Применение сингулярных разложений к обнаружению выбросов в данных о потреблении контента

В. Г. Мосин

Аннотация

Рассмотрен метод выявления аномальных объектов, основанные на редукции данных к главным направлениям и не требующий предварительной разметки данных на нормальные и аномальные объекты. Метод реализован в виде алгоритма и протестирован на данных о потреблении контента одного из ведущих хостингов.

Содержание

1 Введение

Одной из важных задач анализа данных является обнаружение шума в данных и его последующее исключение для получения более надежных результатов, не зависящих от случайных флуктуаций (см. [6], [7]).

Шум в данных — это нежелательные, случайные или неточные значения, которые вносят искажения или искажают истинные данные. Шум может возникать по разным причинам, таким как ошибки измерений, неполадки в оборудовании, случайные факторы или неправильные записи. Шум может иметь различные формы и влиять на данные по-разному. Некоторые типы шума включают случайный шум, выбросы (аберрации), грубые ошибки, систематические искажения и многие другие.

Поскольку шум может вносить искажения в данные, его наличие может затруднить анализ и интерпретацию данных. Это может привести к неправильным выводам и недостоверным результатам. Поэтому важно применять методы обработки данных и фильтрации шума для улучшения качества данных перед анализом или использованием их для принятия решений. Методы борьбы с шумом могут включать фильтрацию, сглаживание (например, сглаживание скользящим средним), применение алгоритмов обнаружения выбросов или применение статистических методов для определения и удаления шумовых значений.

Вместе с тем, важно отметить, что в некоторых случаях шум может содержать полезную информацию или быть результатом естественных изменений в данных. Поэтому необходимо тщательно анализировать и контролировать шум, чтобы минимизировать его влияние на данные и сохранить релевантные и достоверные сведения.

1.1 Теоретическая часть

Обнаружение выбросов относится к предварительной обработке данных, которая, как правило, предшествует основной задаче: задаче моделирования и прогнозирования целевых признаков. Для того чтобы модели

обладали лучшими характеристиками, а их прогнозы были более точными, из обучающих наборов данных важно исключать записи, являющиеся нетипичными или аномальными (см. [7]).

Это связано с тем, что обучение модели на типичных объектах позволяет ей лучше улавливать общие закономерности и характеристики в данных. Аномальные объекты могут содержать случайные или экстремальные значения, которые могут отличаться от общих трендов и снизить способность модели к обобщению на новые, неизвестные объекты.

Кроме того, аномальные объекты могут вносить существенные искажения в данные и повлиять на модель, если они включены в процесс обучения. Модель может стремиться адаптироваться к этим выбросам, в результате чего потеряется способность модели к правильному предсказанию на типичных данных. И наконец, обучение на типичных объектах помогает создать более надежную и стабильную модель. Такие модели обычно лучше справляются с новыми данными и более точно предсказывают результаты. Тогда как модель, обученная на аномальных объектах, может быть менее надежной и неустойчивой к изменениям в данных.

Не менее важной является и задача прогнозирования (см. [6]). Обученная стабильная модель, дающая хорошие результаты на типичных объектах, вполне может оказаться непригодной при попытке спрогнозировать значения целевых признаков в ситуациях, сильно отличающихся от тех, на которых модель обучалась.

Это проявляется, прежде всего, в низкой точности прогнозов. Если модель не была обучена на разнообразных данных и не встречала сильно отличающиеся друг от друга примеры, ее прогноз может быть неточным на таких данных. Модель может не уловить закономерности или связи между признаками в новых данных, что приведет к плохим прогнозам.

Вдобавок к этому происходит потеря обобщающей способности. Если модель была обучена на данных с определенными характеристиками и структурой, она может не получить достаточной обобщающей способности для предсказания на данных с другой структурой или характеристиками. Необходимо помнить, что модель машинного обучения является результатом обучения на конкретных данных, и ее прогнозы оказываются наиболее точными в пределах набора данных, на котором она обучалась. Для этого нужно уметь различать объекты, подаваемые на вход модели для прогнозирования, нужно уметь выявлять нетипичные объекты и исключать их из процесса еще до подачи в модель.

