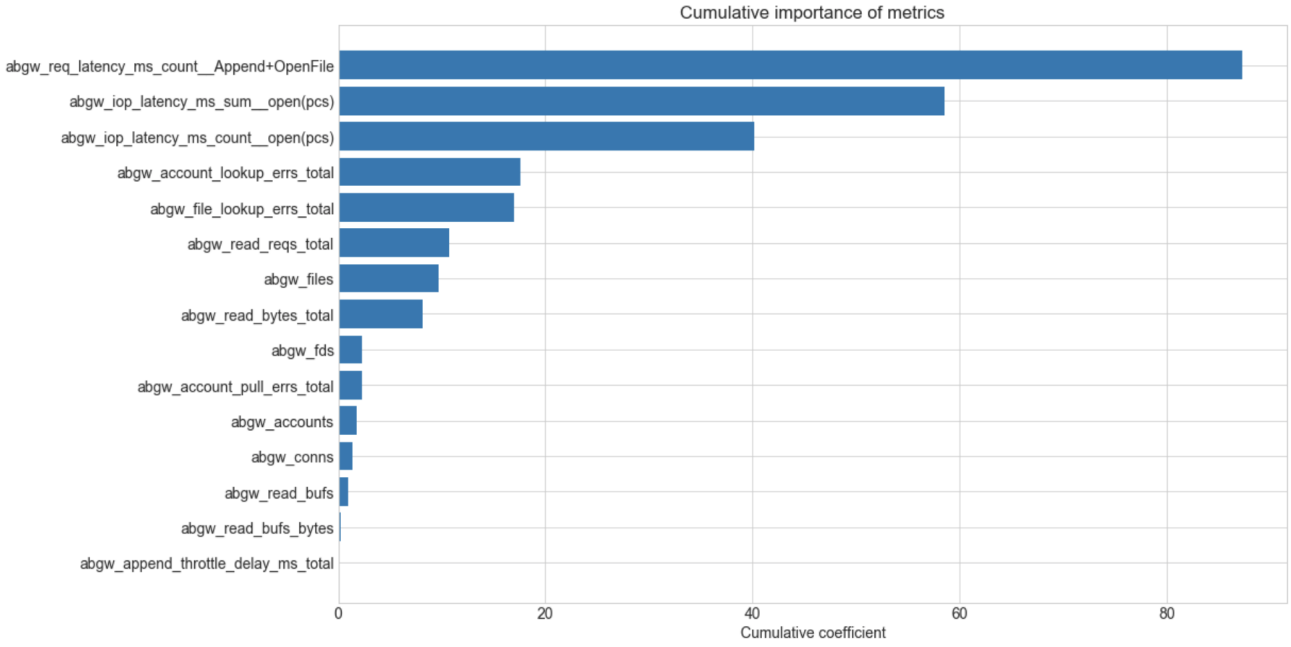

Рис. 8: Суммарная по всем хранилищам данных значимость метрик

Посмотрим, как кумулятивная ошибка зависит от количества признаков в модели:

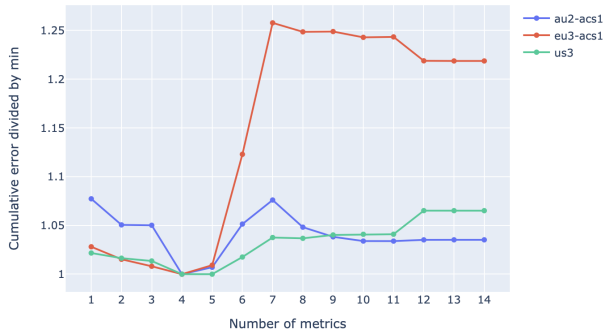

Рис. 9: Кумулятивная ошибка от количества признаков

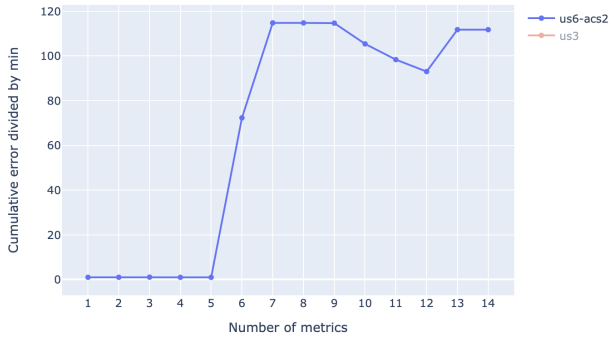

Рис. 10: Кумулятивная ошибка от количества признаков

Таким образом необходимо взять $r = 5$ наиболее значимых метрик.

5.2.2 Поиск оптимального количества задержек целевой переменной и признаков

Построим зависимость средней за минуту ошибки модели от количества задержек:

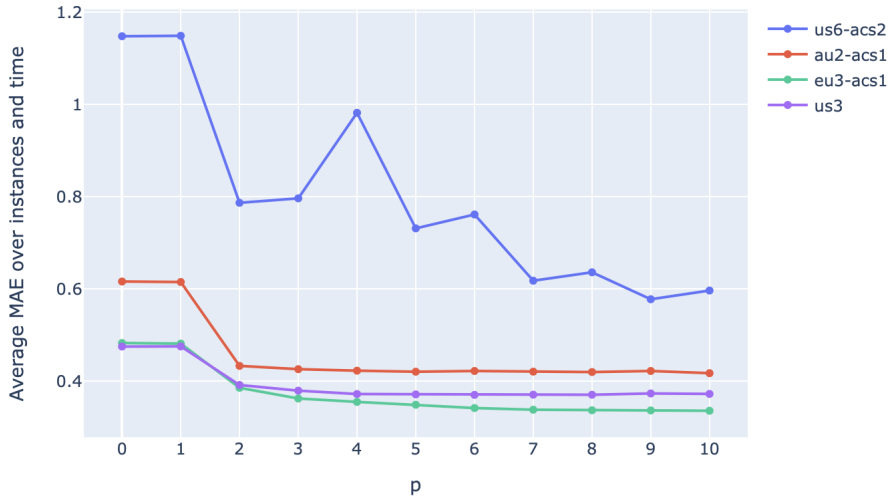

Рис. 11: Средняя за минуту ошибка модели от количества задержек

Можно сделать вывод, что необходимо взять $p = 2$, так как добавление большего числа задержек не приносит значительных улучшений, а лишь вносит дополнительный шум.