1.2 Постановки задачи

Если данные, используемые для обучения модели, являются размеченными, то есть, каждый объект обладает меткой «норма/аномалия», то

задача обнаружения выбросов на обучающих данных вообще не стоит, а обнаружение выбросов на новых данных сводится к задаче бинарной классификации по признаку-метке (см. [4]).

Допустим, что изначально у нас нет никаких сведений о том, какие объекты относятся к числу типичных объектов, пригодных для обучения модели, а какие — к аномальным, то есть, тем, которые следует исключить из процесса обучения, чтобы избежать неприятностей, о которых было подробно сказано выше.

1.2.1 Предмет исследования

Предметом нашего исследования является метод, позволяющий выявить нетипичные объекты в данных, не являющихся изначально размеченными по признаку «норма/аномалия».

1.2.2 Методика исследования

Метод основан на сингулярных матричных разложениях и последующей редукции данных к главным направлениям. Если редуцированные данные обладают слишком большой реконструкционной ошибкой, то такие объекты признаются аномальными. Мы применяем в качестве критерия выброса процент от общего объема обучающих данных, то есть, относим к типичным их объектам K-й перцентиль (где K достаточно велико), а объекты, не попадающие в него, объявляем нетипичными выбросами.

1.2.3 Цель исследования

Наша цель — получить алгоритм, индексирующий объекты обучающей выборки как аномальные по заданному перцентилю выбросов. Кроме того, мы собираемся применить этот алгоритм к новым данным, и, используя полученный на обучающей выборке порог отсечения, выделить в них нетипичные объекты.

1.3 Библиотеки

Для выполнения вычислений и анализа данных мы пользуемся средой `Jupyter Notebook`, которая предоставляет удобные средства для работы с языком программирования Python и его главными библиотеками: `NumPy`, `Pandas`, `sklearn` и `matplotlib`. Благодаря этим инструментам, мы можем эффективно работать с данными, выполнять исследования и визуализировать результаты (см. [1], [2]).

2 Описание данных

Используются данные о потреблении контента одного из ведущих хостингов за 500 дней: с 2021-08-20 по 2023-01-01. Объектами таблицы служат даты. Признаки описывают различные характеристики объекта: 'Просмотры', 'Время просмотра (часы)' и т. д. Полную структуру данных см. ниже, шаг 3.2 алгоритма исследования.

3 Алгоритм

3.1 Чтение данных

Методом `read_csv` библиотеки `pandas` загружаем в среду исполнения набор данных и формируем дата-фрейм.

	Дата	Просмотры	Поделились	...	Лайки
0	2023-01-01	475.0	9.0	...	16.0
1	2022-12-31	174.0	1.0	...	4.0
2	2022-12-30	490.0	3.0	...	3.0
...	...	...	...	...	...
498	2021-08-21	222.0	0.0	...	4.0
499	2021-08-20	209.0	0.0	...	1.0

В данных присутствуют 500 объектов. В качестве индекса выступает дата, таким образом, объектами служат даты. Все признаки относятся к типу с плавающей запятой, пропущенных значений нет.

3.2 Разделение на train и test

Используем модуль `model_selection` библиотеки `sklearn`. Применяем метод `train_test_split` с параметром `test_size=0.2`. Получаем два датафрейма: `df_train` объемом 400 объектов и `df_test` объемом 100 объектов.

3.3 Нормализация обучающей выборки

Применяем к обучающей выборке метод `describe` библиотеки `pandas` и получаем сведения о статистиках по всем признакам.

	min	mean	max	std
Просмотры	159.00	936.51	2200.00	418.144
Время просмотра (часы)	5.48	37.17	96.72	16.64
Поделились	0.00	6.96	71.00	6.25
Постоянные зрители	30.00	163.34	463.00	78.90
Новые комментарии	0.00	0.53	6.00	0.83
Отказались от подписки	0.00	2.77	29.00	2.55
Новые подписчики	0.00	6.49	19.00	3.51
Новые зрители	60.00	366.83	735.00	174.01
Среднее число просмотров одним пользователем	1.31	1.79	2.85	0.21
Уникальные зрители	96.00	530.18	1103.00	239.22
CTR для значков видео (%)	1.25	5.54	8.52	1.11
Показы	1938.00	8093.78	39479.00	3816.08
Подписчики	0.00	3.72	15.00	4.02
Средний процент просмотра (%)	18.68	26.72	41.29	3.41
Процент лайков	0.00	92.02	100.00	10.31
Средняя продолжительность просмотра	96.07	144.33	211.02	15.66
Дизлайки	0.00	1.28	10.00	1.34
Лайки	0.00	15.80	70.00	9.13

Значения некоторых признаков (например, 'Показы' и 'Среднее число просмотров одним пользователем') отличаются на порядки. Чтобы избежать дисбаланса размерностей, нормализуем данные обучающей выборки приведением к стандартному виду:

```
df_train_norm = (df_train - df_train.mean())/df_train.std()
```

После этого все признаки, во-первых, оказываются центрированными, то есть, обладают нулевыми средними, а во-вторых, их дисперсии становятся единичными, и дисбаланс размерностей исчезает:

	min	mean	max	std
Просмотры	–1.85	0.00	3.02	1.00
Время просмотра (часы)	–1.90	0.00	3.57	1.00
Поделились *–1.11	0.00	10.23	1.00	

	min	mean	max	std
Постоянные зрители	–1.69	0.00	3.79	1.00
Новые комментарии	–0.64	0.00	6.57	1.00
Отказались от подписки	–1.08	0.00	10.27	1.00
Новые подписчики	–1.84	0.00	3.55	1.00
Новые зрители	–1.76	0.00	2.11	1.00
Среднее число просмотров одним пользователем	–2.23	0.00	4.92	1.00
Уникальные зрители	–1.81	0.00	2.39	1.00
CTR для значков видео (%)	–3.84	0.00	2.67	1.00
Показы	–1.61	0.00	8.22	1.00
Подписчики	–6.63	0.00	2.80	1.00
Средний процент просмотра (%)	–2.35	0.00	4.26	1.00
Процент лайков	–8.92	0.00	0.77	1.00
Средняя продолжительность просмотра	–3.08	0.00	4.25	1.00
Дизлайки	–0.95	0.00	6.49	1.00
Лайки	–2.38	0.00	5.93	1.00

3.4 Сингулярное разложение

Сингулярное разложение матрицы — это представление матрицы X в виде произведения

$$X = U\Sigma V^{-1}$$

где U и V — ортогональные матрицы, а Σ — диагональная матрица той же конфигурации, что и X. Столбцы матриц U и V называются соответственно левым и правым сингулярными базисами, а диагональные элементы матрицы Σ — сингулярными значениями матрицы X.

Теорема о сингулярном разложении утверждает, что такое разложение существует для любой матрицы X, в частности, оно существует для нашей матрицы обучающих данных.

3.4.1 Вывод массива из датафрейма

Для сингулярного разложения данных обучающей выборки мы будем пользоваться методами библиотеки `numpy`, поэтому, прежде чем приступать к разложению, мы переводим датафрейм `df_train_norm` в массив

`numpy` при помощи метода `to_numpy` библиотеки `pandas`. В результате получаем двумерный массив `X_train` нужного нам формата.

3.4.2 Сингулярное разложение на train'е

Получение левого и правого сингулярных базисов и множества сингулярных значений происходит за счет метода `svd` из модуля `linalg` библиотеки `numpy`. Применяем этот метод к матрице `X_train`, он возвращает три объекта:

1. двумерный массив `U` (столбцы которого образуют левый сингулярный базис),
2. одномерный массив `Sigma` (элементы которого служат диагональю матрицы Σ),
3. и двумерный массив `V` (который представляет собой уже обращенную матрицу V, то есть, строки этого массива образуют правый сингулярный базис).

Согласно общей теории, метод `svd` должен был бы возвращать матрицу V, а не ее обращение. Однако в библиотеке `numpy` метод `svd` реализован именно так, а с учетом того, что обращение ортогональной матрицы эквивалентно ее транспонированию, фактически этот метод возвращает не матрицу V, а транспонированную матрицу V^{T}.