5.2.3 Поиск оптимального количества задержек ошибки

Так как мы не можем наблюдать ошибку, то построим модель без задержек ошибки, найдём остатки и построим модели с полученными остатками в качестве задержек ошибки. Посмотрим на зависимость средней по экземплярам хранилища данных кумулятивной ошибки от количества задержек ошибки.

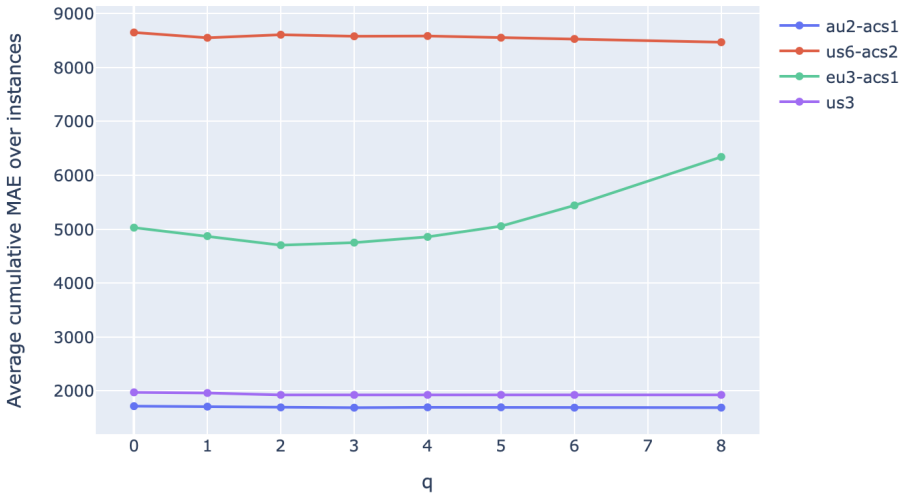

Рис. 12: Средняя за минуту ошибка модели от количества задержек

Можно сделать вывод, что необходимо взять $q = 2$, так как добавление большего числа задержек не приносит значительных улучшений, а лишь вносит дополнительный шум.

6 Поиск аномалий

6.1 Алгоритм

На основе имеющихся временных рядов и предсказаний нашей модели, создадим временные ряды, анализ которых и приведёт нас к обнаружению аномалий. Как уже было отмечено выше, смысл целевой переменной - суммарная задержка рассматриваемых процессов. Логично предположить, что среднее значение и дисперсия задержек процессов входят в число временных рядов, анализ которых даёт информацию о аномальности системы. Чтобы получить эти временные ряды, рассмотрим следующую метрику:

```
abgw_iop_latency_ms_bucket{err="OK", iop="pread", proxied="0"}
```

Данная метрика показывает, как много запросов (нарастающим итогом) в данный момент времени завершилось с задержкой менее, чем соответствующее значение. Из неё получим метрику, содержащую информацию о количестве запросов, которые в данный момент времени завершились с задержкой в соответствующем диапазоне.

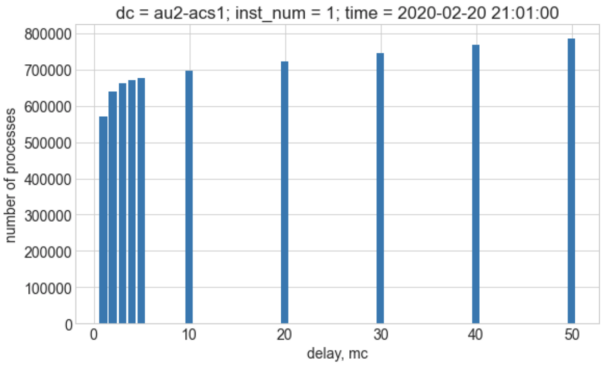

Рис. 13: Количество запросов с задержкой менее, чем соответствующие значения (нарастающим итогом).

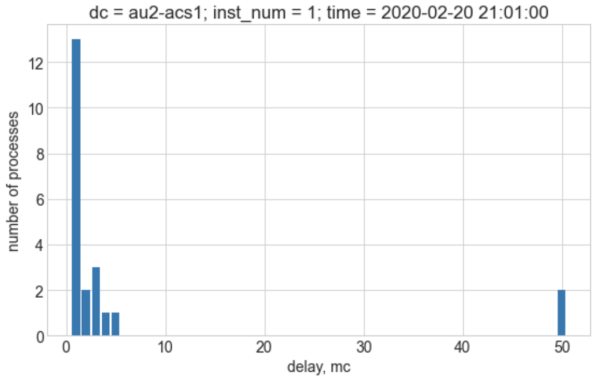

Рис. 14: Количество запросов с задержкой в соответствующем диапазоне.

На основе этого находим выборочное среднее и дисперсию задержек запросов в каждый момент времени.

$$mean_k = \frac{1}{\mathbb{I}([0,\infty])} \cdot \sum_{i,j>i} \mathbb{I}([i,j]) \cdot \frac{j-i}{2}$$

$$variance_k = \frac{1}{\mathbb{I}([0,\infty])} \cdot \sum_{i,j>i} \mathbb{I}([i,j]) \cdot \left(\frac{j-i}{2} - mean_k\right)^2$$

где $\mathbb{I}([i,j])$ - количество запросов, задержка которых содержится в интервале от i ms до j ms.

date	mean	variance
2020-01-01 00:02:00	5.547592	459.308209
2020-01-01 00:03:00	7.253323	659.858491
2020-01-01 00:04:00	10.782562	1175.027361
2020-01-01 00:05:00	5.526192	256.485501
2020-01-01 00:06:00	6.498989	609.439365

Рис. 15: Среднее значение и дисперсия задержки в каждый момент времени

Предложенный алгоритм поиска аномалий опирается на предположение о нормальности среднего значения и дисперсии задержек запросов. Проверим эту гипотезу.