3.4.3 Приведение train'а к сингулярному базису

Если матричное равенство из теоремы о сингулярном разложении умножить на V справа, то оно приобретает вид:

$$XV = U\Sigma$$

Заметим, что после нормализации данные оказались центрированными, поэтому левая часть этого равенства представляет собой не что иное, как теорему о замене базиса (см. [3]), то есть, строки матрицы, расположенной в левой части — это координаты точек облака обучающих данных в правом сингулярном базисе.

Чтобы получить координаты облака обучающих данных в правом сингулярном базисе, мы заводим массив `S_train` и при помощи метода `dot` библиотеки `numpy` присваиваем ему результат матричного произведения массивов `X_train` и `V.T`, где правый множитель мы транспонировали из-за специфики реализации алгоритма сингулярного разложения в библиотеке `numpy` (см. замечание выше).

3.5 Проекция train'а на первое направление

В этом исследовании для редукции данных мы будем использовать одно главное направление, а именно: первое главное направление, то есть, то,

которое отвечает наибольшему сингулярному значению. Данные, редуцированные к первому главному направлению — это проекция точек облака данных на прямую, порожденную первым сингулярным вектором. Поэтому координаты редуцированных данных в сингулярном базисе получаются заменой всех координат, кроме первой, на нулевые значения.

Чтобы получить координаты редуцированных данных в сингулярном базисе мы берем, полученный выше массив `S_train`, заменяем нулями все его столбцы, кроме первого, и присваиваем получившемуся массиву имя `S_trian_reduce`.

Строки массива S_trian_reduce представляют собой координаты редуцированных данных в сингулярном базисе, в тот момент как нас интересуют их координаты в исходном базисе. Чтобы получить координаты редуцированных данных в старом базисе, нужно еще раз воспользоваться теоремой о замене базиса и выполнить обратный переход, но применить его не к полной матрице `S_trian`, а к ранее редуцированной матрице `S_trian_reduce`.

Мы заводим массив `X_train_reduce` и при помощи метода `dot` библиотеки `numpy` присваиваем ему результат матричного произведения массивов `S_train_reduce` и `V`. Теперь строки массива `X_train_reduce` содержат координаты проекций исходных данных на первое главное направление.

3.6 Реконструкционная ошибка на train'е

Реконструкционной ошибкой называется норма разности между исходными и редуцированными данными. Другими словами, это расстояние от точки облака данных до ее проекции на прямую, заданную первым главным направлением. Те объекты, у которых реконструкционная ошибка слишком велика, мы будем признавать нетипичными и объявлять их выбросами. Для получения реконструкционной ошибки мы используем метод `norm` из модуля `linalg` библиотеки `numpy`, применяя его к разности массивов `X_train_reduce - X_train` и указав значение атрибута `axis=1`, чтобы вычислению нормы разности подвергались именно строки.

Результат записываем в массив `reconstruction_error_train`, в нем оказываются реконструкционные ошибки объектов обучающей выборки.

3.7 Порог отсечения

Мы будем устанавливать порог реконструкционной ошибки, превышение которого будем считать выбросом в данных, исходя из объема обучающей выборки. Допустим, при обучении какой-либо модели мы готовы пожертвовать $K\%$ данных для того, чтобы оставшиеся данные давали

более устойчивую модель. Тогда в качестве порога отсечения мы выбираем $(100 - K)$-й перцентиль массива всех реконструкционных ошибок.

Для этого мы применяем метод `pecentil` библиотеки numpy к массиву рекнострукционных ошибок `reconstruction_error_train`, в результате чего получается число, его мы записываем в специально для этого заведенную новую переменную `threshold_train`.

3.8 Выбросы в train'e

Теперь для получения нетипичных объектов обучающей выборки нам достаточно применить метод `where` библиотеки `numpy` с условием:

```
reconstruction_error_train > threshold_train
```

Этот метод возвращает индексы выбросов, то есть, некий набор номеров, которые мы записываем в массив `anomaly_indices_train`. Результаты этого действия визуализированы (см. рис. 1) и описаны в комментарии к визуализации. Заметим, что объем выбросов на обучающей выборке всегда будет одним и тем же: при любом случайном разбиения данных на обучающую и тестовую выборки алгоритм будет помечать $K\%$ обучающей выборке как выбросы.