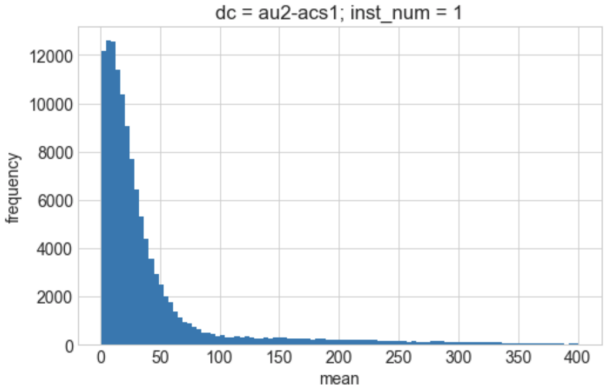

Рис. 16: Гистограмма среднего значения задержки

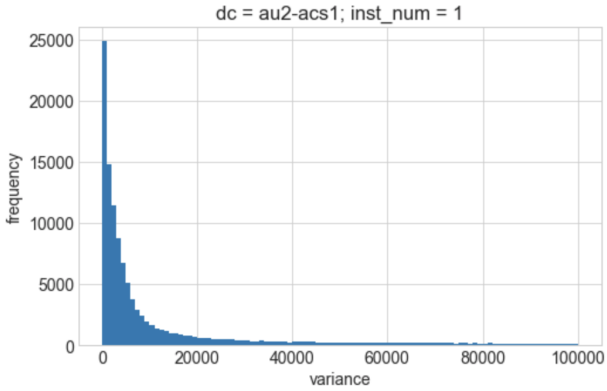

Рис. 17: Гистограмма дисперсии задержки

Для визуальной проверки используем Q-Q plot. В статистике Q-Q plot - это график, который предоставляет графический метод для сравнения двух распределений путем построения их квантилей, где квантиль - это значение, которое случайная величина не превышает с заданной вероятностью

$$F_X(x_\alpha) = \mathbb{P}(X \leqslant x_\alpha) \geqslant \alpha$$

Данные выборки сортируются в порядке возрастания, а затем наносятся на график в сравнении с квантилями, рассчитанными из теоретического распределения. Таким образом, получившаяся кривая является параметрической кривой с параметром, который является номером интервала для квантиля. Если распределения линейно связаны, то точки на графике будут приблизительно лежать на линии. Данный метод просто визуальная

проверка, а не строгое доказательство, но он позволяет нам сразу увидеть, является ли наше предположение правдоподобным или нет.

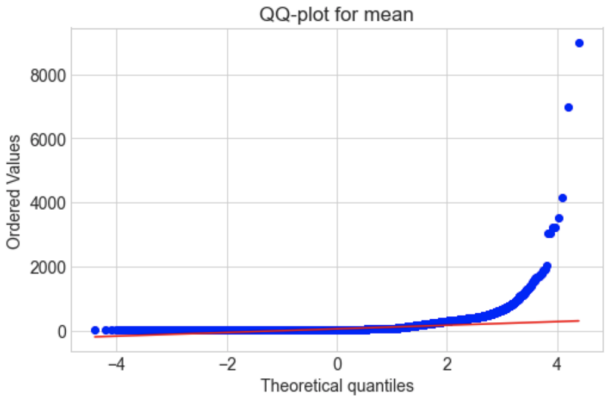

Рис. 18: dc = au2-acs1; inst = 1

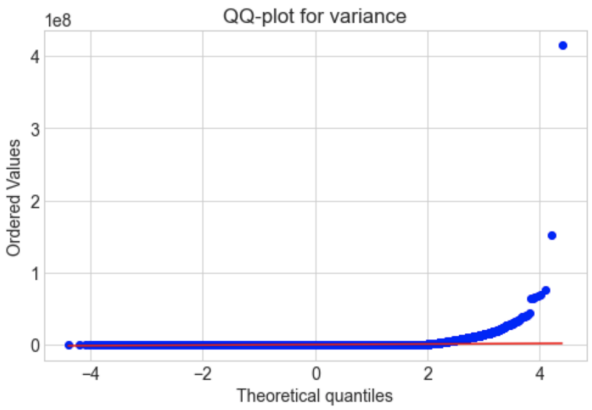

Рис. 19: dc = au2-acs1; inst = 1

Из графиков можно сделать вывод о том, что наше предположение правдоподобно, так как большая часть данных действительно лежит на прямой. Более того, не лежащие на прямой точки являются потенциально аномальными.

Проверим наше предположение с помощью критерия Харке-Бера:
Выборка: X^n
Нулевая гипотеза: $H_0 : X \sim \mathcal{N}(\mu, \sigma^2)$
Альтернатива: H_0 неверна
Статистика: $\chi^2(X^n) = \dfrac{n}{6} \cdot \left(\gamma_1^2 + \dfrac{1}{4}\gamma_2^2\right)$
Нулевое распределение: χ_2^2
Достигаемый уровень значимости: $p(\chi^2) = 1 - F_{\chi_2^2}(\chi^2)$

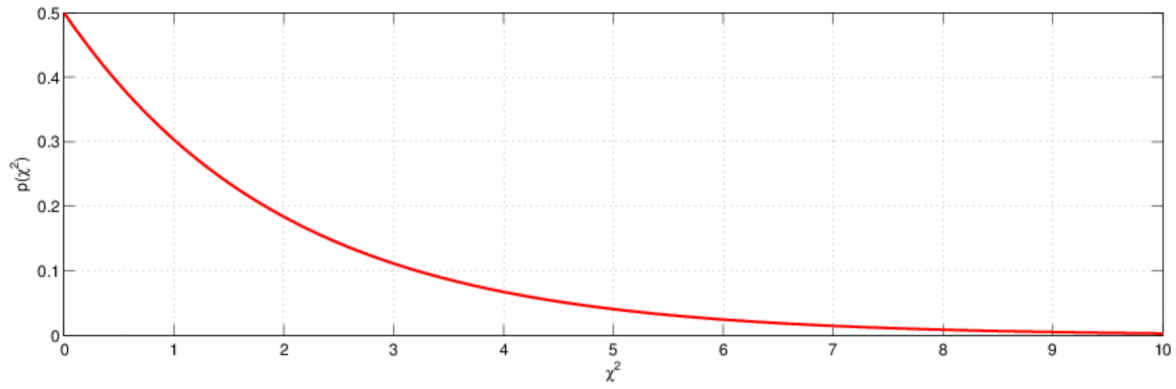

Рис. 20: Достигаемый уровень значимости

Так как подобные тесты очень чувствительны к выбросам, для проверки используем семплирование и усреднение p-value по семплам. Тест показал, что гипотеза нормальности среднего значения и дисперсии задержек процессов принимается с уровнем значимости $\alpha = 0.05$.

Зная реальное значение целевой переменной и её предсказание, мы можем найти ошибку модели в каждый момент времени - это ещё один временной ряд, который нам необходим для идентификации аномального поведения системы. Масштабирование ошибки необходимо для лучшей интерпретируемости, в идеале

$$0 \leqslant error \leqslant 1$$

Но MinMaxScaler не стабилен для данных с выбросами, так как порядок целевой переменной сильно зависит от наличия выбросов в данных, как следствие это влияет на точность модели, что в свою очередь ведёт к ухудшению интерпретируемости результатов. Поэтому для оценки качества модели используем более робастный вариант - масштабирование делением на среднее значение:

$$error_i = \frac{|a(x_i) - y_i|}{\frac{1}{l} \cdot \sum_i^l y_i}$$

Также предложенный алгоритм поиска аномалий опирается на предположение о нормальности ошибки модели. Проверим эту гипотезу.