3.9 Нормализация тестовой выборки

Применяем к тестовой выборке метод `describe` библиотеки `pandas` и получаем сведения о статистиках по всем признакам.

	min	mean	max	std
Просмотры	126.00	916.10	1805.00	422.82
Время просмотра (часы)	4.69	36.23	83.17	17.23
Поделились	0.00	7.15	49.00	6.90
Постоянные зрители	21.00	161.16	354.00	79.91
Новые комментарии	0.00	0.66	6.00	0.89
Отказались от подписки	0.00	3.37	95.00	9.46
Новые подписчики	1.00	6.62	15.00	3.32
Новые зрители	46.00	354.15	693.00	168.35
Среднее число просмотров одним пользователем	1.40	1.80	3.04	0.24
Уникальные зрители	70.00	515.31	1030.00	239.32
CTR для значков видео (%)	1.21	5.44	8.43	1.34

	min	mean	max	std
Показы	1701.00	8418.39	69432.00	6887.19
Подписчики	–85.00	3.25	13.00	9.58
Средний процент просмотра (%)	16.86	26.42	42.56	4.23
Процент лайков	0.00	92.54	106.25	11.92
Средняя продолжительность просмотра	88.6	142.94	195.57	18.22
Дизлайки	–1.00	1.12	5.00	1.18
Лайки	0.00	17.86	63.00	12.48

Данные тестовой выборки необходимо привести к облаку обучающей выборки. Для этого мы выполняем центрирование и масштабирование, применяя к тестовой выборке средние и дисперсии обучающей выборки:

```
df_test_norm = (df_test - df_train.mean())/df_train.std()
```

Заметим, что после этой операции средние значения признаков тестовой выборки отличаются от нуля, а их дисперсии не являются единичными, так как и центр облака и его рассеяние были настроены ранее на обучающей выборке:

	min	mean	max	std
Просмотры	–1.93	–0.04	2.07	1.01
Время просмотра (часы)	–1.95	–0.05	2.76	1.03
Поделились	–1.11	0.02	6.71	1.10
Постоянные зрители	–1.80	–0.02	2.41	1.01
Новые комментарии	–0.64	0.15	6.57	1.07
Отказались от подписки	–1.08	0.23	36.13	3.70
Новые подписчики	–1.56	0.03	2.41	0.94
Новые зрители	–1.84	–0.07	1.87	0.96
Среднее число просмотров одним пользователем	–1.81	0.04	5.83	1.13
Уникальные зрители	–1.92	–0.06	2.08	1.00
CTR для значков видео (%)	–3.88	–0.08	2.58	1.20
Показы	–1.67	0.08	16.07	1.80
Подписчики	–22.03	–0.11	2.30	2.38
Средний процент просмотра (%)	–2.88	–0.08	4.63	1.24

	min	mean	max	std
Процент лайков	–8.92	0.05	1.37	1.15
Средняя продолжительность просмотра	–3.55	–0.08	3.27	1.16
Дизлайки	–1.70	–0.12	2.77	0.88
Лайки	–1.73	0.22	5.16	1.36

Подобно тому, как мы делали это выше, мы переводим датафрейм тестовых данных `df_test_norm` в массив при помощи метода `to_numpy` библиотеки pandas. В результате получаем двумерный массив `X_test` нужного нам формата: это массив, в нем 100 строк и 18 столбцов.

3.10 Реконструкционная ошибка на test'е

Координаты точек тестовой выборки в сингулярном базисе мы вычисляем точно так же, как и выше. При этом мы не выполняем новое сингулярное разложение для тестовых данных, а используем правый сингулярный базис из разложения, полученного для обучающей выборки. В результате получается двумерный массив `S_test`.