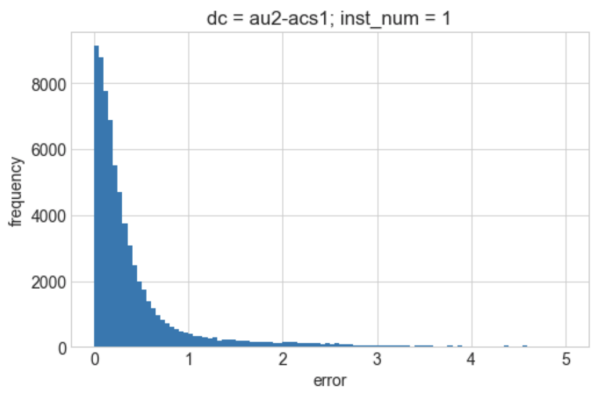

Рис. 21: dc = au2-acs1; inst = 1

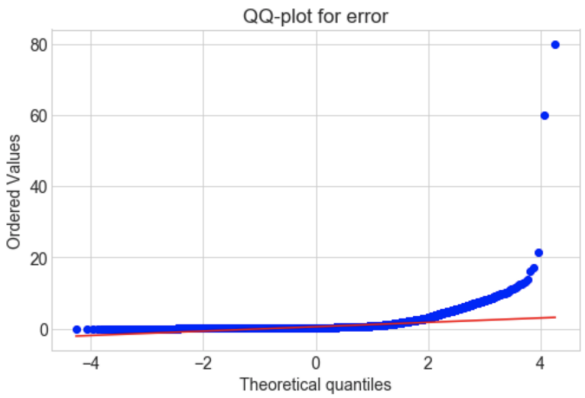

Рис. 22: dc = au2-acs1; inst = 1

Аналогичное применение критерия Харке-Бера показало, что гипотеза нормальности ошибки модели принимается с уровнем значимости $\alpha = 0.05$.

Таким образом мы имеем три нормально распределённые случайные величины для анализа:

1. `Среднее значение задержек запросов`
2. `Дисперсия задержек запросов`
3. `Ошибка модели = отклонение поведения целевой переменной от ожидаемого`

Правило трёх сигм - с вероятностью $0,9973$ значения нормально распределённой случайной величины лежат в интервале $(\mu - 3\sigma, \mu + 3\sigma)$. Мы используем некую вариацию данного правила, используя логичное предположение о том, что нормальное поведение системы находится в зоне высокой вероятности, а аномальное - вне этой зоны.

То есть будем считать, что нормальное поведение системы находится в интервале $(\mu - k\sigma, \mu + k\sigma)$, где k - гиперпараметр, который мы подберём учитывая мнения экспертов. Очевидно, что значение k зависит от характерной частоты аномального поведения системы и от определения отклонения поведения системы от стандартного. Оба параметра могут быть настроены в соответствии с требованиями заинтересованных сторон.

Будем считать, что характерное количество аномалий на некотором хранилище данных за D дней задаётся следующей формулой:

$$n = \frac{N_{dc} \cdot \nu \cdot D}{N}$$

N_{dc} - количество загруженных экземпляров данного хранилища данных
$N = 30$ - максимальное количество экземпляров хранилища данных
ν - характерная частота аномального поведения системы

Дальнейшие вычисления приведены в предположении, что $\nu = \frac{20}{30}$ и $D = 60$. То есть мы предполагаем, что для системы характерно иметь 40 аномалий за 60 дней на полностью загруженном хранилище данных.

Таким образом мы имеем три метрики и решающее правило для поиска аномального поведения. Применение к любой метрике решающего правила позволяет разделить входные данные на две группы - моменты аномального и нормального поведения системы. Наша модель должна удовлетворять требованиям заинтересованных сторон, по мнению которых аномальное поведение системы зачастую длится больше минуты. Поэтому аномальные моменты времени необходимо кластеризовать в аномальные

интервалы времени, длительностью больше одной минуты.

Применим следующий алгоритм:

1. С помощью метрики и решающего правила разделим входные данные на две группы - моменты аномального и нормального поведения системы.

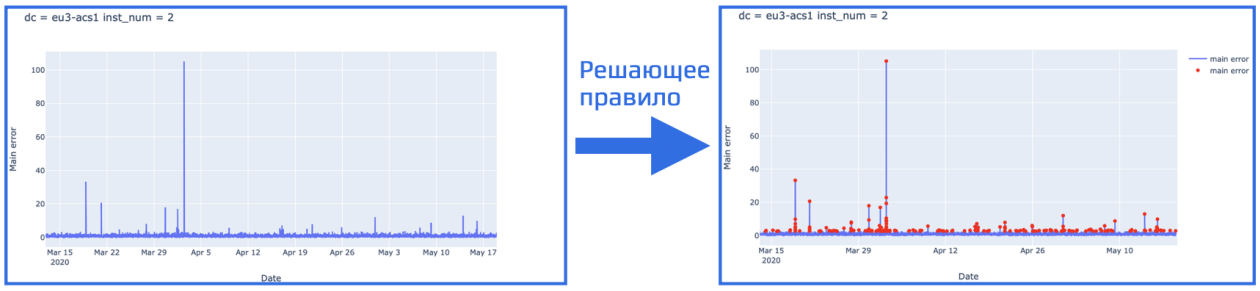

Рис. 23: Применение решающего правила к ошибке модели

2. Для каждой аномальной точки рассмотрим окно в T минут. Если внутри такого окна аномальных моментов времени больше, чем n, то интервал между рассматриваемой аномальной точкой и последней аномальной точкой в окне помечается, как аномальный.