Затем мы редуцируем этот массив до первого столбца, получая при этом массив `S_test_reduce`, и возвращаемся к исходному базису, умножая редуцированные сингулярные координаты на матрицу `V`. В итоге возникает массив `X_test_reduce`, строки которого представляют собой проекции точек тестовой выборки на первое главное направление обучающей выборки.

После этого мы выстраиваем массив реконструкционных ошибок тестовых данных `reconstruction_error_test`.

3.11 Выбросы в test'е

Порог отсечения реконструкционной ошибки мы получили выше, когда мы вычислили его на обучающей выборке. К тестовой выборке мы применяем тот же самый порог отсечения:

```
reconstruction_error_test > threshold_train
```

после чего метод `where` библиотеки `numpy` возвращает индексы нетипичных объектов. Результат этого действия представлен на рис. 2 и описан в комментарии к визуализации.

Заметим, что для тестовой выборки количество выбросов может получиться произвольным. Например, алгоритм может сообщить, что в тестовой выборке вообще нет выбросов, или, наоборот, в тестовой выборке

может оказаться гораздо больше выбросов, чем в обучающей. В силу того, что порог отсечения настраивался на обучающей выборке, результат будет зависеть от того, как соотносятся облака тестовых и обучающих данных. На этом шаге мы получаем список `anomaly_indices_test`, индексы тех объектов, которые алгоритм посчитал нетипичными объектами среди тестовых данных по отношению к обучающим данным.

4 Результаты

Напомним, что наши данные о потреблении контента индексированы датами, то есть, объектами (не важно, типичными или нетипичными) являются даты, которые характеризуются рядом параметров: числом просмотров, количеством лайков и т. д. В нашем распоряжении есть 500 записей по датам с 2021-08-20 по 2023-01-01, которые мы случайным образом разбиваем на обучающую и тестовую выборки. Соотношение разбиения составляет 80% к 20%, таким образом, в обучающую выборку попадает 400 объектов, а в тестовую — 100 объектов.

Для того чтобы исключить выбросы из обучения модели, мы готовы пожертвовать 2% обучающих данных, и, установив в качестве гиперпараметра алгоритма значение $K = 2$, мы запускаем наш алгоритм определения выбросов. На рисунке ниже (см. рис. 1) по горизонтали откладываются даты наблюдений, а по вертикали — соответствующая дате реконструкционная ошибка. Горизонтальная прямая устанавливает порог и отсекает нетипичные объекты.

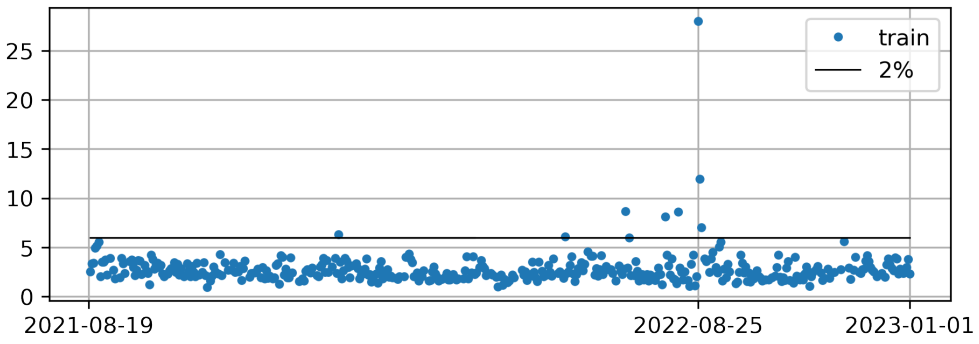

Рис. 1: Выбросы обучающей выборки

На обучающей выборке алгоритм естественным образом возвращает 8 объектов (это ровно 2% от 400 объектов обучающей выборки). А именно, алгоритм рекомендует детектировать как обладающие нетипичными признаками следующие даты:

2022-07-12, 2022-01-18, 2022-06-05, 2022-08-13, 2022-08-05, 2022-08-25, 2022-08-27, 2022-08-26.

Затем мы применяем алгоритм к тестовой выборке для того, чтобы исключить нетипичные объекты из пула объектов, для работы прогнозирующей модели, так как на нетипичных объектах (читай — выбросах) модель заведомо будет давать нелепый прогноз, в силу того, что обучалась на типичных данных.

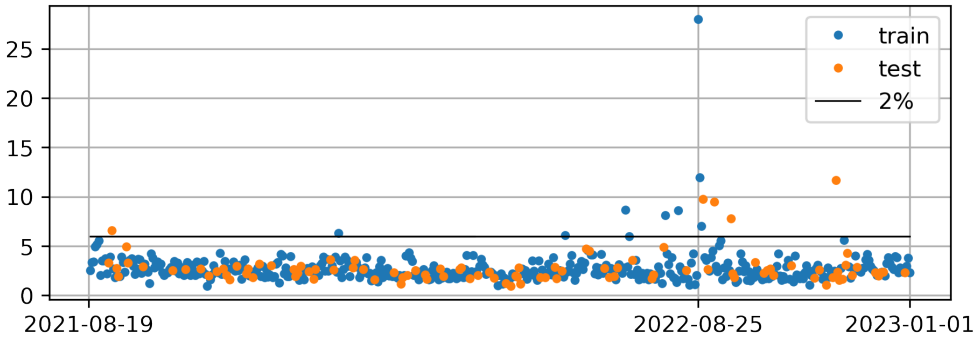

Рис. 2: Выбросы тестовой выборки

Это больше, чем 2% от 100 объектов. Тем не менее, хорошо видно, что эти объекты действительно далеко отстоят от остальных дат по величине своих реконструкционных ошибок. В силу настройки, полученной на обучающей выборке (была настроена высота горизонтальной линии отсечения), это действительно выбросы.

5 Выводы

Обнаружение выбросов обычно рассматривается как задача бинарной классификации, особенно когда исследователя интересует лишь определение, является ли точка выбросом или нет. В этом случае выбросы рассматриваются как класс «положительных» или аномальных точек, в то время как все остальные точки будут принадлежать классу «отрицательных» или не-выбросов. Для решения этой задачи используются различные алгоритмы машинного обучения для бинарной классификации, такие как логистическая регрессия, случайный лес, градиентный бустинг и так далее (см. [5]).

Однако в ситуации, когда изначальная разметка данных на нормальные и аномальные объекты отсутствует, классификация невозможна. Тогда необходимы другие методы, один из которых был рассмотрен в настоящей статье и показал на примере данных о потреблении контента одного из ведущих хостингов весьма неплохие результаты. Разумеется,

этот метод не является единственным в своем роде, но важно отметить, что обнаружение выбросов является субъективной задачей, и разные методы могут давать разные результаты. Поэтому всегда следует использовать комбинацию различных методов и подходов для достижения наилучших результатов.

6 Литература

1. 1.Хейдт М. Изучаем Pandas / М. Хейдт; — Москва: ДМК Пресс, 2018. — 438 с.
2. Бурков А. Машинное обучение без лишних слов / А. Бурков; — СПб: Питер, 2020. — 192 с.
3. Вьюгин, В. В. Математические основы теории машинного обучения и прогнозирования / В. В. Вьюгин; — М.: МЦИМО. — 2013. — 387 с.
4. Бринк Х. Машинное обучение / Х. Бринк, Дж. Ричардс, М. Феверолф — СПб.: Питер, 2017. — 336 с.
5. Дрейпер Н. Р. Прикладной регрессионный анализ / Дрейпер Н. Р., Смит Г.; ред. пер. Саит-Аметова М.; Пер. с англ. и ред. пер. Власенко М., Имамутдинова Р. Г., Орехова Н. А., Саит-Аметова М. — 3-е изд. — М. : Диалектика : Вильямс, 2007. — 911 с.
6. Безменов И. В. Метод очистки измерительных данных от выбросов: поиск оптимального решения с минимальным количеством отбракованных результатов измерений // Измерительная техника. 2023. № 1. С. 16–23.
7. Безменов И. В., Дроздов А. Э., Пасынок С. Л. Стратегия поиска выбросов в рядах зашумлённых данных с неизвестным трендом // Измерительная техника. 2022. № 5. С. 29–34.

-