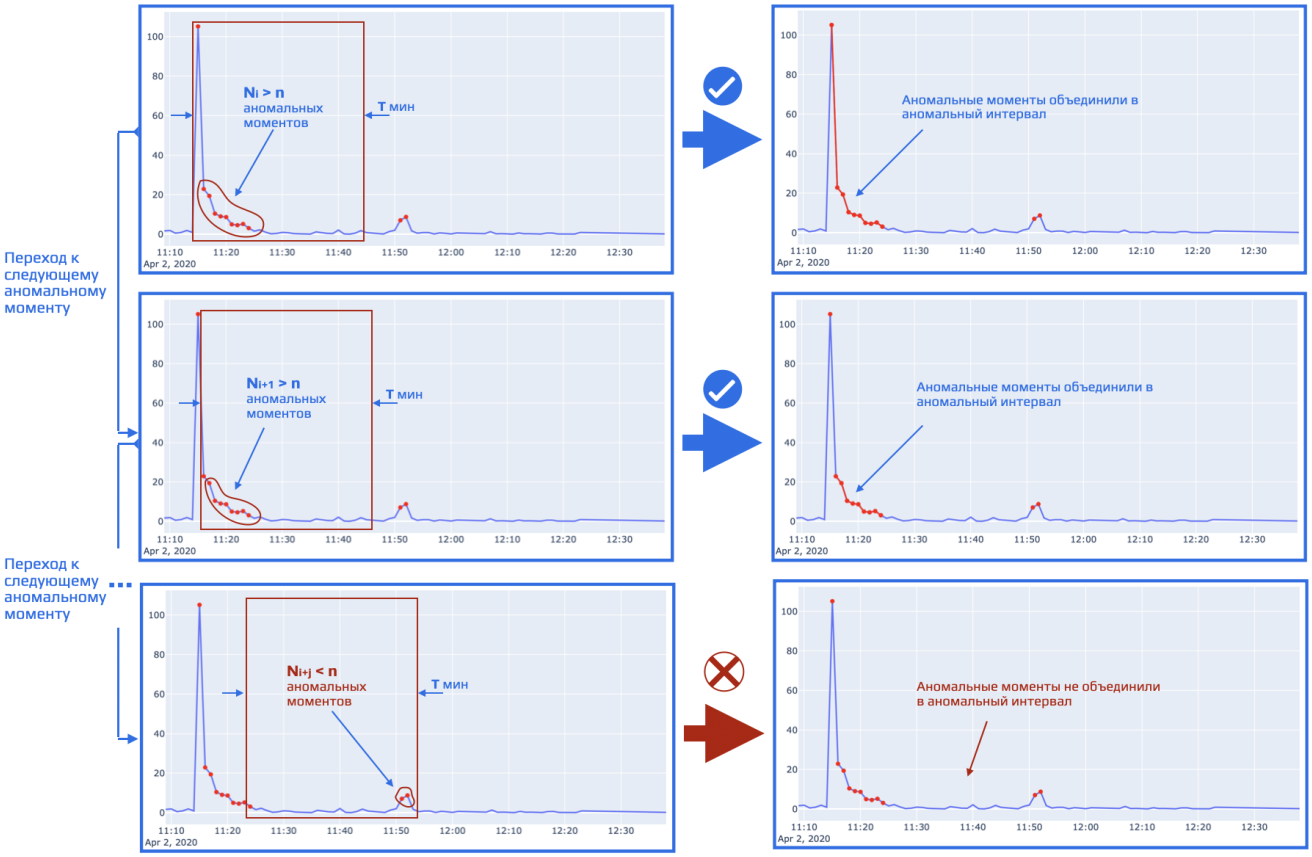

Рис. 24: Процедура построения аномальных интервалов

3. Далее все пересекающиеся аномальные интервалы объединяются.

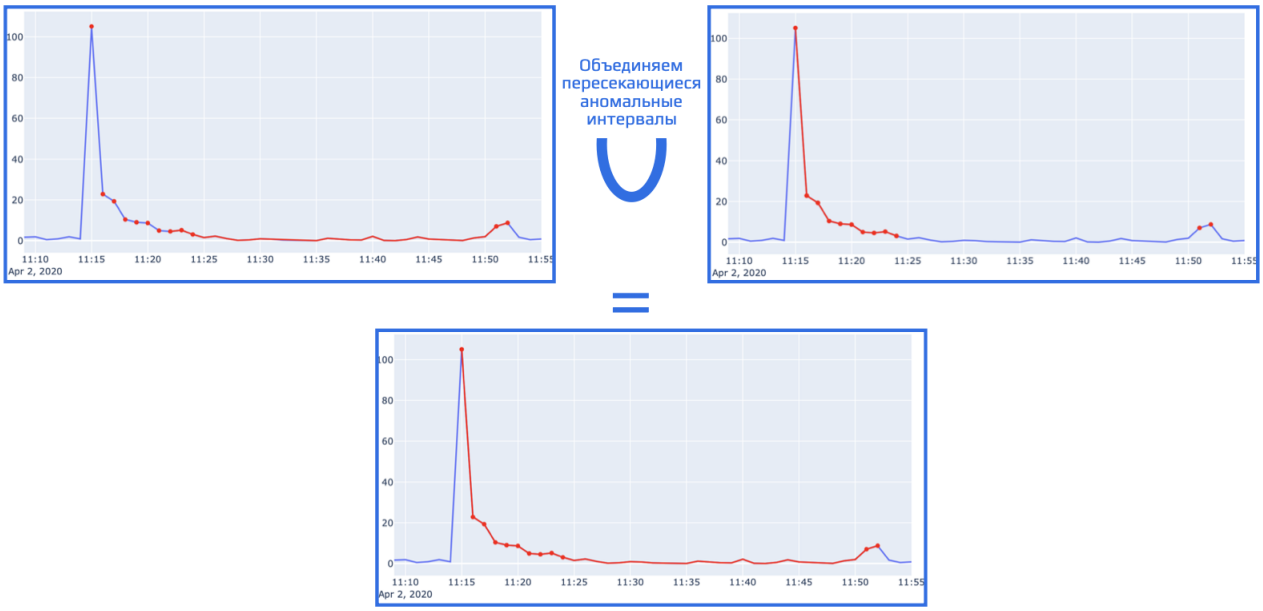

Рис. 25: Процедура объединения аномальных интервалов

Таким образом для рассматриваемой метрики мы получаем аномальные интервалы времени:

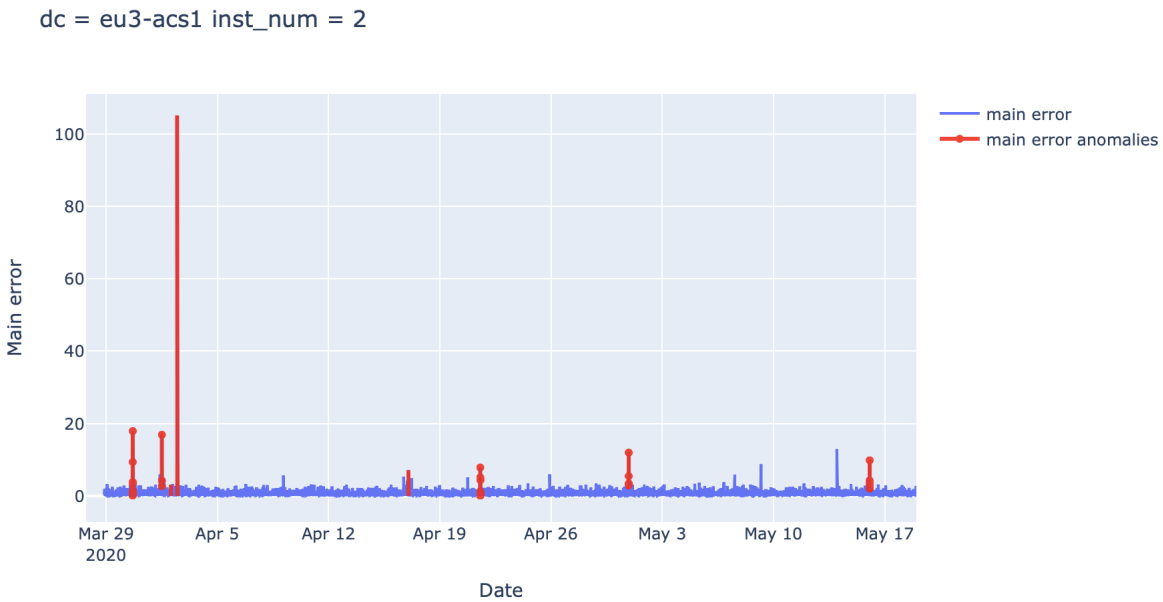

Рис. 26: Итог алгоритма поиска аномальных интервалов по соответствующей метрике

Выберем наиболее информативную метрику для детектирования аномалий. Для этого рассмотрим зависимость количества аномальных интервалов от числа k для имеющихся у нас нормальных случайных величин, характеризующих поведение системы.

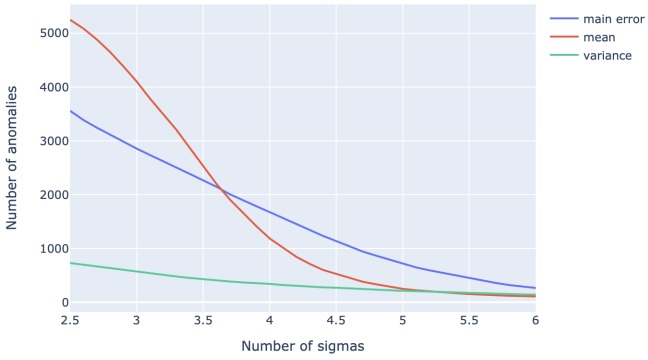

Рис. 27: dc = au2-acs1

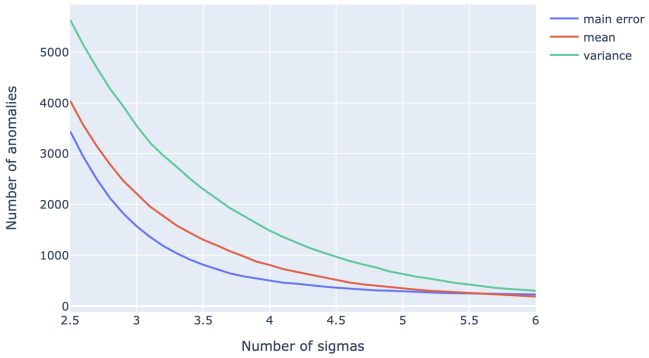

Рис. 28: dc = eu3-acs1

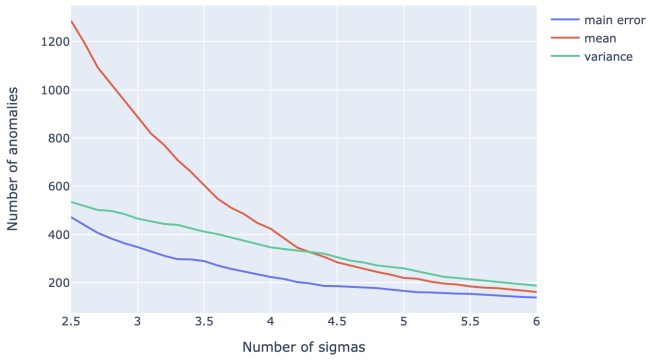

Рис. 29: dc = us3

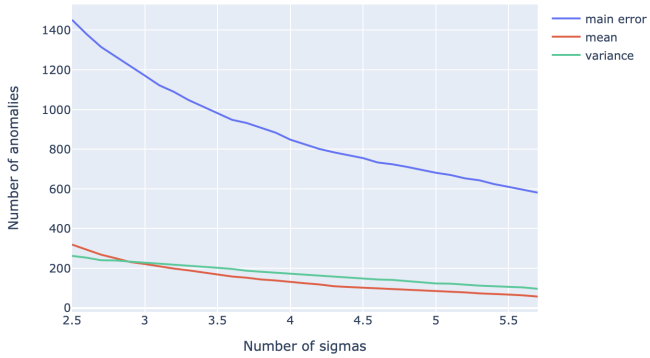

Рис. 30: dc = us6-acs2

Логично предположить, что во время аномального поведения системы сразу несколько из этих параметров должны отклонятся от стандартного поведения. То есть аномальные интервалы, вычисленные по разным метрикам, должны совпадать, ну или, по крайней мере, пересекаться.

Судя по графикам, дисперсия задержек не является универсальной метрикой для поиска аномалий, так как её поведение сильно отличается на разных хранилищах данных. Поэтому рассмотрим сложносоставную метрику - "`логическое И`" результатов поиска аномалий по двум оставшимся метрикам. Таким образом аномальными интервалами будем считать все непустые пересечения аномальных интервалов, найденных с помощью среднего значения задержек процессов и ошибки модели.

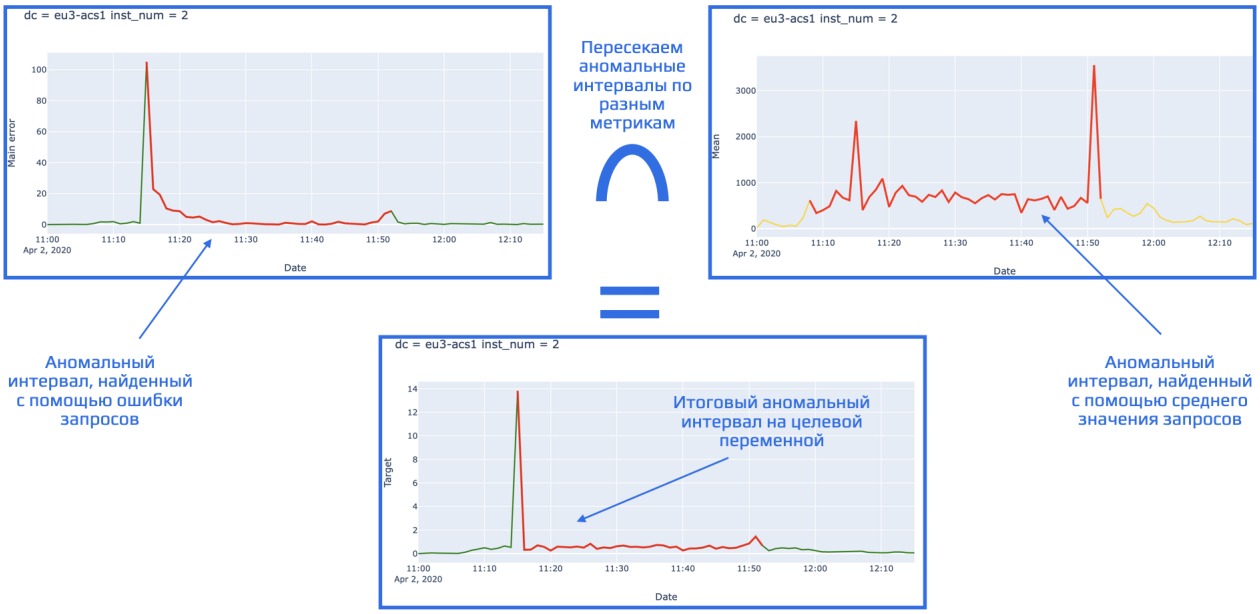

Рис. 31: Сложносоставная метрика

С помощью этой метрики подберём параметр k и протестируем наш алгоритм.

7 Результаты работы алгоритма

	Number of sigmas	Number of anomalies	Expected number of anomalies	Amount of anomaly-time
us3	8	28	20	923
au2-acs1	6.4	8	21	65
eu3-acs1	7.6	42	35	688
us6-acs2	5.7	27	29	917

Рис. 32: Сводная таблица результатов работы алгоритма

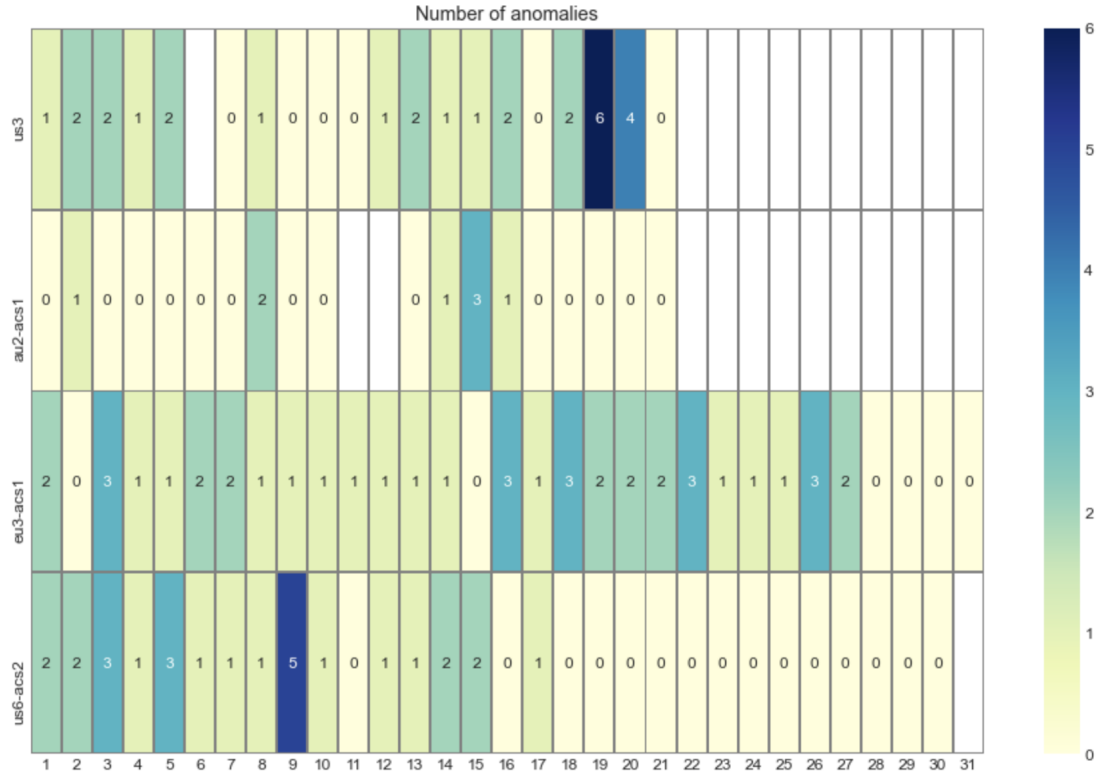

Рис. 33: Тепловая карта количества аномалий

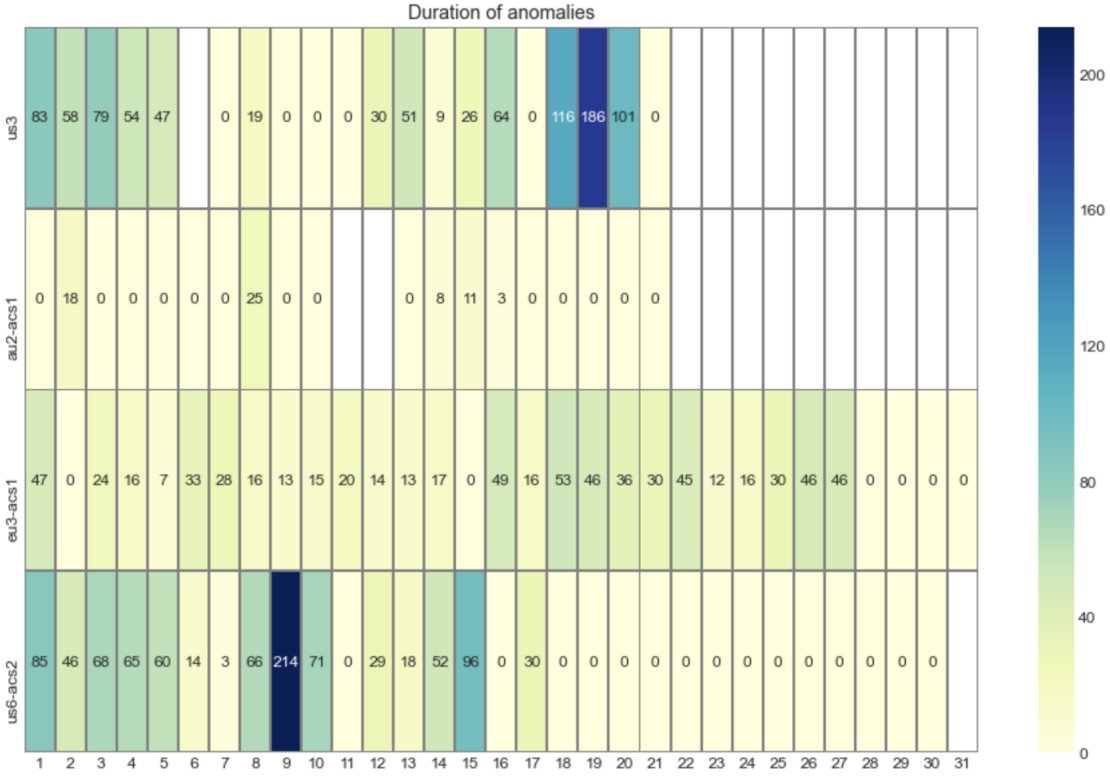

Рис. 34: Тепловая карта суммарной длительности аномалий в минутах

8 Примеры найденных аномалий

Рис. 35: Пример найденной аномалии

9 Проверка корректности алгоритма

По мнению экспертов, аномальное поведение системы зачастую сопровождается резким увеличением 99-го персентиля задержки запросов. Проверим корректность алгоритма следующим образом: если найденная аномалия сопровождается ростом 99-го персентиля задержки запросов, то считаем, что алгоритм сработал верно, иначе считаем, что алгоритм сработал неверно.

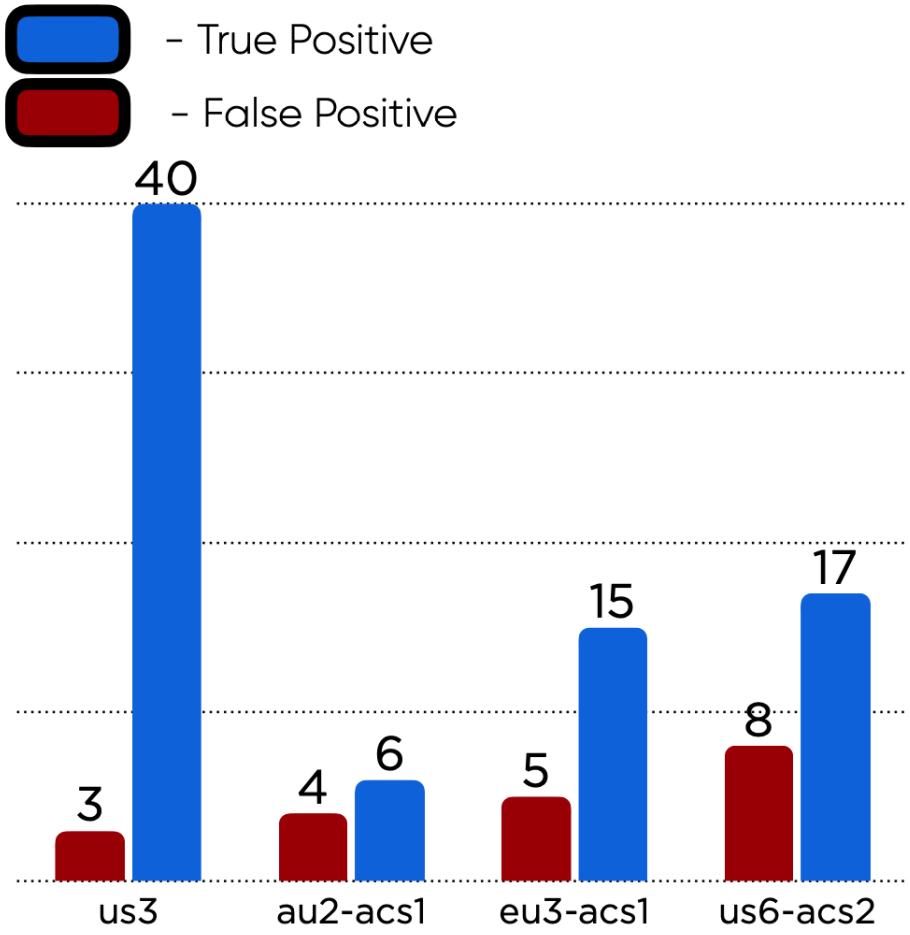

Рис. 36: Корректность алгоритма

Таким образом точность алгоритма:

$$\textbf{accuracy} = \textbf{82\%}$$

10 Заключение

В результате работы была создан универсальный для всего многообразия хранилищ данных компании Acronis алгоритм детектирования аномалий, имеющий высокую точность.

Преимуществами алгоритма являются:

1. Полное отсутсвие ручной работы.
2. Универсальность для всего многообразия хранилищ данных компании Acronis.
3. Легко настраивается под требования заинтересованных сторон.
4. Одинаково корректная работа при любой нагрузке системы.
5. Высокая точность.
6. Низкий уровень ложных срабатываний.

Выявление аномального поведения в автоматическом режиме позволит повысить качество работы системы Acronis Storage, а также поможет предотвращать нештатные ситуации на этапах их зарождения.

Логическим продолжением этой работы является использование алгоритма для разметки исторических данных, которая необходима для построения модели заблаговременного предсказания аномального поведения системы.

Список литературы

[1] *Prometheus - an open-source systems monitoring and alerting toolkit.* URL: https://prometheus.io/.

[2] *G. E. P. Box, G. M. Jenkins, and G. C. Reinsel.* Time Series Analysis: Forecasting and Control, John Wiley & Sons, New York, NY, USA, 2013.

[3] *Arindam Banerjee Varun Chandola and Vipin Kumar.* Anomaly detection : A survey, ACM Computing Surveys, 2009.

[4] *Dominique T. Shipmon, Jason M. Gurevitch, Paolo M. Piselli, Steve Edwards.* Detection of Anomalous Drops with Limited Features and Sparse Examples in Noisy Highly Periodic Data.

[5] *Ane Blazquez-Garcia and Angel Conde.* A review on outlier/anomaly detection in time series data.